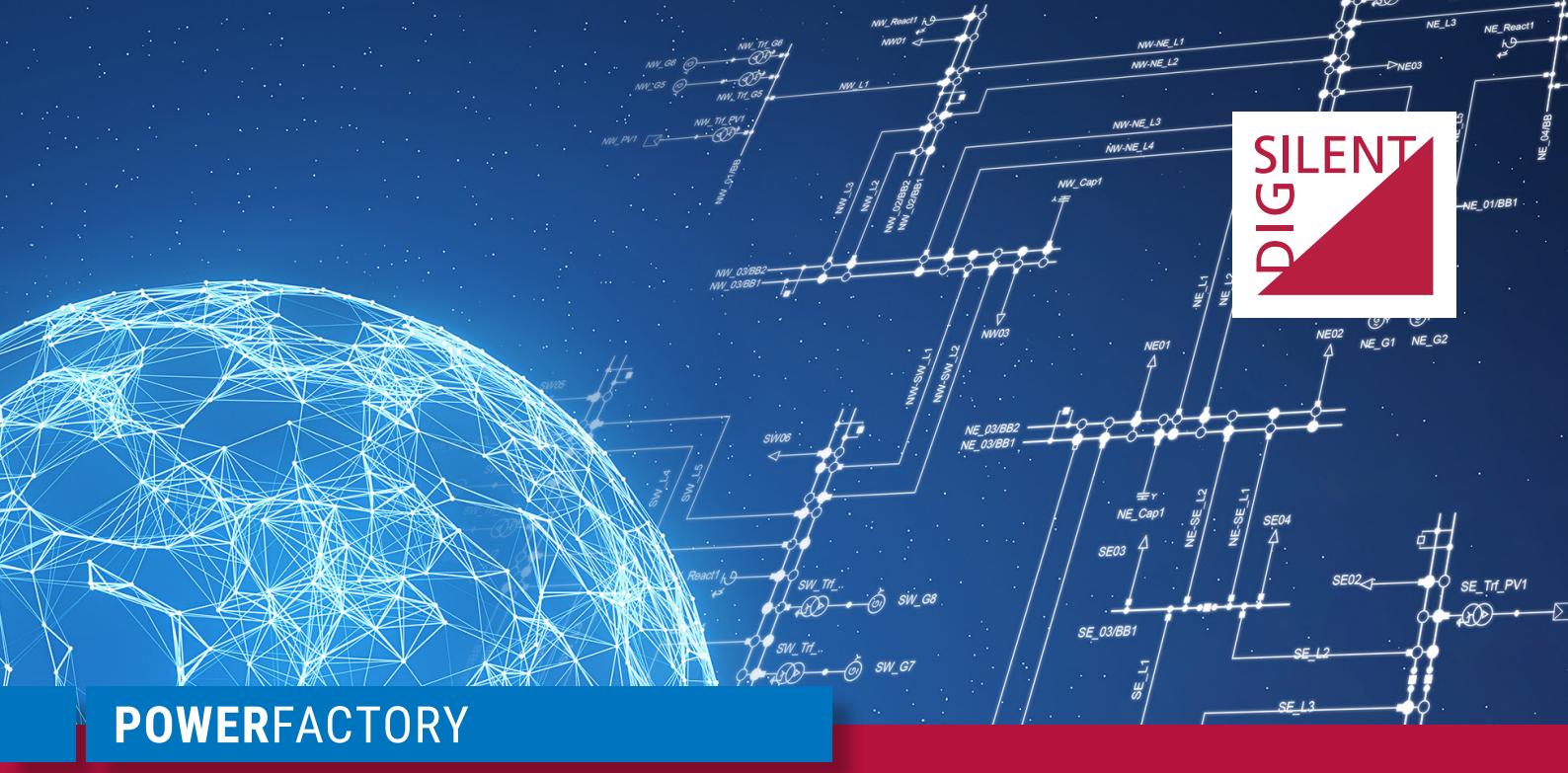


# PF2025



## PowerFactory 2025

### User Manual

**POWER SYSTEM SOLUTIONS**  
MADE IN GERMANY

# **DIGSILENT PowerFactory**

Version 2025

## **User Manual**

---

Online Edition

DIGSILENT GmbH

Gomaringen, Germany

April 2025

**Publisher:**  
DIgSILENT GmbH  
Heinrich-Hertz-Straße 9  
72810 Gomaringen / Germany  
Tel.: +49 (0) 7072-9168-0  
Fax: +49 (0) 7072-9168-88

Please visit our homepage at:  
<https://www.digsilent.de>

**Copyright © DIgSILENT GmbH**  
All rights reserved. No part of this  
publication may be reproduced or  
distributed in any form without written  
permission of the publisher.

April 2025  
r13243

# Contents

<b>I General Information</b>	<b>1</b>
<b>1 About this Guide</b>	<b>2</b>
1.1 Contents of the User Manual . . . . .	2
1.2 Used Conventions . . . . .	2
<b>2 Contact</b>	<b>3</b>
2.1 Direct Technical Support . . . . .	3
2.1.1 <i>PowerFactory</i> Support Package . . . . .	4
2.1.2 Licence Support Package . . . . .	4
2.2 Knowledge Base . . . . .	4
2.3 General Information . . . . .	5
<b>3 Documentation and Help System</b>	<b>6</b>
<b>4 PowerFactory Overview</b>	<b>8</b>
4.1 General Concept . . . . .	9
4.2 <i>PowerFactory</i> Simulation Functions . . . . .	10
4.3 General Design of <i>PowerFactory</i> . . . . .	12
4.4 Type and Element Data . . . . .	13
4.5 Data Arrangement . . . . .	14
4.5.1 <i>DlgSILENT</i> Library . . . . .	14
4.5.2 Custom Global Library . . . . .	15
4.5.3 Project Library . . . . .	15
4.5.4 Diagrams . . . . .	15
4.5.5 Network Data . . . . .	15
4.5.6 Variations . . . . .	16

## CONTENTS

---

4.5.7	Operation Scenarios . . . . .	16
4.5.8	Study Cases . . . . .	17
4.5.9	Settings . . . . .	17
4.6	Project Structure . . . . .	17
4.6.1	Nodes . . . . .	18
4.6.2	Edge elements . . . . .	18
4.6.3	Branches . . . . .	18
4.6.4	Cubicles . . . . .	18
4.6.5	Switches . . . . .	19
4.6.6	Substations . . . . .	19
4.6.7	Secondary Substations . . . . .	19
4.6.8	Sites . . . . .	19
4.6.9	Branch Elements . . . . .	19
4.7	User Interface . . . . .	19
4.7.1	Overview . . . . .	20
4.7.2	Desktop ( <i>SetDesktop</i> ) . . . . .	21
4.7.3	Tab groups ( <i>SetTabgroup</i> ) . . . . .	21
4.7.4	Key Features of the user interface . . . . .	22
4.7.5	Menu Bar . . . . .	24
4.7.6	Main Toolbar . . . . .	24
4.7.7	The Output Window . . . . .	28
4.7.8	Use of Colouring in <i>PowerFactory</i> . . . . .	30
4.8	Scripting in <i>PowerFactory</i> . . . . .	32
4.8.1	<i>DlgSILENT</i> Programming Language (DPL) Scripts . . . . .	32
4.8.2	Python Scripts . . . . .	33
<b>II</b>	<b>Administration</b>	<b>34</b>
<b>5</b>	<b><i>PowerFactory</i> Installation and Configuration</b>	<b>35</b>
5.1	Program Installation . . . . .	35
5.2	<i>PowerFactory</i> Configuration . . . . .	35
5.2.1	General Page . . . . .	36
5.2.2	Database Page . . . . .	36

5.2.3	Workspace Page . . . . .	36
5.2.4	External Applications Page . . . . .	36
5.2.5	Network Page . . . . .	37
5.2.6	Geographic Maps Page . . . . .	38
5.2.7	Advanced Page . . . . .	38
5.3	Licence . . . . .	39
5.3.1	Select Licence . . . . .	39
5.3.2	Activate / Update / Deactivate / Move Licence . . . . .	39
5.3.3	Licence and Maintenance Status . . . . .	40
5.4	Workspace Options . . . . .	40
5.4.1	Show Workspace Directory . . . . .	41
5.4.2	Import and Export Workspace . . . . .	41
5.4.3	Show Default Export Directory . . . . .	41
5.5	Database Migration . . . . .	41
<b>6</b>	<b>PowerFactory Administration</b>	<b>42</b>
6.1	<i>PowerFactory</i> Database Overview . . . . .	42
6.2	The Administrator Account . . . . .	43
6.3	Administration Menu . . . . .	44
6.3.1	User Management . . . . .	44
6.3.2	Security and Privacy . . . . .	44
6.3.3	Calculation Settings . . . . .	45
6.3.4	Housekeeping . . . . .	45
6.4	Creating and Managing User Accounts and Groups . . . . .	45
6.4.1	Users . . . . .	45
6.4.2	User Groups . . . . .	46
6.5	Security and Privacy . . . . .	47
6.5.1	Audit Log . . . . .	47
6.5.2	Login Policy Options . . . . .	48
6.5.3	Idle Session Timeout . . . . .	48
6.5.4	External Data Access . . . . .	48
6.5.5	File Exchange Settings . . . . .	48
6.5.6	Privacy . . . . .	49

## CONTENTS

---

6.6 Calculation Settings . . . . .	49
6.6.1 Parallel Computing Manager . . . . .	49
6.7 User Interface Customisation . . . . .	50
6.7.1 User-defined Tools Toolbar . . . . .	50
6.7.2 Profiles . . . . .	51
<b>7 User Settings</b>	<b>57</b>
7.1 Window Settings . . . . .	57
7.2 Data/Network Model Manager Settings . . . . .	57
7.3 Graphic Windows Settings . . . . .	59
7.3.1 General tab . . . . .	59
7.3.2 Advanced tab . . . . .	60
7.4 Output Window Settings . . . . .	60
7.5 Profile Settings . . . . .	61
7.6 Functions Settings . . . . .	61
7.7 Editor Settings . . . . .	62
7.8 Colours Settings . . . . .	62
7.9 <i>StationWare</i> Settings . . . . .	63
7.10 Offline Settings . . . . .	63
7.11 Parallel Computing . . . . .	64
7.12 Miscellaneous Settings . . . . .	64
7.13 Hotkeys and Quick Access . . . . .	65
7.13.1 Hotkeys . . . . .	65
7.13.2 Quick Access . . . . .	67
<b>8 Multi-User Database</b>	<b>69</b>
8.1 Housekeeping . . . . .	69
8.1.1 Introduction . . . . .	69
8.1.2 Configuring Permanently Logged-On Users . . . . .	69
8.1.3 Configuring Housekeeping Tasks . . . . .	70
8.1.4 Project Archiving . . . . .	70
8.1.5 Configuring Deletion of Old Projects . . . . .	70
8.1.6 Configuring Purging of Projects . . . . .	71

8.1.7	Configuring Emptying of Recycle Bins . . . . .	72
8.1.8	Delete orphaned archive files . . . . .	72
8.1.9	Purge of object keys . . . . .	72
8.1.10	Monitoring Housekeeping . . . . .	72
8.1.11	Summary of Housekeeping Deployment . . . . .	72
8.2	Offline Mode User Guide . . . . .	73
8.2.1	Functionality in Offline Mode . . . . .	73
8.2.2	Functionality in Online Mode . . . . .	76
8.2.3	Terminate Offline Session . . . . .	76
<b>III</b>	<b>Handling</b>	<b>77</b>
<b>9</b>	<b>Basic Project Definition</b>	<b>78</b>
9.1	Defining and Configuring a Project . . . . .	78
9.1.1	Project Directory . . . . .	79
9.1.2	Project Dialog . . . . .	80
9.1.3	Project Settings . . . . .	81
9.1.4	Activating and Deactivating Projects . . . . .	87
9.1.5	Exporting and Importing Projects . . . . .	87
9.1.6	External References . . . . .	91
9.1.7	Including Additional Documents . . . . .	91
9.2	Project Overview . . . . .	92
9.3	Creating New Grids . . . . .	93
<b>10</b>	<b>Network Graphics</b>	<b>94</b>
10.1	Introduction . . . . .	94
10.2	Graphic Windows and Database Objects . . . . .	94
10.2.1	Network Diagrams and other graphics . . . . .	94
10.2.2	Active Graphics, Study Case and Desktop . . . . .	95
10.2.3	Single Line Graphics and Data Objects . . . . .	96
10.2.4	Creating New Graphic Windows . . . . .	97
10.2.5	Page Tab . . . . .	97
10.2.6	Tab Group . . . . .	98

10.2.7	Drawing Tools . . . . .	98
10.2.8	Active Grid Folder (Target Folder) . . . . .	100
10.3	Graphic Commands and Settings . . . . .	100
10.3.1	Freeze Mode . . . . .	100
10.3.2	Rebuild . . . . .	100
10.3.3	View commands . . . . .	100
10.3.4	Select commands . . . . .	102
10.3.5	Diagram Settings . . . . .	102
10.3.6	Drawing Format . . . . .	106
10.3.7	Layers . . . . .	106
10.3.8	Show Legend Layer . . . . .	112
10.3.9	Show Result Layer . . . . .	113
10.3.10	Colouring Options . . . . .	113
10.3.11	Node Default Options . . . . .	117
10.3.12	Print and Export Options . . . . .	118
10.3.13	Diagram Layout Tool . . . . .	118
10.3.14	Measure Distance . . . . .	118
10.3.15	Context Menu Options . . . . .	118
10.4	Graphical Symbols of Elements . . . . .	120
10.4.1	Create a new symbol based on a library symbol . . . . .	120
10.4.2	Create a new symbol from the Data Manager . . . . .	122
10.5	Result Boxes, Text Boxes and Labels . . . . .	123
10.5.1	Result Boxes . . . . .	123
10.5.2	Text Boxes . . . . .	125
10.5.3	Labels . . . . .	125
10.5.4	Free Text Labels . . . . .	125
10.6	Annotations . . . . .	125
10.6.1	Annotation of Protection Device . . . . .	126
10.7	Command Buttons . . . . .	126
10.8	Geographic Diagrams . . . . .	127
10.8.1	Using an External Map Provider . . . . .	128
10.8.2	Using Local Maps . . . . .	132

10.9 Graphic Windows Hotkeys . . . . .	133
<b>11 Data Manager and Network Model Manager</b>	<b>135</b>
11.1 Introduction . . . . .	135
11.2 Data Manager and Network Model Manager Features . . . . .	136
11.2.1 Browser window layout . . . . .	136
11.2.2 Selecting objects and general object-specific actions . . . . .	138
11.2.3 Detail Mode and browser tabs . . . . .	138
11.2.4 Flexible Data Page . . . . .	139
11.2.5 Sorting . . . . .	140
11.2.6 Finding objects by name . . . . .	140
11.2.7 Auto-filter functions . . . . .	141
11.2.8 Editing objects from a Data Manager or Network Model Manager . . . . .	141
11.2.9 Other context menu options . . . . .	143
11.2.10 Display of multi-dimensional attributes . . . . .	143
11.2.11 Exporting and importing data in spreadsheet format . . . . .	145
11.3 Data Manager Features . . . . .	150
11.3.1 Address bar . . . . .	150
11.3.2 Navigating the database hierarchy . . . . .	151
11.3.3 Searching and filtering . . . . .	151
11.3.4 Adding, deleting and moving objects . . . . .	153
11.3.5 Data import and export . . . . .	155
11.3.6 Input Window . . . . .	156
11.4 Network Model Manager Features . . . . .	157
11.4.1 Object classification . . . . .	157
11.4.2 Saving of filters . . . . .	158
11.4.3 Scenario Manager . . . . .	158
<b>12 Building Networks</b>	<b>159</b>
12.1 Introduction . . . . .	159
12.2 Defining Network Models using the Graphical Editor . . . . .	159
12.2.1 Adding New Power System Elements . . . . .	159
12.2.2 Nodes . . . . .	160

12.2.3	Edge Elements . . . . .	160
12.2.4	Cubicles . . . . .	162
12.2.5	Marking and Editing Power System Elements . . . . .	162
12.2.6	Interconnecting Power Subsystems . . . . .	163
12.2.7	Substations . . . . .	165
12.2.8	Sites . . . . .	171
12.2.9	Composite Branches . . . . .	172
12.2.10	Single and Two Phase Elements . . . . .	173
12.3	Lines and Cables . . . . .	173
12.3.1	Defining a Line ( <i>ElmLne</i> ) . . . . .	174
12.3.2	Defining Line Sections . . . . .	175
12.3.3	Defining Line Compensation . . . . .	176
12.3.4	Defining Line Couplings . . . . .	176
12.3.5	Defining Cable Systems . . . . .	179
12.4	Neutral Winding Connection in Network Diagrams . . . . .	184
12.5	Defining Network Models using the Data Manager . . . . .	186
12.5.1	Defining New Network Components in the Data Manager . . . . .	186
12.5.2	Connecting Network Components in the Data Manager . . . . .	186
12.5.3	Defining Substations in the Data Manager . . . . .	187
12.5.4	Defining Composite Branches in the Data Manager . . . . .	187
12.5.5	Defining Sites in the Data Manager . . . . .	188
12.6	Drawing Existing Elements using the Diagram Layout Tool . . . . .	188
12.6.1	Action . . . . .	189
12.6.2	Node Layout . . . . .	191
12.6.3	Edge Elements . . . . .	192
12.6.4	Protection Devices . . . . .	193
12.6.5	Bays and Sites . . . . .	194
12.6.6	Miscellaneous . . . . .	194
12.6.7	Interchanges . . . . .	196
12.7	Other Options for Drawing Existing Elements . . . . .	196
12.7.1	Drawing Existing Elements using Drag & Drop . . . . .	196
12.7.2	Drawing Existing Stations using the Neighbourhood Expansion . . . . .	196

12.7.3 Single Line Diagrams Based on Geographic Diagrams . . . . .	197
<b>13 Study Cases</b>	<b>198</b>
13.1 Introduction . . . . .	198
13.2 Creating and Using Study Cases . . . . .	199
13.2.1 The Study Case Manager . . . . .	199
13.3 Summary Grid . . . . .	200
13.4 Study Time . . . . .	201
13.5 The Study Case Dialog . . . . .	201
13.5.1 Basic Data . . . . .	202
13.5.2 Calculation Options . . . . .	202
13.6 Variation Configuration . . . . .	203
13.7 Operation Scenarios . . . . .	203
13.8 Commands . . . . .	203
13.9 Events . . . . .	204
13.9.1 Broken Conductor Event . . . . .	204
13.9.2 Dispatch Event . . . . .	205
13.9.3 External Measurement Event . . . . .	205
13.9.4 Intercircuit Fault Event . . . . .	205
13.9.5 Load Event . . . . .	205
13.9.6 Message Event . . . . .	206
13.9.7 Outage Event . . . . .	206
13.9.8 Parameter Event . . . . .	206
13.9.9 Save Results . . . . .	206
13.9.10 Save Snapshot . . . . .	207
13.9.11 Short-Circuit Event . . . . .	207
13.9.12 Stop Event . . . . .	207
13.9.13 Switch Event . . . . .	207
13.9.14 Synchronous Machine Event . . . . .	208
13.9.15 Tap Event . . . . .	208
13.9.16 Power Transfer Event . . . . .	208
13.10 Simulation Scan . . . . .	208
13.11 Results Objects . . . . .	208

## CONTENTS

---

13.12 Triggers . . . . .	208
13.13 Desktop . . . . .	209
<b>14 Libraries</b>	<b>210</b>
14.1 Introduction . . . . .	210
14.2 <i>DlgSILENT</i> Library . . . . .	210
14.2.1 Versioning in the <i>DlgSILENT</i> Library . . . . .	211
14.3 Custom Global Library . . . . .	211
14.4 Project Library . . . . .	211
14.5 Equipment Type Library . . . . .	212
14.6 Operational Library . . . . .	214
14.6.1 Circuit Breaker Ratings . . . . .	215
14.6.2 Coincidence Definitions . . . . .	216
14.6.3 Demand Transfers . . . . .	216
14.6.4 Fault Cases and Fault Groups . . . . .	217
14.6.5 Capability Curves (Mvar Limit Curves) for Generators . . . . .	218
14.6.6 Planned Outages . . . . .	221
14.6.7 Planned Outages <i>IntOutage</i> . . . . .	221
14.6.8 QP-Curves . . . . .	223
14.6.9 QV-Curves . . . . .	223
14.6.10 Remedial Action Schemes (RAS) . . . . .	223
14.6.11 Running Arrangements . . . . .	223
14.6.12 Thermal Ratings . . . . .	226
14.6.13 V-Control-Curves . . . . .	227
14.7 Reports Library . . . . .	227
14.8 Scripts Library . . . . .	227
14.9 Templates Library . . . . .	228
14.9.1 General Templates . . . . .	229
14.9.2 Substation Templates . . . . .	232
14.9.3 Busbar Templates . . . . .	232
14.9.4 Composite Branch Templates . . . . .	232
14.10 Dynamic Models Library . . . . .	233
14.11 Quasi-Dynamic Models Library . . . . .	234

14.12 Protection Devices Library . . . . .	234
14.13 Harmonics Library . . . . .	234
<b>15 Grouping Objects</b>	<b>235</b>
15.1 Introduction . . . . .	235
15.2 Areas . . . . .	235
15.3 Virtual Power Plants . . . . .	236
15.3.1 Defining and Editing a New Virtual Power Plant . . . . .	237
15.3.2 Applying a Virtual Power Plant . . . . .	237
15.3.3 Inserting a Generator into a Virtual Power Plant . . . . .	237
15.4 Boundaries . . . . .	237
15.4.1 Boundary Definition Tool . . . . .	238
15.4.2 Element Boundary . . . . .	239
15.5 Circuits . . . . .	240
15.6 Feeders . . . . .	241
15.6.1 Feeder Tools . . . . .	243
15.7 Meteorological Stations . . . . .	247
15.8 Operators . . . . .	248
15.9 Owners . . . . .	248
15.10 Paths . . . . .	248
15.11 Routes . . . . .	250
15.12 Zones . . . . .	250
<b>16 Operation Scenarios</b>	<b>251</b>
16.1 Introduction . . . . .	251
16.2 Operation Scenarios Background . . . . .	251
16.3 How to use Operation Scenarios . . . . .	253
16.3.1 How to create an Operation Scenario . . . . .	253
16.3.2 How to save an Operation Scenario . . . . .	254
16.3.3 How to activate an existing Operation Scenario . . . . .	255
16.3.4 How to deactivate an Operation Scenario . . . . .	255
16.3.5 How to identify operational data parameters . . . . .	256
16.4 The Operation Scenario Manager . . . . .	257

## CONTENTS

---

16.4.1	Accessing the Operation Scenario Manager . . . . .	257
16.4.2	Selecting scenarios and setting up variable configurations . . . . .	258
16.4.3	Viewing and editing data within the Operation Scenario Manager . . . . .	260
16.5	Working with Operation Scenarios . . . . .	260
16.5.1	How to view objects missing from the Operation Scenario data . . . . .	261
16.5.2	How to compare the data in two operation scenarios . . . . .	261
16.5.3	How to view the non-default Running Arrangements . . . . .	261
16.5.4	How to transfer data from one Operation Scenario to another . . . . .	262
16.5.5	How to update the default data with operation scenario data . . . . .	263
16.5.6	How exclude a grid from the Operation Scenario data . . . . .	263
16.5.7	How to create a time-based Operation Scenario . . . . .	263
16.6	Advanced Configuration of Operation Scenarios . . . . .	265
16.6.1	How to change the automatic save settings for Operation Scenarios . . . . .	265
16.6.2	How to modify the data stored in Operation Scenarios . . . . .	265
<b>17</b>	<b>Network Variations and Expansion Stages</b>	<b>267</b>
17.1	Introduction . . . . .	267
17.2	Variations . . . . .	268
17.3	Expansion Stages . . . . .	269
17.4	The Study Time . . . . .	269
17.5	The Recording Expansion Stage . . . . .	270
17.6	The Variation Scheduler . . . . .	270
17.7	Variation and Expansion Stage Example . . . . .	271
17.8	The Variation Manager . . . . .	272
17.8.1	Stage Order . . . . .	273
17.8.2	Object Modifications . . . . .	274
17.8.3	Attribute Modifications . . . . .	275
17.8.4	Object States . . . . .	276
17.9	Variation and Expansion Stage Management . . . . .	277
17.9.1	Applying Changes from Expansion Stages . . . . .	277
17.9.2	Consolidating Variations . . . . .	277
17.9.3	Splitting Expansion Stages . . . . .	277
17.9.4	Comparing Variations and Expansion Stages . . . . .	278

17.9.5	Colouring Variations the Single Line Graphic . . . . .	278
17.9.6	Variation Conflicts . . . . .	278
17.9.7	Error Correction Mode . . . . .	278
<b>18</b>	<b>Parameter Characteristics, Load States, and Tariffs</b>	<b>280</b>
18.1	Introduction . . . . .	280
18.2	Parameter Characteristics . . . . .	280
18.2.1	Scales and Triggers . . . . .	281
18.2.2	Available Characteristics . . . . .	281
18.2.3	Usage . . . . .	282
18.2.4	Characteristic Curves . . . . .	282
18.2.5	Creating a Characteristic . . . . .	283
18.2.6	Time Characteristics . . . . .	284
18.2.7	Profile Characteristics . . . . .	291
18.2.8	Scaling Factor . . . . .	292
18.2.9	Linear Functions . . . . .	292
18.2.10	Vector Characteristics . . . . .	293
18.2.11	Matrix Parameter Characteristics . . . . .	294
18.2.12	Parameter Characteristics from Files . . . . .	295
18.2.13	Characteristic References . . . . .	295
18.2.14	Edit Characteristic Dialog . . . . .	296
18.2.15	Characteristics Tab in Data Filters . . . . .	296
18.2.16	Example Application of Characteristics . . . . .	297
18.3	Load States . . . . .	300
18.3.1	Creating Load States . . . . .	300
18.3.2	Viewing Existing Load States . . . . .	300
18.3.3	Load State Object Properties . . . . .	301
18.3.4	Example Load States . . . . .	301
18.4	Load Distribution States . . . . .	303
18.4.1	Creating Load Distribution States . . . . .	303
18.4.2	Viewing Existing Load Distribution States . . . . .	304
18.4.3	Load Distribution State Object Properties . . . . .	304
18.4.4	Example Load Distribution States . . . . .	304

## CONTENTS

---

18.5 Tariffs . . . . .	305
18.5.1 Defining Time Tariffs . . . . .	306
18.5.2 Defining Energy Tariffs . . . . .	307
<b>19 Reporting and Visualising Results</b> . . . . .	<b>308</b>
19.1 Introduction . . . . .	308
19.2 Result Boxes . . . . .	308
19.2.1 Editing Result Boxes . . . . .	308
19.3 Variable Selection . . . . .	310
19.3.1 Selection Page . . . . .	311
19.3.2 Editor Page . . . . .	314
19.3.3 Format/Header Page . . . . .	314
19.4 Documentation of Device Data . . . . .	315
19.4.1 Documentation of Device Data - Settings . . . . .	315
19.4.2 Documentation of Device Data - Filter/Annex . . . . .	316
19.5 Output Reports . . . . .	316
19.5.1 Report Generation Command . . . . .	317
19.5.2 PDF Viewer . . . . .	321
19.5.3 Report Manager . . . . .	322
19.5.4 Report Templates . . . . .	322
19.5.5 Report Designer . . . . .	325
19.6 Comparisons Between Calculations . . . . .	328
19.7 Results Objects . . . . .	328
19.7.1 Exporting Results . . . . .	330
19.7.2 Results in Database Format . . . . .	331
19.8 Plots . . . . .	333
19.8.1 Plot Area . . . . .	335
19.8.2 Data Series . . . . .	336
19.8.3 Complex Data Definition . . . . .	340
19.8.4 Axes and Gridlines . . . . .	342
19.8.5 Plot Legend . . . . .	344
19.8.6 Plots Toolbar . . . . .	345
19.8.7 Context Menu Tools . . . . .	352

19.8.8 User-Defined Styles . . . . .	353
19.8.9 Plot Types . . . . .	353
<b>20 Data Extensions</b>	<b>370</b>
20.1 Introduction . . . . .	370
20.2 Data Extension Configuration . . . . .	370
20.2.1 Creating Data Extensions . . . . .	370
20.2.2 User Defined Classes . . . . .	371
20.3 Using Data Extensions . . . . .	371
20.4 Sharing Data Extensions . . . . .	372
<b>21 Data Management</b>	<b>373</b>
21.1 Introduction . . . . .	373
21.2 Project Versions . . . . .	373
21.2.1 What is a Version? . . . . .	373
21.2.2 How to Create a Version . . . . .	374
21.2.3 How to Rollback a Project . . . . .	375
21.2.4 How to Check if a Version is the Base for a Derived Project . . . . .	375
21.2.5 How to Delete a Version . . . . .	376
21.3 Derived Projects . . . . .	376
21.3.1 Derived Projects Background . . . . .	376
21.3.2 How to Create a Derived Project . . . . .	378
21.4 Comparing and Merging Projects . . . . .	379
21.4.1 Compare and Merge Tool Background . . . . .	379
21.4.2 How to Merge or Compare Two Projects Using the Compare and Merge Tool . .	380
21.4.3 How to Merge or Compare Three Projects Using the Compare and Merge Tool .	380
21.4.4 Compare and Merge Tool Advanced Options . . . . .	382
21.4.5 Compare and Merge Tool 'diff browser' . . . . .	383
21.5 How to Update a Project . . . . .	389
21.5.1 Updating a Derived Project from a new Version . . . . .	389
21.5.2 Updating a Base Project from a Derived Project . . . . .	390
21.5.3 Tips for Working with the Compare and Merge Tool . . . . .	390
21.6 Sharing Projects . . . . .	391

## CONTENTS

---

21.7 Combining Projects . . . . .	392
21.7.1 Project Combination Assistant . . . . .	392
21.7.2 Project Connection Assistant . . . . .	393
21.7.3 Final Project State . . . . .	394
21.7.4 Project Normalisation . . . . .	395
21.8 Database Archiving . . . . .	395
<b>22 Task Automation</b>	<b>396</b>
22.1 Introduction . . . . .	396
22.2 Configuration of Task Automation . . . . .	396
22.2.1 Study Cases Page . . . . .	397
22.2.2 Tasks Page . . . . .	397
22.2.3 Parallel Computing Page . . . . .	398
22.2.4 Output Page . . . . .	400
22.3 Task Automation Results . . . . .	400
<b>23 Scripting</b>	<b>401</b>
23.1 The <i>DlgSILENT</i> Programming Language - DPL . . . . .	401
23.1.1 The Principle Structure of a DPL Command . . . . .	402
23.1.2 The DPL Command . . . . .	403
23.1.3 The DPL Script Editor . . . . .	405
23.1.4 The DPL Script Language . . . . .	406
23.1.5 Access to Other Objects . . . . .	410
23.1.6 Access to Locally Stored Objects . . . . .	412
23.1.7 Accessing the General Selection . . . . .	412
23.1.8 Accessing External Objects . . . . .	413
23.1.9 Remote Scripts and DPL Command Libraries . . . . .	414
23.1.10 DPL Functions and Subroutines . . . . .	417
23.1.11 Data Container Objects . . . . .	417
23.2 Tabular Reports . . . . .	418
23.2.1 Basic Structure of a Tabular Report . . . . .	419
23.2.2 The Tabular Report Command . . . . .	419
23.2.3 A minimal Tabular Report . . . . .	421

23.2.4	Handling different kinds of data . . . . .	421
23.2.5	Advanced Features . . . . .	422
23.2.6	Tabular Report Callback Script Reference . . . . .	426
23.3	Python . . . . .	428
23.3.1	Installation of a Python Interpreter . . . . .	428
23.3.2	The Python <i>PowerFactory</i> Module . . . . .	429
23.3.3	The Python Command ( <i>ComPython</i> ) . . . . .	430
23.3.4	Running <i>PowerFactory</i> in Non-interactive Mode . . . . .	432
23.3.5	Performance of Python Scripts . . . . .	433
23.3.6	Debugging Python Scripts . . . . .	433
23.3.7	Example of a Python Script . . . . .	434
23.4	Editor . . . . .	435
23.5	Add On Modules . . . . .	436
23.5.1	Add On Module framework . . . . .	436
23.5.2	Creating a new Add-on Module command . . . . .	437
23.5.3	Executing an Add-on Module command . . . . .	439
23.5.4	Adding Add On Modules to the User-Defined Tools toolbar . . . . .	440
23.6	Command Buttons . . . . .	440
<b>24</b>	<b>Interfaces</b>	<b>441</b>
24.1	Introduction . . . . .	441
24.2	DGS Interface . . . . .	441
24.2.1	DGS Interface Typical Applications . . . . .	442
24.2.2	DGS Structure (Database Schemas and File Formats) . . . . .	443
24.2.3	DGS Import . . . . .	443
24.2.4	DGS Export . . . . .	444
24.3	ANAREDE and ANAFAS Interface . . . . .	444
24.4	PSS/E File Interface . . . . .	444
24.4.1	PSS/E File Types and Versions . . . . .	445
24.4.2	Importing PSS/E Data . . . . .	445
24.4.3	Exporting a project to a PSS/E file . . . . .	446
24.5	PSS/U Interface . . . . .	447
24.5.1	Importing PSS/U Data . . . . .	447

## CONTENTS

---

24.6 PSS/ADEPT Import Converter . . . . .	448
24.7 NEPLAN Interface . . . . .	448
24.7.1 Importing NEPLAN Data . . . . .	448
24.8 INTEGRAL Interface . . . . .	450
24.8.1 Importing Integral Data . . . . .	450
24.8.2 Export Integral Data . . . . .	450
24.9 PSS SINCAL Interface . . . . .	451
24.9.1 Importing PSS SINCAL Data . . . . .	451
24.10 UCTE-DEF Interface . . . . .	451
24.10.1 Importing UCTE-DEF Data . . . . .	452
24.10.2 Exporting UCTE-DEF Data . . . . .	452
24.11 CIM Interface . . . . .	453
24.11.1 Importing CIM Data . . . . .	453
24.11.2 Exporting CIM Data . . . . .	454
24.12 CGMES Tools . . . . .	455
24.12.1 CIM Data Import . . . . .	455
24.12.2 CIM Data Export . . . . .	456
24.12.3 CIM to Grid Conversion . . . . .	456
24.12.4 Grid to CIM Conversion . . . . .	457
24.12.5 CIM Data Validation . . . . .	458
24.12.6 Import and Export of the EIC as additional parameter . . . . .	458
24.13 Functional Mock-Up Interface . . . . .	459
24.14 OPC Interface . . . . .	459
24.14.1 OPC Interface Typical Applications . . . . .	460
24.15 StationWare Interface . . . . .	460
24.15.1 About StationWare . . . . .	460
24.15.2 Component Architecture . . . . .	461
24.15.3 Fundamental Concepts . . . . .	462
24.15.4 Configuration . . . . .	466
24.15.5 Getting Started . . . . .	467
24.15.6 Description of the Menu and Dialogs . . . . .	477
24.16 API (Application Programming Interface) . . . . .	482

<b>IV Power System Analysis Functions</b>	<b>483</b>
<b>25 Load Flow Analysis</b>	<b>484</b>
25.1 Introduction . . . . .	484
25.2 Technical Background . . . . .	486
25.2.1 Network Representation and Calculation Methods . . . . .	488
25.3 Executing Load Flow Calculations . . . . .	491
25.3.1 Basic Options . . . . .	491
25.3.2 Active Power Control . . . . .	492
25.3.3 Advanced Options . . . . .	495
25.3.4 Calculation Settings . . . . .	498
25.3.5 Outputs . . . . .	502
25.3.6 Load/Generation Scaling . . . . .	503
25.4 Detailed Description of Load Flow Calculation Options . . . . .	504
25.4.1 Active and Reactive Power Control . . . . .	504
25.4.2 Voltage Dependency of Loads . . . . .	511
25.4.3 Feeder Load Scaling . . . . .	512
25.4.4 Temperature Dependency of Lines and Cables . . . . .	517
25.4.5 Load Flow initialisation and saving of results . . . . .	518
25.4.6 Load flow calculation for train simulation . . . . .	519
25.5 Results Analysis . . . . .	521
25.5.1 Viewing Results in the Single Line Diagram . . . . .	521
25.5.2 Flexible Data Page . . . . .	521
25.5.3 Standard Reports . . . . .	522
25.5.4 Diagram Colouring . . . . .	522
25.5.5 Load Flow Sign Convention . . . . .	523
25.5.6 Results for Unbalanced Load Flow Calculations . . . . .	523
25.5.7 Update Database . . . . .	524
25.6 Troubleshooting Load Flow Calculation Problems . . . . .	525
25.6.1 General Troubleshooting . . . . .	526
25.6.2 Data Model Problem . . . . .	527
25.6.3 Some Load Flow Calculation Messages . . . . .	527
25.6.4 Too many Inner Loop Iterations . . . . .	528

25.6.5 Too Many Outer Loop Iterations . . . . .	529
<b>26 Short-Circuit Analysis</b>	<b>532</b>
26.1 Introduction . . . . .	532
26.2 Technical Background . . . . .	533
26.2.1 The IEC 60909/VDE 0102 Part 0/DIN EN 60909-0 Method . . . . .	535
26.2.2 The ANSI Method . . . . .	538
26.2.3 The Complete Method . . . . .	540
26.2.4 The IEC 61363 Method . . . . .	542
26.2.5 The IEC 61660 (DC)/VDE0102 part 10(DC)/DIN EN 61660 Method . . . . .	543
26.2.6 The ANSI/IEEE 946 (DC) Method . . . . .	545
26.3 Executing Short-Circuit Calculations . . . . .	545
26.3.1 Toolbar/Main Menu Execution . . . . .	545
26.3.2 Context Menu Execution . . . . .	546
26.3.3 Faults on Busbars/Terminals . . . . .	546
26.3.4 Faults on Lines and Branches . . . . .	547
26.3.5 Multiple Faults Calculation . . . . .	547
26.4 Short-Circuit Calculation Options . . . . .	549
26.4.1 Basic Options . . . . .	549
26.4.2 Advanced Options . . . . .	556
26.4.3 VDE/IEC . . . . .	558
26.4.4 ANSI . . . . .	561
26.4.5 Complete Method . . . . .	562
26.4.6 IEC 61363 . . . . .	565
26.4.7 VDE/IEC (DC) . . . . .	565
26.4.8 ANSI (DC) . . . . .	566
26.4.9 Output/Results . . . . .	566
26.5 Results Analysis . . . . .	568
26.5.1 Viewing Results in the Single Line Diagram . . . . .	568
26.5.2 Flexible Data Page . . . . .	568
26.5.3 Predefined Reports . . . . .	568
26.5.4 Diagram Colouring . . . . .	569
26.6 Short-Circuit Calculation for Single Contingency . . . . .	569

26.7 Capacitive Earth-Fault Current . . . . .	570
<b>27 Contingency Analysis</b>	<b>572</b>
27.1 Introduction . . . . .	572
27.2 Short Overview . . . . .	572
27.2.1 Contingency Analysis Objects . . . . .	573
27.2.2 Results Recording . . . . .	574
27.2.3 Configuring Network Restoration . . . . .	574
27.2.4 Visualisation . . . . .	575
27.3 Contingency Analysis Toolbar . . . . .	575
27.3.1 Contingency Definition . . . . .	575
27.3.2 Contingency Analysis Command . . . . .	575
27.3.3 Contingency Comparison . . . . .	576
27.3.4 Show Contingencies . . . . .	576
27.3.5 Show Fault Cases / Groups . . . . .	576
27.3.6 Remedial Action Schemes . . . . .	576
27.3.7 Edit Results Variables . . . . .	576
27.3.8 Tracing Buttons . . . . .	576
27.3.9 Contingency Analysis Reports . . . . .	576
27.3.10 Load Contingency Analysis Results . . . . .	576
27.4 Command dialog and Options . . . . .	577
27.4.1 Basic Options . . . . .	577
27.4.2 Recording of Results . . . . .	578
27.4.3 Time Phases . . . . .	580
27.4.4 Effectiveness . . . . .	583
27.4.5 Time Sweep . . . . .	584
27.4.6 Topology . . . . .	585
27.4.7 Screening . . . . .	586
27.4.8 Output . . . . .	587
27.4.9 Linearised Calculation . . . . .	587
27.4.10 Parallel Computing . . . . .	587
27.4.11 Calculating an Individual Contingency . . . . .	588

## CONTENTS

---

27.4.12 Representing Contingency Situations Contingency Cases . . . . .	588
27.5 Reporting Results . . . . .	590
27.5.1 Predefined Reports . . . . .	590
27.5.2 Customised reports . . . . .	593
27.6 Trace Function for Multiple Time Phase and/or RAS . . . . .	593
27.7 Creating Contingencies . . . . .	594
27.7.1 Creating Contingencies Using the Contingency Definition Command . . . . .	594
27.7.2 Creating Contingencies Using Fault Cases and Groups . . . . .	596
27.7.3 Creating Dynamic Contingencies . . . . .	596
27.8 Fault Cases and Groups . . . . .	597
27.8.1 Browsing Fault Cases and Fault Groups . . . . .	598
27.8.2 Defining a Fault Case from Network Element(s) . . . . .	598
27.8.3 Defining Fault Cases using the Contingency Definition Command . . . . .	599
27.8.4 Representing Contingency Situations with Post-Fault Actions . . . . .	599
27.8.5 Defining a Fault Group . . . . .	600
27.9 Comparing Contingency Results . . . . .	600
27.10 Managing variables to be recorded . . . . .	602
27.10.1 Using filters to enable selective results recording . . . . .	602
27.11 Remedial Action Schemes (RAS) . . . . .	603
27.11.1 Creating a RAS object . . . . .	603
27.11.2 Trigger Conditions . . . . .	604
27.11.3 Logical combinations of triggers . . . . .	604
27.11.4 Remedial actions . . . . .	604
27.11.5 RAS groups . . . . .	605
27.11.6 Using Remedial Action Schemes in Contingency Analysis . . . . .	605
27.11.7 Results and reporting . . . . .	605
27.11.8 Visualising RAS using the Trace Function . . . . .	607
27.12 Load Contingency Analysis Results . . . . .	608
27.12.1 Load Individual Contingencies . . . . .	608
<b>28 Quasi-Dynamic Simulation</b>	<b>609</b>

---

28.1	Introduction	609
28.2	Technical background	609
28.3	How to execute a Quasi-Dynamic Simulation	611
28.3.1	Defining the variables for monitoring in the Quasi-Dynamic simulation	612
28.3.2	Considering maintenance outages	613
28.3.3	Considering simulation events	613
28.3.4	Running the Quasi-Dynamic simulation	615
28.3.5	Configuring the Quasi-Dynamic Simulation for parallel computation	616
28.3.6	Configure the Quasi-Dynamic Simulation for real time simulation	617
28.3.7	Use Neural Network approximation in the Quasi-Dynamic Simulation	617
28.4	Analysing the QDS results	617
28.4.1	Plotting	618
28.4.2	Quasi-Dynamic simulation reports	618
28.4.3	Statistical summary of monitored variables	618
28.4.4	Loading Results	619
28.5	Developing QDSL models	619
28.5.1	<i>PowerFactory</i> objects for implementing user defined models	620
28.5.2	QDSL Model Encryption	622
28.5.3	Overview of modelling approach	624
28.5.4	Algorithm flow of user defined Quasi-Dynamic models	624
28.5.5	Scripting Functions for Quasi-Dynamic Simulation	628
28.5.6	Example: Modelling a battery as a <i>Quasi-Dynamic</i> user defined model	633
<b>29</b>	<b>RMS/EMT Simulations</b>	<b>641</b>
29.1	Introduction	641
29.2	Calculation Methods	643
29.2.1	Balanced RMS Simulation	643
29.2.2	Three-Phase RMS Simulation	644
29.2.3	Three-Phase EMT Simulation	644
29.3	Calculation of Initial Conditions command	644
29.3.1	Initial Conditions - Basic Options	645
29.3.2	Initial Conditions - Step Size	648
29.3.3	Initial Conditions - Solver Options	650

29.3.4	Initial Conditions - Simulation Scan . . . . .	653
29.3.5	Initial Conditions - Noise Generation . . . . .	653
29.3.6	Initial Conditions - Real Time . . . . .	654
29.3.7	Initial Conditions - Snapshot . . . . .	655
29.3.8	Advanced Simulation Options - Load Flow . . . . .	655
29.4	Run Simulation command . . . . .	656
29.5	Results Objects . . . . .	658
29.5.1	Monitoring variables of an element . . . . .	659
29.5.2	Saving Results from Previous Simulations . . . . .	660
29.6	Simulation Events . . . . .	660
29.7	Executing the Simulation . . . . .	662
29.8	Creating Simulation Plots . . . . .	663
29.9	Simulation Scan . . . . .	663
29.9.1	Fault Ride Through Scan Module . . . . .	663
29.9.2	Frequency Scan Module . . . . .	666
29.9.3	Loss of Synchronism Scan Module . . . . .	667
29.9.4	Synchronous Machine Speed Scan Module . . . . .	668
29.9.5	Variable Scan Module . . . . .	668
29.9.6	Voltage Scan Module . . . . .	669
29.10	Save/Load Snapshot . . . . .	671
29.10.1	Saving a snapshot . . . . .	671
29.10.2	Loading a snapshot . . . . .	672
29.11	Single/Multiple Domain Co-simulation . . . . .	672
29.11.1	Overview of the Single/Multiple Domain Co-simulation . . . . .	672
29.11.2	Configuring the <i>Internal Co-simulation</i> . . . . .	674
29.11.3	Performing a Co-simulation and Retrieving Results . . . . .	677
29.11.4	The “ <i>Initial conditions for co-simulation</i> ” ( <i>ComCosim</i> ) Command . . . . .	677
29.11.5	Terms and Definitions for Co-simulation . . . . .	681
29.12	Co-simulation with External Application . . . . .	682
29.12.1	Overview of the Co-simulation with External Application . . . . .	682
29.12.2	Configuring the <i>Co-simulation with external application</i> . . . . .	683
29.12.3	Performing out a Co-simulation and Retrieving Results . . . . .	694

---

29.12.4 The “Preparation as FMU Agent” ( <i>ComCosimsetup</i> ) Command . . . . .	695
29.12.5 FMU Agent I/O and FMU Agent objects . . . . .	699
29.12.6 The “Initial conditions for co-simulation” ( <i>ComCosim</i> ) Command . . . . .	699
29.12.7 Terms and Definitions for Co-simulation with External Application . . . . .	700
29.13 Frequency Response Analysis . . . . .	702
29.13.1 Basic Usage . . . . .	702
29.13.2 Basic Data Page . . . . .	703
29.13.3 Output Page . . . . .	705
29.13.4 Advanced Page . . . . .	706
29.13.5 Output Plots . . . . .	706
29.13.6 Output Results Files . . . . .	706
29.13.7 Application notes and guidelines . . . . .	706
29.14 Frequency Analysis . . . . .	710
29.14.1 Prony Analysis Overview . . . . .	710
29.14.2 Basic Usage . . . . .	712
29.14.3 Basic Options Page . . . . .	712
29.14.4 FFT Page . . . . .	713
29.14.5 Prony Analysis Page . . . . .	714
29.14.6 Recalculation . . . . .	716
29.14.7 Output Plots . . . . .	716
29.14.8 Output Results Files . . . . .	717
29.14.9 General Recommendations on the Use of <i>Prony Analysis</i> . . . . .	718
29.14.10 Quick Overview of Used Formulas . . . . .	721
<b>30 Models for Dynamic Simulations</b> . . . . .	<b>723</b>
30.1 System Modelling Approach . . . . .	724
30.1.1 Overview of dynamic models in <i>PowerFactory</i> . . . . .	724
30.1.2 Model availability within <i>PowerFactory</i> : <i>Built-in</i> or <i>User-defined</i> models . . . . .	725
30.1.3 Externally interfaced versus <i>PowerFactory</i> native models . . . . .	726
30.1.4 Complete <i>Power Equipment</i> simulation models . . . . .	727
30.1.5 Compilation of <i>PowerFactory</i> native dynamic models . . . . .	727
30.2 High-level Control System Representation . . . . .	730
30.2.1 Data Structures for Dynamic Models within <i>PowerFactory</i> . . . . .	730

30.2.2	Composite Model Frames and Composite Models . . . . .	730
30.2.3	Creating a new <i>Composite Model Frame</i> . . . . .	733
30.2.4	Drawing a high-level control system in a <i>Composite Model Frame</i> . . . . .	734
30.2.5	Configuration of <i>Composite Model Frame</i> components . . . . .	737
30.2.6	Creating and configuring a <i>Composite Model</i> . . . . .	741
30.2.7	Array signal distribution/aggregation in high-level control systems . . . . .	744
30.2.8	Using power system measurements in a high-level control system . . . . .	762
30.3	DSL: Integrating <i>DSL Models</i> into a Simulation . . . . .	763
30.3.1	<i>DSL Models</i> and <i>DSL Model Types</i> . . . . .	763
30.3.2	Creating a <i>DSL Model</i> . . . . .	767
30.3.3	Saving and plotting model variables . . . . .	767
30.3.4	Example of a <i>DSL Model</i> implementation . . . . .	767
30.4	DSL: The <i>DlgSILENT Simulation Language</i> . . . . .	770
30.4.1	Introduction to DSL . . . . .	770
30.4.2	Structure of a dynamic model using DSL . . . . .	771
30.4.3	Terms and Abbreviations . . . . .	773
30.4.4	General DSL Syntax . . . . .	773
30.4.5	DSL Variables . . . . .	774
30.4.6	DSL Model Structure . . . . .	776
30.4.7	Initial Conditions . . . . .	776
30.4.8	Equation Code . . . . .	781
30.4.9	Various model definitions . . . . .	783
30.4.10	DSL Macros . . . . .	784
30.4.11	Events and Messages . . . . .	785
30.4.12	Advanced DSL Features . . . . .	786
30.4.13	Graphically defined DSL Model Types . . . . .	786
30.5	DSL: Overview of the <i>DSL Model Type</i> . . . . .	786
30.5.1	Creating a new <i>DSL Model Type</i> . . . . .	786
30.5.2	The Edit Dialog of the <i>DSL Model Type</i> . . . . .	786
30.6	DSL: Creating <i>DSL Model Types</i> using Block Diagrams . . . . .	794
30.6.1	Drawing Diagrams of <i>DSL Model Types</i> . . . . .	796
30.7	DSL: Coded <i>DSL Model Types</i> (non-graphically defined) . . . . .	802

---

30.8 Modelica: Integrating Modelica Models into a Simulation . . . . .	803
30.8.1 <i>Modelica Models</i> and <i>Modelica Model Types</i> . . . . .	803
30.8.2 Creating a new <i>Modelica Model</i> . . . . .	805
30.8.3 Saving and plotting model variables . . . . .	805
30.8.4 Support of <i>PowerFactory</i> functionalities . . . . .	805
30.8.5 Example of a <i>Modelica Model</i> implementation . . . . .	805
30.9 Modelica: A Non-proprietary, Object-oriented, Equation-based Language . . . . .	808
30.9.1 Modelica: Overview of an Open and Comprehensive Systems Modelling Lan- guage . . . . .	808
30.9.2 Supported <i>Modelica</i> functionality . . . . .	808
30.9.3 Terms and definitions . . . . .	808
30.10 Modelica: Creating Modelica Models using Block Diagrams . . . . .	811
30.10.1 Graphical environment for Modelica Models . . . . .	811
30.10.2 Developing Modelica Models . . . . .	814
30.10.3 Integrating Modelica Models into a <i>PowerFactory</i> simulation . . . . .	822
30.11 Modelica: Creating Modelica Models using Code . . . . .	826
30.11.1 Clocked Dynamic model definition . . . . .	826
30.11.2 Hybrid Dynamic model definition . . . . .	829
30.11.3 Dynamic model parameterisation . . . . .	831
30.12 Modelica: Overview of the <i>Modelica Model Type</i> . . . . .	831
30.12.1 Creating a new <i>Modelica Model Type</i> . . . . .	832
30.12.2 The Edit Dialog of the <i>Modelica Model Type</i> . . . . .	832
30.13 Modelica: Overview of the <i>Modelica Model (ElmMdl)</i> . . . . .	841
30.14 Interfaces for Dynamic Models . . . . .	843
30.14.1 Functional Mock-Up Interface (version 2.0) . . . . .	843
30.14.2 External C Interface acc. to IEC 61400-27 . . . . .	849
30.14.3 DSL-C Interface . . . . .	852
30.14.4 Management of compiled files of dynamic models . . . . .	854
30.14.5 MATLAB Interface . . . . .	856
30.15 Developing User-defined Power Electronics Models for EMT Simulation . . . . .	856
30.15.1 Model development as a <i>Submodel</i> of a built-in element . . . . .	857
30.15.2 Model development as an independent EMT model . . . . .	859

30.16 DSL Reference . . . . .	865
30.16.1 DSL Standard Functions . . . . .	865
30.16.2 DSL Special Functions . . . . .	866
30.16.3 DSL Global Library Macros . . . . .	883
<b>31 System Parameter Identification . . . . .</b>	<b>884</b>
31.1 Introduction . . . . .	884
31.2 Performing a Parameter Identification . . . . .	885
31.2.1 Basic Options . . . . .	885
31.2.2 Controls / Compared Signals . . . . .	886
31.2.3 PSO (Particle Swarm Optimisation) . . . . .	888
31.2.4 Nelder Mead . . . . .	888
31.2.5 DIRECT (DIviding RECTangle) . . . . .	888
31.2.6 Random Number Generation . . . . .	888
31.2.7 Gradient Calculation . . . . .	889
31.2.8 Stopping Criteria . . . . .	889
31.2.9 Output . . . . .	890
31.3 Algorithms . . . . .	891
31.3.1 Problem Description . . . . .	891
31.3.2 Particle Swarm Optimisation (PSO) . . . . .	891
31.3.3 Nelder Mead . . . . .	893
31.3.4 DIRECT . . . . .	894
31.3.5 BFGS . . . . .	895
31.3.6 Legacy (Quasi-Newton) . . . . .	896
31.4 What solver should I pick? (Pros and Cons of the different solvers) . . . . .	896
31.4.1 PSO - Particle Swarm Optimisation . . . . .	896
31.4.2 Nelder-Mead . . . . .	896
31.4.3 DIRECT . . . . .	896
31.4.4 BFGS . . . . .	896
31.4.5 Legacy (Quasi-Newton) . . . . .	897
31.5 What can I do if a solver has difficulties in finding good parameters? . . . . .	897
31.5.1 PSO - Particle Swarm Optimisation . . . . .	897
31.5.2 Nelder Mead . . . . .	897

---

31.5.3	DIRECT . . . . .	897
31.5.4	BFGS . . . . .	898
31.5.5	Legacy . . . . .	898
<b>32</b>	<b>Modal Analysis / Eigenvalue Calculation</b>	<b>899</b>
32.1	Theory of Modal Analysis . . . . .	899
32.2	How to Execute a Modal Analysis . . . . .	902
32.3	Modal/Eigenvalue Analysis Command . . . . .	902
32.3.1	Basic Options . . . . .	902
32.3.2	Algorithm . . . . .	904
32.3.3	Results . . . . .	905
32.3.4	Output . . . . .	906
32.4	Viewing Modal Analysis Results . . . . .	909
32.4.1	Modal/Eigenvalue Analysis Results Command . . . . .	910
32.4.2	Modal Analysis Results in Built-in Plots . . . . .	911
32.4.3	Eigenvalues Results in Single Line Diagrams . . . . .	917
32.4.4	Modal Analysis Results in the Output Window . . . . .	917
32.4.5	Modal Analysis Results in the Data Browser . . . . .	918
<b>33</b>	<b>Protection</b>	<b>920</b>
33.1	Introduction . . . . .	920
33.1.1	The modelling structure . . . . .	921
33.1.2	The relay frame . . . . .	922
33.1.3	The relay type . . . . .	922
33.1.4	The relay element . . . . .	923
33.2	How to define a protection scheme in <i>PowerFactory</i> . . . . .	924
33.2.1	Overview . . . . .	924
33.2.2	Adding protective devices to the network model . . . . .	925
33.2.3	Graphical representations of protection devices in single line diagrams . . . . .	927
33.2.4	Locating protection devices within the network model . . . . .	929
33.3	Basics of an overcurrent protection scheme . . . . .	930
33.3.1	Overcurrent relay model setup - basic data page . . . . .	930
33.3.2	Overcurrent relay model setup - max/min fault currents tab . . . . .	931

33.3.3	Configuring the current transformer . . . . .	932
33.3.4	Configuring the voltage transformer . . . . .	934
33.3.5	Configuring a combined Instrument transformer . . . . .	936
33.3.6	How to add a fuse to the network model . . . . .	937
33.3.7	Basic relay blocks for overcurrent relays . . . . .	938
33.4	The time-overcurrent plot . . . . .	943
33.4.1	How to create a time-overcurrent plot . . . . .	944
33.4.2	Understanding the time-overcurrent plot . . . . .	945
33.4.3	Showing the calculation results on the time-overcurrent plot . . . . .	945
33.4.4	Displaying the grading margins . . . . .	946
33.4.5	Adding a user defined permanent current line to the time-overcurrent plot . . . . .	947
33.4.6	Configuring the single line diagram associated with time-overcurrent plots . . . . .	947
33.4.7	Time-overcurrent plot configuration options . . . . .	947
33.4.8	Axes and Gridlines . . . . .	950
33.4.9	Plot Legend . . . . .	950
33.4.10	Altering protection device characteristic settings from the time-overcurrent plot .	951
33.4.11	How to split the relay/fuse characteristic . . . . .	951
33.4.12	Equipment damage curves . . . . .	954
33.5	Basics of a distance protection scheme . . . . .	965
33.5.1	Distance relay model setup - basic data page . . . . .	965
33.5.2	Primary or secondary Ohm selection for distance relay parameters . . . . .	965
33.5.3	Basic relay blocks used for distance protection . . . . .	966
33.6	The impedance plot (R-X diagram) . . . . .	973
33.6.1	How to create an R-X diagram . . . . .	973
33.6.2	Understanding the R-X diagram . . . . .	974
33.6.3	Modifying the relay settings and branch elements from the R-X plot . . . . .	975
33.6.4	R-X plot configuration options . . . . .	975
33.6.5	Adding user defined constants to the R-X plot . . . . .	979
33.6.6	Converting fault loop impedance markers generated from a calculation into permanently displayed impedance markers . . . . .	979
33.7	The relay operational limits plot (P-Q diagram) . . . . .	980
33.7.1	How to create a P-Q diagram . . . . .	980

---

33.7.2	Understanding the P-Q diagram . . . . .	980
33.7.3	Modifying the starting element settings from the P-Q plot . . . . .	981
33.7.4	Configuring the P-Q plot . . . . .	981
33.7.5	Adding user defined constants to the P-Q plot . . . . .	983
33.7.6	Converting P-Q markers generated from a load flow calculation into permanently displayed P-Q markers . . . . .	983
33.7.7	How to split a P-Q characteristic . . . . .	984
33.8	The time-distance plot . . . . .	984
33.8.1	Forward and reverse plots . . . . .	985
33.8.2	The path axis . . . . .	986
33.8.3	Methods of calculating tripping times . . . . .	986
33.8.4	Short-circuit calculation settings . . . . .	987
33.8.5	The distance axis units . . . . .	987
33.8.6	The reference relay . . . . .	988
33.8.7	Capture relays . . . . .	988
33.8.8	Double-click positions . . . . .	988
33.8.9	The context menu . . . . .	988
33.9	Basics of a differential protection scheme . . . . .	989
33.9.1	Differential relay model setup-basic data page . . . . .	989
33.9.2	Basic relay blocks used for differential protection . . . . .	989
33.10	Differential Plots . . . . .	990
33.10.1	Magnitude biased differential diagram . . . . .	990
33.10.2	Phase comparison differential diagram . . . . .	991
33.11	The Short-Circuit Sweep command . . . . .	992
33.12	Short-Circuit Sweep Plots . . . . .	994
33.12.1	Configuration of Short-Circuit Sweep plots . . . . .	995
33.13	Protection coordination assistant . . . . .	997
33.13.1	Technical background . . . . .	997
33.13.2	General Handling . . . . .	1000
33.13.3	Basic Options . . . . .	1001
33.13.4	Overcurrent Protection . . . . .	1002
33.13.5	Distance Protection . . . . .	1005

## CONTENTS

---

33.13.6 Grading Times . . . . .	1017
33.13.7 Advanced Options . . . . .	1017
33.13.8 Prerequisites for using the protection coordination tool . . . . .	1017
33.13.9 How to run the protection coordination calculation . . . . .	1018
33.13.10 How to output results from the protection coordination assistant . . . . .	1018
33.14 Accessing results . . . . .	1022
33.14.1 Quick access to protection plots . . . . .	1022
33.14.2 Tabular protection setting report . . . . .	1024
33.14.3 Results in single line graphic . . . . .	1026
33.15 Protection Audit . . . . .	1026
33.15.1 Protection Audit Command Handling . . . . .	1027
33.15.2 Protection Audit Results Command Handling . . . . .	1028
33.15.3 Report Handling and interpretation . . . . .	1032
33.16 Short circuit trace . . . . .	1035
33.16.1 Short Circuit Trace Handling . . . . .	1037
33.17 Protection Graphic Assistant . . . . .	1039
33.17.1 Reach Colouring . . . . .	1040
33.17.2 Short-Circuit Sweep Plot . . . . .	1042
33.17.3 Diagram Update . . . . .	1044
33.18 Building a basic overcurrent relay model . . . . .	1045
33.19 Appendix - other commonly used relay blocks . . . . .	1048
33.19.1 The frequency measurement block . . . . .	1048
33.19.2 The frequency block . . . . .	1049
33.19.3 The under-/overvoltage block . . . . .	1049
33.20 Relay block technical references . . . . .	1049
<b>34 Arc-Flash Hazard Analysis</b>	<b>1050</b>
34.1 Introduction . . . . .	1050
34.2 Arc-Flash Hazard Analysis Background . . . . .	1050
34.2.1 General . . . . .	1050
34.2.2 Determination of arc duration and consideration of arcing current . . . . .	1051
34.2.3 Input Data . . . . .	1052
34.2.4 Arc Models and Ranges of Calculation Models . . . . .	1054

34.3	Arc-Flash Hazard Analysis Calculation Options . . . . .	1057
34.3.1	Arc-Flash Hazard Analysis Basic Options Page . . . . .	1057
34.3.2	Arc-Flash Hazard Analysis Advanced Options Page . . . . .	1059
34.4	Arc-Flash Hazard Analysis Results . . . . .	1060
34.4.1	Viewing Results in the Single Line Graphic . . . . .	1060
34.4.2	Arc-Flash Reports . . . . .	1060
34.5	Example Arc-Flash Hazard Analysis Calculation . . . . .	1062
<b>35</b>	<b>Cable Analysis</b>	<b>1065</b>
35.1	Cable Sizing . . . . .	1066
35.1.1	Cable Models for Cable Sizing . . . . .	1066
35.1.2	Cable Sizing Calculation . . . . .	1070
35.1.3	Cable Sizing Results . . . . .	1077
35.2	Cable Ampacity . . . . .	1079
35.2.1	Cable Models for Cable Ampacity . . . . .	1080
35.2.2	Cable Ampacity Calculation Command . . . . .	1086
35.3	Cable Ampacity Report . . . . .	1088
<b>36</b>	<b>Power Quality and Harmonics Analysis</b>	<b>1089</b>
36.1	Introduction . . . . .	1089
36.2	Harmonic Load Flow . . . . .	1090
36.2.1	Basic Options Harmonic Load Flow . . . . .	1090
36.2.2	Results/Output . . . . .	1092
36.2.3	IEC 61000-3-6 . . . . .	1093
36.2.4	Advanced Options Harmonic Load Flow . . . . .	1093
36.2.5	Harmonic Load Flow Result Analysis . . . . .	1094
36.3	Frequency Sweep . . . . .	1099
36.3.1	Basic Options Frequency Sweep . . . . .	1100
36.3.2	Advanced Options Frequency Sweep . . . . .	1101
36.3.3	Frequency Sweep Result Analysis . . . . .	1102
36.4	Modelling Harmonic Sources . . . . .	1104
36.4.1	Definition of Harmonic Injections . . . . .	1105
36.4.2	Assignment of Harmonic Injections . . . . .	1110

## CONTENTS

---

36.4.3	Frequency Dependent Parameters . . . . .	1111
36.5	Flicker Analysis (IEC 61400-21) . . . . .	1112
36.5.1	Continuous Operation . . . . .	1113
36.5.2	Switching Operations . . . . .	1113
36.5.3	Flicker Contribution of Wind Turbine Generator Models . . . . .	1114
36.5.4	Definition of Flicker Coefficients . . . . .	1114
36.5.5	Assignment of Flicker Coefficients . . . . .	1115
36.5.6	Flicker Results Variables . . . . .	1116
36.6	Short-Circuit Power . . . . .	1116
36.6.1	Balanced Harmonic Load Flow . . . . .	1116
36.6.2	Unbalanced Harmonic Load Flow . . . . .	1116
36.6.3	Sk Result Variables . . . . .	1117
36.6.4	Short-Circuit Power of the External Grid . . . . .	1117
36.7	Definition of Result Variables . . . . .	1118
36.7.1	Definition of Variable Selections . . . . .	1119
36.7.2	Definition of Frequency Dependent Network Equivalent Data . . . . .	1119
<b>37 Connection Request Assessment</b>		<b>1121</b>
37.1	How to execute a Connection Request Assessment . . . . .	1121
37.2	Connection Request Assessment: D-A-CH-CZ . . . . .	1123
37.2.1	Basic Options D-A-CH-CZ . . . . .	1123
37.2.2	Advanced D-A-CH-CZ . . . . .	1125
37.3	Connection Request Assessment: BDEW, 4th Supplement . . . . .	1125
37.3.1	Basic Options BDEW . . . . .	1126
37.4	Connection Request Assessment: VDE-AR-N 4100/4105 . . . . .	1127
37.4.1	Basic Options VDE-AR-N 4100/4105 . . . . .	1128
37.5	Connection Request Assessment: VDE-AR-N 4110 . . . . .	1130
37.5.1	Basic Options VDE-AR-N 4110 . . . . .	1131
37.6	Connection Request Assessment Report . . . . .	1133
<b>38 Flickermeter</b>		<b>1134</b>
38.1	Introduction . . . . .	1134
38.2	Flickermeter (IEC 61000-4-15) . . . . .	1134

---

38.2.1	Calculation of Short-Term Flicker . . . . .	1134
38.2.2	Calculation of Long-Term Flicker . . . . .	1135
38.3	Flickermeter Calculation . . . . .	1135
38.3.1	Flickermeter Command . . . . .	1135
38.3.2	Data Source . . . . .	1135
38.3.3	Signal Settings . . . . .	1136
38.3.4	Advanced Options . . . . .	1137
38.3.5	Input File Types . . . . .	1138
38.3.6	How to use the Flickermeter . . . . .	1140
<b>39</b>	<b>Optimal Power Flow</b>	<b>1141</b>
39.1	Introduction . . . . .	1141
39.2	AC Optimisation (Interior Point Method) . . . . .	1141
39.2.1	AC Optimisation - Basic Options . . . . .	1142
39.2.2	Controls/Constraints . . . . .	1145
39.2.3	AC Optimisation - Initialisation . . . . .	1148
39.2.4	AC Optimisation - Algorithm . . . . .	1148
39.2.5	AC Optimisation - Iteration Control . . . . .	1149
39.2.6	AC Optimisation - Results/Output . . . . .	1150
39.3	DC Optimisation (Linear Programming) . . . . .	1152
39.3.1	DC Optimisation - Basic Options . . . . .	1152
39.3.2	DC Optimisation - Controls/Constraints . . . . .	1154
39.3.3	DC Optimisation - Algorithm . . . . .	1155
39.3.4	DC Optimisation - Iteration Control . . . . .	1155
39.4	Contingency Constrained DC Optimisation (LP Method) . . . . .	1156
39.4.1	Contingency Constrained DC Optimisation - Basic Options . . . . .	1157
39.4.2	Contingency Constrained DC Optimisation - Controls/Constraints . . . . .	1157
39.4.3	Contingency Constrained DC Optimisation - Algorithm . . . . .	1158
39.4.4	Contingency Constrained DC Optimisation - Iteration Control . . . . .	1158
39.4.5	Contingency Constrained DC Optimisation - Results/Output . . . . .	1158
39.5	Troubleshooting Optimal Power Flow Problems . . . . .	1159
39.5.1	Verification of Load Flow Options and Results . . . . .	1159
39.5.2	Verifications of OPF Constraints . . . . .	1160

---

39.5.3 Verification of the OPF Controls . . . . .	1160
39.5.4 Step-by-Step Approach . . . . .	1161
<b>40 Unit Commitment and Dispatch Optimisation</b>	<b>1162</b>
40.1 Introduction . . . . .	1162
40.2 Application Cases for the Unit Commitment . . . . .	1162
40.2.1 Full Unit Commitment . . . . .	1163
40.2.2 Market Simulation . . . . .	1163
40.2.3 Redispatch Calculation . . . . .	1163
40.3 Unit Commitment Command . . . . .	1164
40.3.1 Basic Options . . . . .	1164
40.3.2 Objective Function . . . . .	1164
40.3.3 Controls . . . . .	1166
40.3.4 Constraints . . . . .	1168
40.3.5 Results/Output . . . . .	1170
40.3.6 Maintenance . . . . .	1171
40.3.7 Algorithm . . . . .	1171
40.3.8 Parallel Computing . . . . .	1173
40.4 Handling of results . . . . .	1173
40.4.1 Reports . . . . .	1173
40.4.2 Plots . . . . .	1174
40.4.3 Colouring Mode . . . . .	1174
40.4.4 Load Unit Commitment Results . . . . .	1174
40.4.5 Flexible Data Page . . . . .	1175
40.5 Generating Units . . . . .	1175
40.5.1 Controls and Limits . . . . .	1175
40.5.2 Operating Costs . . . . .	1176
40.5.3 Reactive Power Costs . . . . .	1177
40.5.4 Redispatch Costs . . . . .	1178
40.5.5 Start-Up/Shut-down Costs . . . . .	1178
40.5.6 Constraints . . . . .	1179
40.6 Storage Units . . . . .	1179
40.6.1 Efficiency curves . . . . .	1180

---

40.7 Virtual Power Plants . . . . .	1180
40.8 Other Network Elements . . . . .	1181
40.8.1 Transformers, voltage regulators and shunts . . . . .	1181
40.8.2 Tap Controllers . . . . .	1182
40.8.3 Loads . . . . .	1182
40.8.4 Boundaries . . . . .	1182
40.8.5 Terminals . . . . .	1182
40.8.6 Lines and Bays . . . . .	1182
40.8.7 Regions: Grids, Zones and Areas . . . . .	1182
40.8.8 HVDC Converters . . . . .	1182
40.8.9 Advanced constraint settings . . . . .	1183
40.9 Troubleshooting . . . . .	1183
40.9.1 Solver Selection . . . . .	1183
40.9.2 Soft Constraints . . . . .	1183
40.9.3 Performance . . . . .	1183
40.9.4 Voltage Controlling Elements . . . . .	1184
40.9.5 Reference Machine . . . . .	1184
40.9.6 Not regarded Constraints . . . . .	1184
40.9.7 Rolling horizon and time dependencies . . . . .	1184
40.10 Redundant Constraint Filter . . . . .	1185
40.10.1 Basic Options . . . . .	1185
40.10.2 Algorithm . . . . .	1185
40.10.3 Results . . . . .	1187
<b>41 Transmission Network Tools</b>	<b>1188</b>
41.1 Introduction . . . . .	1188
41.2 PV Curves . . . . .	1189
41.2.1 PV Curves Calculation . . . . .	1189
41.2.2 PV Curves Plot . . . . .	1191
41.2.3 Outputs and Results . . . . .	1191
41.3 QV Curves . . . . .	1192
41.3.1 QV Curves Calculation . . . . .	1193
41.3.2 QV Curves Plot . . . . .	1195

## CONTENTS

---

41.4 Power Transfer Distribution Factors (PTDF) . . . . .	1196
41.4.1 Calculation Options . . . . .	1196
41.5 Transfer Capacity Analysis . . . . .	1198
41.5.1 Basic Data . . . . .	1198
41.5.2 Constraints . . . . .	1199
41.5.3 Output . . . . .	1200
41.5.4 Iteration Control . . . . .	1201
41.5.5 Advanced . . . . .	1201
41.6 Flow Decomposition . . . . .	1203
41.6.1 Technical Background . . . . .	1204
41.6.2 Flow Decomposition Command . . . . .	1205
41.6.3 Flow Decomposition Tabular Report . . . . .	1206
41.6.4 Flow Decomposition Plots . . . . .	1207
<b>42 Distribution Network Tools</b>	<b>1208</b>
42.1 Introduction . . . . .	1208
42.2 Hosting Capacity . . . . .	1209
42.2.1 Technical Background . . . . .	1210
42.2.2 Hosting Capacity Analysis Configuration . . . . .	1211
42.2.3 Results of the Hosting Capacity . . . . .	1218
42.3 Backbone Calculation . . . . .	1221
42.3.1 Basic Options Page . . . . .	1221
42.3.2 Scoring Settings Page . . . . .	1222
42.3.3 Tracing Backbones . . . . .	1223
42.3.4 Example Backbone Calculation . . . . .	1223
42.4 Voltage Sag . . . . .	1224
42.4.1 Calculation Options . . . . .	1224
42.4.2 How to Perform a Voltage Sag Table Assessment . . . . .	1225
42.4.3 Voltage Sag Table Assessment Results . . . . .	1226
42.5 Low Voltage Load Flow Calculation . . . . .	1227
42.5.1 Technical Background . . . . .	1228
42.5.2 Coincidence Definition . . . . .	1229
42.5.3 Setting up Low Voltage Loads . . . . .	1230

42.5.4	Setting up Generation Units . . . . .	1232
42.5.5	Low Voltage Load Flow Configuration . . . . .	1233
42.5.6	Results of the Low Voltage Load Flow Calculation . . . . .	1238
42.6	Tie Open Point Optimisation . . . . .	1238
42.6.1	Technical Background . . . . .	1239
42.6.2	How to run a Tie Open Point Optimisation . . . . .	1239
42.6.3	Results of the Tie Open Point Optimisation . . . . .	1244
42.7	Phase Balance Optimisation . . . . .	1245
42.7.1	Objective functions . . . . .	1246
42.7.2	Methods . . . . .	1246
42.7.3	Elements considered . . . . .	1247
42.7.4	Representation of solution . . . . .	1247
42.7.5	Output . . . . .	1247
42.8	Voltage Profile Optimisation . . . . .	1248
42.8.1	Optimisation Procedure . . . . .	1249
42.8.2	Basic Options Page . . . . .	1251
42.8.3	Output Page . . . . .	1252
42.8.4	Advanced Options Page . . . . .	1253
42.8.5	Results of Voltage Profile Optimisation . . . . .	1253
42.9	Optimal Equipment Placement . . . . .	1254
42.9.1	Optimal Equipment Placement Configuration . . . . .	1255
42.9.2	Results of the Optimal Equipment Placement . . . . .	1262
42.9.3	Troubleshooting . . . . .	1264
42.10	Optimal Capacitor Placement . . . . .	1265
42.10.1	OCP Objective Function . . . . .	1265
42.10.2	OCP Optimisation Procedure . . . . .	1267
42.10.3	Basic Options Page . . . . .	1267
42.10.4	Available Capacitors Page . . . . .	1269
42.10.5	Load Characteristics Page . . . . .	1269
42.10.6	Advanced Options Page . . . . .	1270
42.10.7	Results . . . . .	1271
42.11	Optimisation Algorithms . . . . .	1272

## CONTENTS

---

42.11.1 Genetic Algorithm . . . . .	1272
42.11.2 Simulated Annealing . . . . .	1274
<b>43 Outage Management</b>	<b>1275</b>
43.1 Introduction . . . . .	1275
43.2 Creating Planned Outages . . . . .	1275
43.2.1 Creating Planned Outages from Graphic or Network Model Manager . . . . .	1275
43.2.2 Creating Planned Outages in Data Manager . . . . .	1276
43.2.3 Recurrent Outages . . . . .	1276
43.2.4 Adding Additional Events to an Outage . . . . .	1277
43.3 Handling Planned Outages using the Outage Management Toolbox . . . . .	1278
43.3.1 Show Planned Outages . . . . .	1278
43.3.2 Apply Planned Outages . . . . .	1278
43.3.3 Reset All Planned Outages . . . . .	1278
43.3.4 Start Recording . . . . .	1279
43.3.5 Outage Schedule Report . . . . .	1279
<b>44 Economic Analysis Tools</b>	<b>1280</b>
44.1 Introduction . . . . .	1280
44.2 Techno-Economical Calculation . . . . .	1280
44.2.1 Technical Background . . . . .	1280
44.2.2 Techno-Economical Calculation Command . . . . .	1282
44.2.3 Handling of Results . . . . .	1286
44.2.4 Example Calculation . . . . .	1287
44.3 Techno-Economical Study Case Comparison . . . . .	1290
44.3.1 Additional Techno-Economical Calculation Quantities . . . . .	1290
44.3.2 Techno-Economical Study Case Comparison Command . . . . .	1291
44.3.3 Techno-Economical Study Case Comparison Example . . . . .	1292
44.4 Power Park Energy Analysis . . . . .	1294
44.4.1 Power Park Energy Analysis Configuration . . . . .	1294
44.4.2 Power Park Energy Analysis Report . . . . .	1298
44.4.3 Visualisation of Power Park Energy Analysis Results . . . . .	1299
<b>45 Probabilistic Analysis</b>	<b>1300</b>

---

45.1	Introduction	1300
45.2	Technical Background	1301
45.2.1	Distributions	1301
45.2.2	Modelling Dependencies	1304
45.2.3	Probabilistic Analysis Methods	1305
45.2.4	Statistics	1307
45.2.5	Distribution Estimation	1308
45.2.6	Distribution Fitting	1310
45.3	Object Settings	1311
45.3.1	Distributions	1311
45.3.2	Dependencies	1313
45.3.3	Distribution Estimation Command	1315
45.3.4	Probabilistic Analysis Command	1316
45.3.5	Continue Probabilistic Analysis	1317
45.3.6	Probabilistic Analysis Player	1317
45.3.7	Results File Handling	1317
45.3.8	Representation of results	1318
<b>46</b>	<b>Reliability Analysis</b>	<b>1322</b>
46.1	Introduction	1322
46.2	Technical Background	1324
46.2.1	Reliability Assessment Procedure	1325
46.2.2	Stochastic Models	1326
46.2.3	Calculated Results for Reliability Assessment	1327
46.2.4	System State Enumeration in Reliability Assessment	1333
46.3	Setting up the Network Model for Reliability Assessment	1334
46.3.1	Defining Stochastic Failure and Repair Models	1334
46.3.2	Defining Feeders for Reliability Assessment	1341
46.3.3	Configuring Switches for Reliability Assessment	1341
46.3.4	Load Modelling for Reliability Assessment	1343
46.3.5	Modelling Load Interruption Costs	1344
46.3.6	System Demand and Load States	1345
46.3.7	Defining Fault Clearance Based on Protection Device Location	1345

## CONTENTS

---

46.3.8	Considering Planned Maintenance . . . . .	1345
46.3.9	Specifying Individual Component Constraints . . . . .	1346
46.3.10	Considering Switching Rules . . . . .	1346
46.4	Running The Reliability Assessment Calculation . . . . .	1346
46.4.1	How to run the Reliability Assessment . . . . .	1346
46.5	Results of the Reliability Analysis . . . . .	1354
46.5.1	Load Reliability Results . . . . .	1354
46.5.2	Creating Reports using the Report Generation command . . . . .	1354
46.5.3	Tabular report of Contributions . . . . .	1355
46.5.4	Selecting loads for post-processing . . . . .	1356
46.5.5	Viewing Results in the Single Line Diagram . . . . .	1356
46.5.6	Viewing Results in the Data Browser . . . . .	1358
46.6	Loss of Grid Assessment . . . . .	1359
46.6.1	Basic Options . . . . .	1359
46.6.2	Protection . . . . .	1360
46.6.3	Output . . . . .	1360
46.6.4	Results . . . . .	1360
<b>47</b>	<b>Optimal Power Restoration</b> . . . . .	<b>1362</b>
47.1	Failure Effect Analysis . . . . .	1362
47.2	Animated Tracing of Individual Cases . . . . .	1367
47.3	Optimal RCS Placement . . . . .	1368
47.3.1	Basic Options Page . . . . .	1368
47.3.2	Output Page . . . . .	1369
47.3.3	Advanced Options Page . . . . .	1369
47.3.4	Example Optimal RCS Calculation . . . . .	1370
47.4	Optimal Manual Restoration . . . . .	1370
47.4.1	OMR Calculation Prerequisites . . . . .	1371
47.4.2	Basic Options Page . . . . .	1371
47.4.3	Advanced Options Page . . . . .	1372
47.4.4	Definition of the objective function . . . . .	1373
47.4.5	Example of an Optimal Manual Restoration Calculation . . . . .	1374
47.5	Optimal Recloser Placement . . . . .	1376

---

47.5.1	Technical Background . . . . .	1376
47.5.2	Basic Options . . . . .	1377
47.5.3	Results . . . . .	1378
47.5.4	Optimal Recloser Placement Reports . . . . .	1378
<b>48</b>	<b>Contingency Restoration Analysis</b>	<b>1380</b>
48.1	Introduction . . . . .	1380
48.2	Technical Background . . . . .	1380
48.3	Contingency Restoration Analysis Toolbar . . . . .	1381
48.3.1	Contingency Restoration Analysis Command . . . . .	1382
48.3.2	Show Contingencies . . . . .	1382
48.3.3	Edit Results Variables . . . . .	1382
48.3.4	Tracing Buttons . . . . .	1382
48.3.5	Contingency Restoration Analysis Reports . . . . .	1382
48.4	Command Dialog, Options and Switch Configuration . . . . .	1382
48.4.1	Basic Options page . . . . .	1383
48.4.2	Protection page . . . . .	1384
48.4.3	Restoration page . . . . .	1384
48.4.4	Constraints page . . . . .	1385
48.4.5	Results page . . . . .	1386
48.4.6	Configuring Switches for the Contingency Restoration Analysis . . . . .	1387
48.5	Managing variables to be recorded . . . . .	1387
48.6	Trace Function . . . . .	1388
48.7	Result Analysis . . . . .	1388
48.7.1	Diagram Colouring . . . . .	1388
48.7.2	Predefined Reports . . . . .	1389
48.7.3	Customised reports . . . . .	1391
<b>49</b>	<b>Generation Adequacy Analysis</b>	<b>1392</b>
49.1	Introduction . . . . .	1392
49.2	Technical Background . . . . .	1392
49.3	Database Objects and Models . . . . .	1395
49.3.1	Stochastic Model for Generation . . . . .	1395

## CONTENTS

---

49.3.2	Power Curve Type . . . . .	1396
49.3.3	Meteorological station . . . . .	1396
49.4	Assignment of Stochastic Model for Generation . . . . .	1397
49.4.1	Definition of a Stochastic Multi-State Model . . . . .	1397
49.4.2	Stochastic Wind Model . . . . .	1398
49.4.3	Time Series Characteristic for Wind Generation . . . . .	1398
49.4.4	Demand definition . . . . .	1400
49.5	Generation Adequacy Analysis toolbar . . . . .	1400
49.5.1	Generation Adequacy Initialisation command . . . . .	1400
49.5.2	Run Generation Adequacy command . . . . .	1402
49.6	Generation Adequacy results . . . . .	1403
49.6.1	Distribution (Cumulative Probability) Plots . . . . .	1403
49.6.2	Monte-Carlo Draws (Iterations) Plots . . . . .	1405
49.6.3	Convergence Plots . . . . .	1406
49.6.4	Summary of variables calculated during the Generation Adequacy Analysis . . .	1408
<b>50</b>	<b>Sensitivities / Distribution Factors</b>	<b>1409</b>
50.1	Overview of Sensitivity / Distribution Factors Calculations . . . . .	1409
50.1.1	Terminology . . . . .	1409
50.1.2	Result Quantities . . . . .	1410
50.2	Sensitivities / Distribution Factors Options . . . . .	1410
50.2.1	Basic Options . . . . .	1410
50.2.2	Results . . . . .	1411
50.2.3	Advanced Options . . . . .	1413
50.2.4	Output . . . . .	1415
50.2.5	Modal/Eigenvalue Analysis . . . . .	1415
50.3	Reporting . . . . .	1415
50.3.1	General . . . . .	1415
50.3.2	Thresholds . . . . .	1416
50.3.3	Used Format . . . . .	1416
50.3.4	Report output . . . . .	1416
50.4	Troubleshooting . . . . .	1416

---

<b>51 Network Reduction</b>	<b>1418</b>
51.1 Introduction . . . . .	1418
51.2 Technical Background . . . . .	1419
51.2.1 Network Reduction using Ward method . . . . .	1419
51.2.2 Network Reduction using REI Method . . . . .	1419
51.2.3 Network Reduction using Regional Equivalents . . . . .	1420
51.2.4 Network Reduction for Dynamic Equivalent . . . . .	1420
51.3 How to Carry Out a Network Reduction . . . . .	1420
51.3.1 How to Backup the Project (optional) . . . . .	1421
51.3.2 How to run the Network Reduction tool . . . . .	1421
51.3.3 Expected Output of the Network Reduction . . . . .	1421
51.4 Network Reduction Command . . . . .	1423
51.4.1 Basic Options . . . . .	1423
51.4.2 Ward Equivalent . . . . .	1424
51.4.3 REI Equivalent . . . . .	1424
51.4.4 Regional Equivalent . . . . .	1427
51.4.5 Dynamic Equivalent . . . . .	1427
51.4.6 Outputs . . . . .	1429
51.4.7 Advanced Options . . . . .	1429
51.4.8 Verification . . . . .	1430
51.5 Network Reduction Example . . . . .	1431
51.6 Tips for using the Network Reduction Tool . . . . .	1432
51.6.1 Network Reduction doesn't Reduce Isolated Areas . . . . .	1433
51.6.2 The Reference Machine is not Reduced . . . . .	1433
<b>52 State Estimation</b>	<b>1434</b>
52.1 Introduction . . . . .	1434
52.2 Objective Function . . . . .	1435
52.3 Components of the <i>PowerFactory</i> State Estimation . . . . .	1436
52.3.1 Plausibility Check . . . . .	1437
52.3.2 Observability Analysis . . . . .	1437
52.3.3 Non-linear optimisation (state estimation) . . . . .	1438
52.4 State Estimation Data Input . . . . .	1439

---

---

## CONTENTS

---

52.4.1	Measurements . . . . .	1439
52.4.2	Editing the Element Data . . . . .	1444
52.5	Running SE . . . . .	1445
52.5.1	Basic Setup Options . . . . .	1445
52.5.2	Bad Data Detection . . . . .	1449
52.5.3	Non-Linear optimisation (state estimation) . . . . .	1449
52.5.4	Results/Output . . . . .	1451
52.6	Results . . . . .	1451
52.6.1	Output Window Report . . . . .	1451
52.6.2	External Measurements . . . . .	1452
52.6.3	Estimated States . . . . .	1454
52.6.4	Colour Representation . . . . .	1454
<b>53</b>	<b>Motor Starting</b>	<b>1456</b>
53.1	Introduction . . . . .	1456
53.2	How to define a motor . . . . .	1456
53.2.1	How to define a motor Type and starting methodology . . . . .	1456
53.2.2	How to define a motor driven machine . . . . .	1458
53.3	How to run a Motor Starting simulation . . . . .	1458
53.3.1	Basic Options Page . . . . .	1458
53.3.2	Output Page . . . . .	1460
53.3.3	Motor Starting simulation results . . . . .	1462
53.3.4	Motor Starting Example . . . . .	1463
<b>54</b>	<b>Artificial Intelligence</b>	<b>1466</b>
54.1	Introduction . . . . .	1466
54.2	Technical Requirements . . . . .	1467
54.3	Technical Background . . . . .	1467
54.3.1	Neural Network Architecture . . . . .	1468
54.4	How to create and use a Neural Network . . . . .	1468
54.5	Setup for Neural Network command . . . . .	1469
54.5.1	Basic Options . . . . .	1470
54.5.2	Input Variables . . . . .	1470

---

54.5.3	Output Variables . . . . .	1470
54.6	Neural Network Training command . . . . .	1470
54.6.1	Basic Options . . . . .	1470
54.6.2	Data Generation . . . . .	1471
54.6.3	Neural Network Training . . . . .	1472
54.6.4	Random Number Generation . . . . .	1473
54.7	Output of Neural Network Training . . . . .	1473
54.7.1	Dataset Object . . . . .	1473
54.7.2	Neural Network Object . . . . .	1473
54.7.3	Consistency Check . . . . .	1474
54.8	Trouble Shooting for Neural Networks . . . . .	1474
54.8.1	The Probabilistic Analysis does not converge. . . . .	1474
54.8.2	The validation error does not decrease during the epochs. . . . .	1475
54.8.3	The approximation of the neural network is not good enough . . . . .	1475
<b>V</b>	<b>Appendix</b>	<b>1477</b>
<b>55</b>	<b>The <i>DlgsILENT</i> Output Language</b>	<b>1478</b>
55.1	Format string, Variable names and text Lines . . . . .	1479
55.2	Placeholders . . . . .	1479
55.3	Variables, Units and Names . . . . .	1480
55.4	Colour . . . . .	1482
55.5	Advanced Syntax Elements . . . . .	1482
55.6	Line Types and Page Breaks . . . . .	1483
55.7	Predefined Text Macros . . . . .	1483
55.8	Object Iterations, Loops, Filters and Includes . . . . .	1484
<b>56</b>	<b>Standard Functions DPL and DSL</b>	<b>1485</b>
<b>Bibliography</b>		<b>1487</b>
<b>Glossary</b>		<b>1491</b>

# **Part I**

# **General Information**

# Chapter 1

## About this Guide

This User Manual is intended to be a reference for users of the DIgSILENT PowerFactory software. This chapter provides general information about the contents and the used conventions of this documentation.

### 1.1 Contents of the User Manual

The first section of the User Manual provides General Information, including an overview of *PowerFactory* software, a description of the basic program settings, and a description of the *PowerFactory* data model.

The next sections describe *PowerFactory* administration, handling, and power system analysis functions. In the Power System Analysis Functions section, each chapter deals with a different calculation, presenting the most relevant theoretical aspects, the *PowerFactory* approach, and the corresponding interface.

The online version of this manual includes additional sections dedicated to the mathematical description of models and their parameters, referred to as Technical References. To facilitate their portability, visualisation, and printing, the papers are attached to the online help as PDF documents. They are opened by clicking on the indicated links within the manual.

It is recommended that new users commence by reading Chapter 4 (*PowerFactory* Overview), and completing the *PowerFactory* Tutorials.

### 1.2 Used Conventions

Conventions to describe user actions are as follows:

**Buttons and Keys** Dialog buttons and keyboard keys are referred to with bold and underline text formatting. For example, press the **OK** button in the *PowerFactory* dialog, or press **CTRL+B** on the keyboard.

**Menus and Icons** Menus and icons are usually referenced using *Italics*. For example, press the *User Settings* icon , or select *Tools* → *User Settings...*

**Other Items** “Speech marks” are used to indicate data to be entered by the user, and also to refer to an item defined by the author. For example, consider a parameter “x”.

# Chapter 2

## Contact

For further information about the company *DIGSILENT*, our products and services please visit our web site, or contact us at:

**DIGSILENT GmbH**

Heinrich-Hertz-Str. 9

72810 Gomaringen / Germany

[www.digsilent.de](http://www.digsilent.de)

### 2.1 Direct Technical Support

*DIGSILENT* experts offer direct assistance to *PowerFactory* users with valid maintenance agreements via telephone or online via support queries raised on the customer portal.

To register for the on-line portal, select *Help* → *Online User Registration...* or go to directly to the registration page (link below). Log-in details will be provided by email shortly thereafter.

To log-in to the portal, enter the email (or Login) and Password provided. When raising a new support query, please include the *PowerFactory* version and build number in your submission, which can be found by selecting *Help* → *About PowerFactory...* from the main menu. Note that including relevant \*.pfd or \*.pfds file(s) may assist with our investigation into your query. The customer portal is shown in Figure 2.1.1.

**Phone:** +49-(0)7072-9168-50 (German)  
+49-(0)7072-9168-51 (English)

**User Registration:** <https://www.digsilent.de/en/user-registration.html>

**User Login:** <https://www.digsilent.de/en/user-login.html>

New Incident

Incident Type: Question/Support request

Title: Line loadings

Product: PowerFactory

Version: 2024

Service Pack: SP1

External ID:

Description: Dear PowerFactory Support,  
In my model it is not clear why Line A and Line B are not loaded equally. They have the same type and length. Could you please assist? I attach my project.  
Regards,  
A. User

Please rate the quality of DigSILENT's support:

Attachments

LineLoadings.pdf

Figure 2.1.1: DlgSILENT customer portal

### 2.1.1 PowerFactory Support Package

Sometimes as part of a ticket investigation customers are asked to provide a “PowerFactory Support Package”. This can be generated from within *PowerFactory* from the top menu by doing *Help → Support → Create Support Package...*. The support package can be saved on the user’s computer then attached to the ticket. The contents of the “PowerFactory Support Package” are listed in the [Advanced Installation and Configuration Manual](#).

### 2.1.2 Licence Support Package

If a customer’s problem is thought to be licence related, the customer may be asked to provide a “Licence Support Package”. This can be generated from the Licence Manager, where a **Create Licence Support Package** button is provided. The support package can then be saved on the user’s computer then attached to the ticket.

## 2.2 Knowledge Base

A “Knowledge Base” database of information, based on an FAQ format, is available for any users (whether registered or not) to look for answers to their questions. The knowledge base contains interesting questions and answers regarding specific applications of *PowerFactory*.

**Knowledge Base:** <https://www.digsilent.de/en/faq-powerfactory.html>

## 2.3 General Information

For general information about *DIGSILENT* or your *PowerFactory* licence, please contact us via:

**Phone:** +49-(0)7072-9168-0

**Fax:** +49-(0)7072-9168-88

**E-mail:** mail@digsilent.de

## Chapter 3

# Documentation and Help System

*PowerFactory* is provided with a complete help package to support users at all levels of expertise. Documents with the basic information of the program and its functionality are combined with references to advanced simulation features, mathematical descriptions of the models and of course application examples.

*PowerFactory* offers the following help resources, which can be accessed either directly from *PowerFactory* or from the *DlgSILENT* download area (<https://www.digsilent.de/en/downloads.html>):

- **Getting Started:** a document describing the first steps to follow after receiving the installation DVD or downloading the software from the *DlgSILENT* download area. The Getting Started document covers the basic installation options.
- **Advanced Installation and Configuration Manual:** in this document, advanced installation options e.g. multi-user database, installation on an application server, and the Offline mode installation, are covered. The Offline mode guide is available in Section 8.2: Offline Mode User Guide.
- **Tutorials:** information for new users and hands-on tutorials. Access via Help menu of *PowerFactory*.
- **User Manual:** this document. Access via Help menu of *PowerFactory*. Current and previous manuals (PDF files) can also be found on the in the *DlgSILENT* download area.
- **Technical References:** description of the models implemented in *PowerFactory* for the different power systems components. The technical reference documents are available on the menu *Help* → *Technical References*.
- **Additional Packages:** additional information and/or examples about specific *PowerFactory* functions are available on the menu *Help* → *Additional Packages*. The additional packages are:
  - Programming Interface (API)
  - DGS Data Exchange Format
  - OPC Interface
  - DPL Functions Extensions
  - Interfaces for Dynamic Models
- **Context Sensitive Help:** pressing the key **F1** while working with *PowerFactory* will lead directly to the related topic inside the User Manual.
- **PowerFactory Examples:** the window *PowerFactory Examples* provides a list of application examples of *PowerFactory* calculation functions. Every example comes with an explanatory document that can be opened by clicking on the *Show Documentation* button (book icon). Additionally, videos demonstrating the software handling and its functionalities are available.

---

The *PowerFactory* Examples window will “pop up” automatically every time the software is open, this could be deactivated by unchecking the *Show at Startup* checkbox. *PowerFactory* Examples are also accessible on the main menu, by selecting *File* → *Examples*....

- **Release Notes:** for all new versions and updates of the program *Release Notes* are provided, which document the implemented changes. They are available on the menu *Help* → *Release Notes*
- **Knowledge base:** accessible via the menu *Help* → *Support*→ *Online Knowledge Base (FAQ)*, described in Chapter 2: Contact
- **Technical Support:** accessible via the menu *Help* → *Support*→ *Online Support Centre*, described in Chapter 2: Contact
- **Website:** [www.digsilent.de](http://www.digsilent.de)
- **Social Media:**
  - [YouTube](#)
  - [Facebook](#)
  - [Twitter](#)
  - [LinkedIn](#)

## Chapter 4

# **PowerFactory Overview**

The calculation program *PowerFactory* from *DIGSILENT*, is a computer-aided engineering tool for the analysis of transmission, distribution, and industrial electrical power systems. It has been designed as an advanced integrated and interactive software package dedicated to electrical power system and control analysis in order to achieve the main objectives of planning and operation optimisation.

“*DIGSILENT*” is an acronym for “**D**igital **S**imu**L**ation of **E**lectrical **N**e**T**works”. *DIGSILENT* Version 7 was the world’s first power system analysis software with an integrated graphical single-line interface. That interactive single-line diagram included drawing functions, editing capabilities and all relevant static and dynamic calculation features.

*PowerFactory* was designed and developed by qualified engineers and programmers with many years of experience in both electrical power system analysis and computer programming. The accuracy and validity of results obtained with *PowerFactory* has been confirmed in a large number of implementations, by organisations involved in the planning and operation of power systems throughout the world.

To address users’ power system analysis requirements, *PowerFactory* was designed as an integrated engineering tool to provide a comprehensive suite of power system analysis functions within a single executable program. Key features include:

1. *PowerFactory* core functions: definition, modification and organisation of cases; core numerical routines; output and documentation functions.
2. Integrated interactive single line graphic and data case handling.
3. Power system element and base case database.
4. Integrated calculation functions (e.g. line and machine parameter calculation based on geometrical or nameplate information).
5. Power system network configuration with interactive or on-line SCADA access.
6. Generic interface for computer-based mapping systems.

Use of a single database, with the required data for all equipment within a power system (e.g. line data, generator data, protection data, harmonic data, controller data), means that *PowerFactory* can easily execute all power simulation functions within a single program environment - functions such as load flow analysis, short-circuit calculation, harmonic analysis, protection coordination, stability analysis, and modal analysis.

Although *PowerFactory* includes highly-sophisticated power system analysis functions, the intuitive user interface makes it possible for new users to very quickly perform common tasks such as load flow and short-circuit calculations.

The functionality purchased by a user is configured in a matrix-like format, where the licensed calculation functions, together with the maximum number of buses, are listed as coordinates. The user can

then, as required, configure the interface and functions according to their requirements.

Depending on user requirements, a specific *PowerFactory* licence may or may not include all of the functions described in this manual. As requirements dictate, additional functionality can be added to a licence. These functions can be used within the same program interface with the same network data. Only additional data, as may be required by an added calculation function, need be added.

## 4.1 General Concept

The general *PowerFactory* program design concept is summarised as follows:

### Functional Integration

*PowerFactory* software is implemented as a single executable program, and is fully compatible with Windows 10 and Windows 11. The programming method employed allows for fast selection of different calculation functions. There is no need to reload modules and update or transfer data and results between different program applications. As an example, the Load Flow, Short-Circuit, and Harmonic Load Flow analysis tools can be executed sequentially without resetting the program, enabling additional software modules and engines, or reading and converting external data files.

### Vertical Integration

*PowerFactory* software uses a vertically integrated model concept that allows models to be shared for all analysis functions. Studies relating to “Generation”, “Transmission”, “Distribution”, and “Industrial” analysis can all be completed within *PowerFactory*. Separate software engines are not required to analyse separate aspects of the power system, or to complete different types of analysis, as *PowerFactory* can accommodate everything within one integrated program and one integrated database.

### Database Integration

**One Database Concept:** *PowerFactory* provides optimal organisation of data and definitions required to perform various calculations, memorisation of settings or software operation options. The *PowerFactory* database environment fully integrates all data required for defining study cases, Operation Scenarios, single line graphics, textual and graphical Results, calculation options, and user-defined models, etc. Everything required to model and simulate the power system is integrated into a single database which can be configured for single and/or multiple users.

**Project Management:** all data that defines a power system model is stored in “Project” folders within the database. Inside a “Project” folder, “Study Cases” are used to define different studies of the system considering the complete network, parts of the network, or Variations on its current state. This “project and study case” approach is used to define and manage power system studies with object-oriented software. *PowerFactory* uses a structure that is easy to use, avoids data redundancy, and simplifies the task of data management and validation for users and organisations. The application of study cases and project Variations in *PowerFactory* facilitates efficient and reliable reproduction of study results.

**Multi-User Operation:** multiple users each holding their own projects or working with data shared from other users are supported by a “Multi-user” database operation. In this case the definition of access rights, user accounting and groups for data sharing are managed by the *PowerFactory* Administrator user.

**Offline Mode:** in some instances, a network connection to a server database may not be available. To address this, *PowerFactory* provides functionality to work in Offline Mode. The required project data is cached to the user’s local machine, which can then later be synchronised to the server database. Offline Mode functionality includes the ability to lock and unlock projects and to edit projects as read-only.

### User Interface Customisation

By default, “Base Package” and “Standard” user profiles are available in *PowerFactory*. Profiles can

be selected from the main menu under *Tools* → *Profiles*. The “Base Package” profile limits the icons displayed on the main toolbar to those typically used by new users, such as load flow and short-circuit commands. The database Administrator can create and customise user profiles, in particular:

- Customise the element dialog pages that are displayed.
- Customise element dialog parameters. Parameters can be Hidden (not shown) or Disabled (shown but not editable).
- Fully configure Main Toolbar and Drawing Toolbar menus, including definition of custom DPL Commands and Templates with user-defined icons.
- Customise Main Menu, Data Manager, and context menu commands.

Chapter [PowerFactory Administration](#), Section [Profiles](#), details the customisation procedure.

### Database, Objects, and Classes

*PowerFactory* uses a hierarchical, object-oriented database. All the data, which represents power system Elements, single line graphics, study cases, system Operation Scenarios, calculation commands, program Settings etc., are stored as objects inside a hierarchical set of folders. The folders are arranged in order to facilitate the definition of the studies and optimise the use of the tools provided by the program.

The objects are grouped according to the kind of element that they represent. These groups are known as “Classes” within the *PowerFactory* environment. For example, an object that represents a synchronous generator in a power system is of a Class called *ElmSym*, and an object storing the settings for a load flow calculation is of a Class called *ComLdf*. Object Classes are analogous to computer file extensions: each Class has a specific set of parameters that defines the objects it represents. As explained in Section [4.7](#) (User Interface), the edit dialogs are the interfaces between the user and an object; the parameters defining the object are accessed through this dialog. This means that there is an edit dialog for each class of object.

The main classes that the *PowerFactory* user is likely to come across fall into these categories:

- **\*.Elm\*** Network elements
- **\*.Typ\*** Type objects, assigned to network element in order to configure parameters often supplied by manufacturers.
- **\*.Int\*** Structural objects, for example \*.IntFolder
- **\*.Set\*** These objects generally contain settings
- **\*.Com\*** Command objects, for example \*.ComLdf for the load flow command

---

**Note:** Everything in *PowerFactory* is an object; an object is defined by its Class and the objects are stored according to a hierarchical arrangement in the database tree.

---

## 4.2 PowerFactory Simulation Functions

*PowerFactory* incorporates a comprehensive list of simulation functions, described in detail in part *Power System Analysis Functions* of the manual, including the following:

- Load Flow Analysis, allowing meshed and mixed 1-,2-, and 3-phase AC and/or DC networks (Chapter [25](#))
- Short-Circuit Analysis, for meshed and mixed 1-,2-, and 3-phase AC networks (Chapter [26](#))

- Sensitivities / Distribution Factors, for voltage, branch flow and transformer sensitivities (Chapter [50](#))
- Contingency Analysis (Chapter [27](#))
- Quasi-Dynamic simulation, which allows the user to perform several load flow calculations in a period of time (Chapter [28](#))
- Network Reduction (Chapter [51](#))
- Protection Analysis (Chapter [33](#))
- Arc-Flash Hazard Analysis (Chapter [34](#))
- Cable Analysis, including cable sizing and cable ampacity calculation (Chapter [35](#))
- Power Quality and Harmonics Analysis (Chapter [36](#))
- Connection Request Assessment (Chapter [37](#))
- Transmission Network Tools (Chapter [41](#)), including:
  - PV curves calculation
  - QV curves calculation
  - Power Transfer Distribution Factors
  - Transfer Capacity Analysis
  - Flow Decomposition
- Distribution Network Tools (Chapter [42](#)), including:
  - Hosting Capacity Analysis
  - Backbone Calculation
  - Voltage Sag
  - Low Voltage Load Flow Calculation
  - Tie Open Point Optimisation
  - Phase Balance Optimisation
  - Voltage Profile Optimisation
  - Optimal Equipment Placement
  - Optimal Capacitor Placement
- Outage Management: tool for management of planned outages (Chapter [43](#))
- Probabilistic Analysis (Chapter [45](#))
- Reliability and Restoration Analysis:
  - Reliability Assessment (Chapter [46](#))
  - Optimal Power Restoration (Chapter [47](#))
  - Optimal Remote Controlled Switch (RCS) Placement (Section [47.3](#))
  - Optimal Manual Restoration (Section [47.4](#))
  - Contingency Restoration Analysis (Chapter [48](#))
  - Generation Adequacy Analysis (Chapter [49](#))
- Optimal Power Flow (Chapter [39](#))
- Unit Commitment and Dispatch Optimisation (Chapter [40](#))
- Economic Analysis Tools (Chapter [44](#)), including:
  - Techno-Economical Calculation (Section [44.2](#))
  - Power Park Energy Analysis (Section [44.4](#))

- State Estimation (Chapter 52)
- RMS Simulation: time-domain simulation for electromechanical transients (Chapter 29)
- EMT Simulation: time-domain simulation of electromagnetic transients (Chapter 29)
- Motor Starting Functions (Chapter 53)
- Modal / Eigenvalue Analysis (Chapter 32)
- System Parameter Identification (Chapter 31)
- Artificial Intelligence (Chapter 54)

### 4.3 General Design of *PowerFactory*

*PowerFactory* is primarily intended to be used and operated in a graphical environment. That is, data is entered by drawing the network elements, and then editing and assigning data to these objects. Data is accessed from the graphics page by double-clicking on an object. An input dialog is displayed and the user may then edit the data for that object.

Figure 4.3.1 shows the *PowerFactory* Graphical User Interface (GUI) when a project is active. The GUI is discussed in further detail in Section 4.7

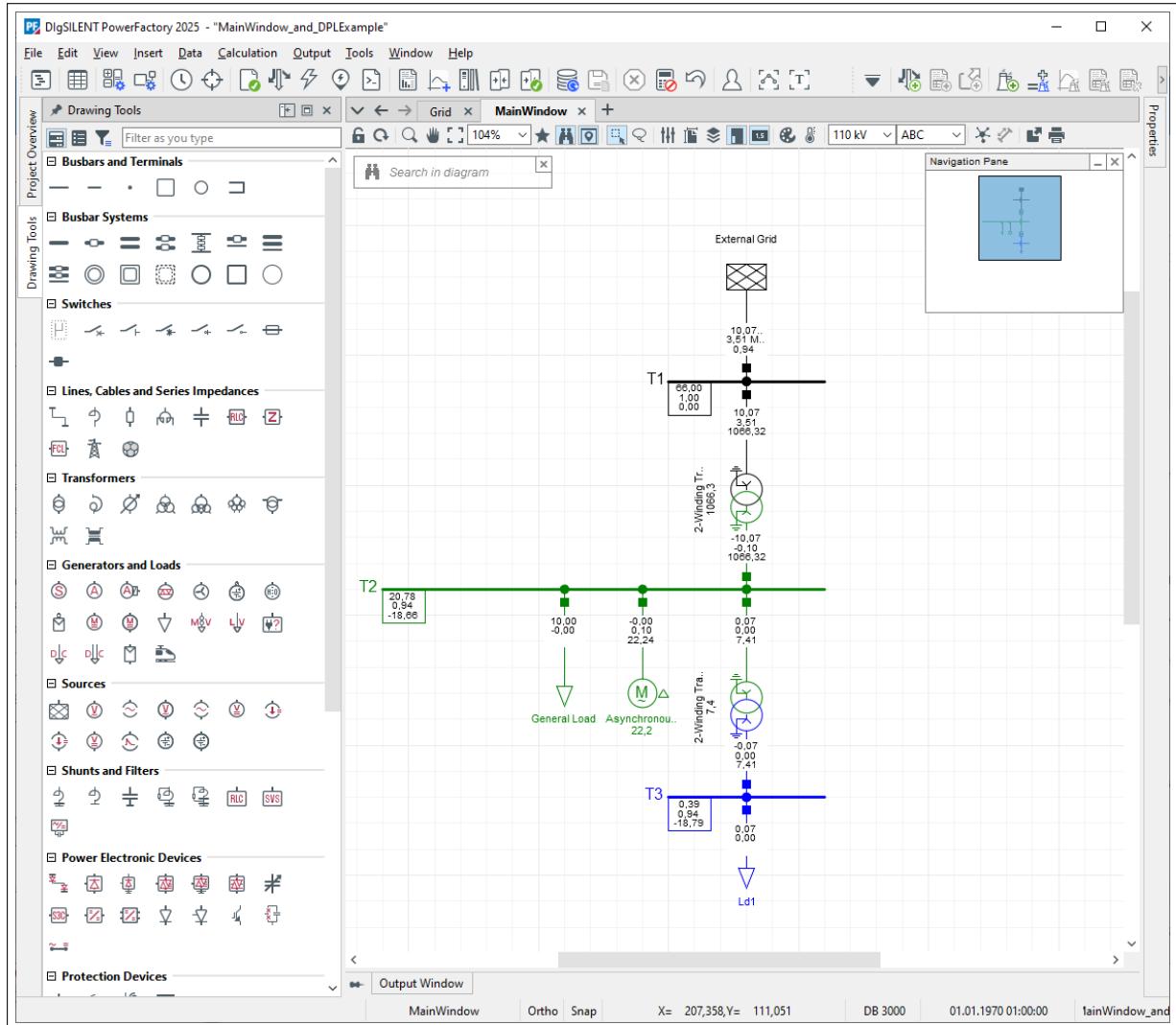


Figure 4.3.1: *PowerFactory* Main Window

All data entered for objects is hierarchically structured in folders for ease of navigation. To view the data and its organisation, a “Data Manager” is used. Figure 4.3.2 shows the Data Manager window. The Data Manager is similar in appearance and functionality to a Windows Explorer window.

Within the Data Manager, information is grouped based on two main criteria:

1. Data that pertains directly to the system under study, that is, electrical data.
2. Study management data, for example, which graphics should be displayed, what options have been chosen for a Load Flow Calculation command, which Areas of the network should be considered for calculation, etc.

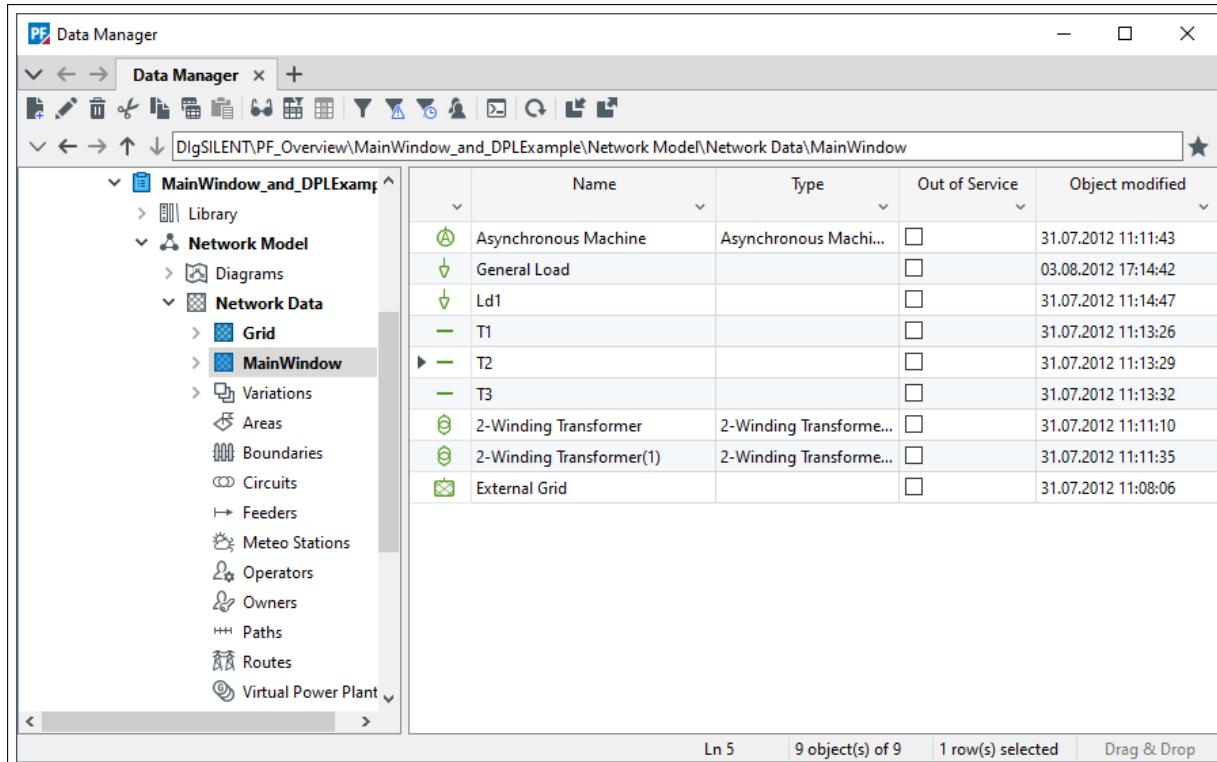


Figure 4.3.2: *PowerFactory* Data Manager

Note that most user-actions can be performed in both the single line graphic and the Data Manager. For example, a new terminal can be added directly to the single line graphic, or alternatively created in the Data Manager. In the latter case, the terminal could be shown in the single line graphic by using the *Diagram Layout Tool*, by “dragging and dropping” from the Data Manager, or by creating a new Graphical Net Object in the Data Manager (advanced).

## 4.4 Type and Element Data

Since power systems are constructed using standardised materials and components, it is convenient to divide electrical data into two sets, namely “Type” data and “Element” data sets.

- Characteristic electrical parameters, such as the reactance per km of a line, or the rated voltage of a transformer are referred to as Type data. Type objects are generally stored in a global library (either the *DIGSILENT* Library or a Custom Library) or Project Library, and are shown in red. For instance, a Line Type object, *TypLne* (🕒).
- Data relating to a particular instance of equipment, such as the length of a line, the derating factor of a cable, the name of a load, the connecting node of a generator, or the tap position of a

transformer are referred to as Element data. Element objects are generally stored in the Network Data folder, and are shown in green. For instance, a Line Element object, *ElmLne* ().

Consider the following example:

- A cable has a Type reactance of “X” Ohms/km, say 0.1 Ohms/km.
- A cable section of length “L” is used for a particular installation, say 600 m, or 0.6 km.
- This section (Element) therefore has an reactance of  $X * L$  Ohms, or 0.06 Ohms.

Note that Element parameters can be modified using Operation Scenarios (which store sets of network operational data), and Parameter Characteristics (which can be used to modify parameters based on the study case Time, or other user-defined trigger).

## 4.5 Data Arrangement

The *PowerFactory* database supports multiple users (as mentioned in 4.1) and each user can manage multiple projects. “**User Account**” folders with access privileges only for their owners (and other users with shared rights) must then be used. User accounts are of course in a higher level than projects.

Figure 4.5.1 shows a snapshot from a database as seen by the user in a Data Manager window, where there is a user account for “User”, and one project titled “Project”. The main folders used to arrange data in *PowerFactory* are summarised below:

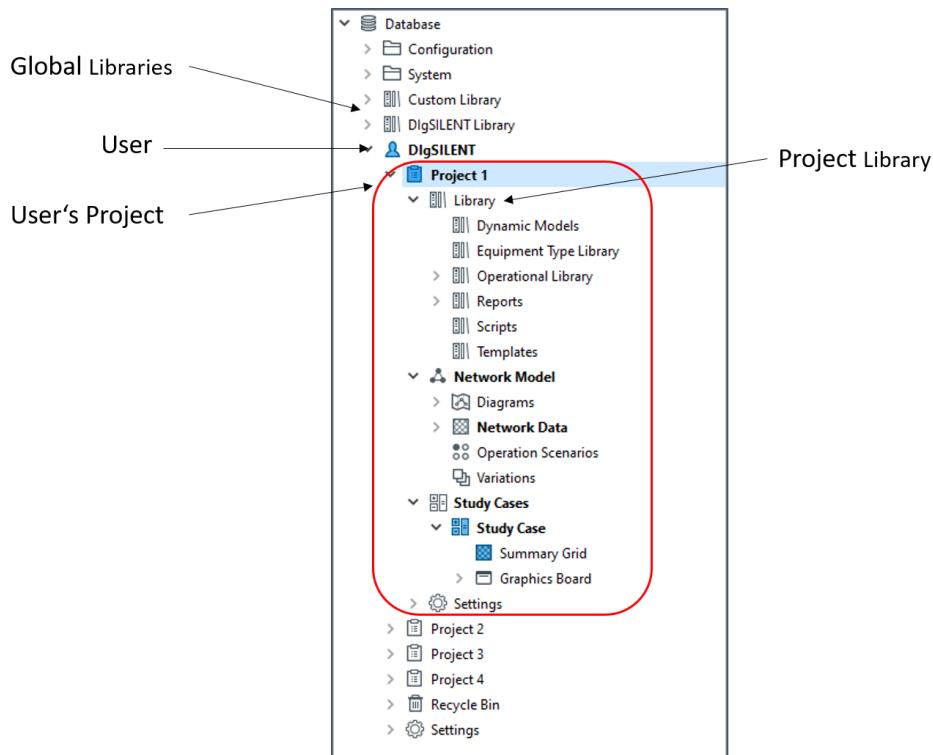


Figure 4.5.1: Structure of a *PowerFactory* project in the Data Manager

### 4.5.1 *DigSILENT* Library

A global library (called “*DigSILENT* Library”) is supplied with *PowerFactory*, it contains a wide range objects and is accessible to all users. The *DigSILENT* Library sub-folders discussed in detail in Chapter 14, Section 14.2 (*DigSILENT* Library).

### 4.5.2 Custom Global Library

When several users work for the same company, there is often a need to share a custom library between these users. Instead of copying the library from user to user, a better approach is to create an additional library in the global area which will be visible for all the users.

More information about how to define a Custom Library and the versioning of it is available in Chapter 14, Section 14.3 (Custom Global Library).

### 4.5.3 Project Library

The Project Library contains the equipment types, network operational information, scripts, templates, and user-defined models (generally) only used within a particular project. A particular project may have references to the project library and / or global library. The Project Library folder and sub-folders are discussed in detail in Chapter 14, Section 14.4 (Project Library).

### 4.5.4 Diagrams

Single line graphics are defined in *PowerFactory* by means of graphic folders of class *IntGrfNet* (). Each diagram corresponds to a *IntGrfNet* folder. They are stored in the *Diagrams* folder () of the Network Model. Single line diagrams are composed of graphical objects, which represent components of the networks under study. Graphical components reference network components and symbol objects (*IntSym*).

The relation between graphical objects and network components allows the definition and modification of the studied networks directly from the single line graphics. Network components can be represented by more than one graphical object (many *IntGrf* objects can refer to the same network component). **Therefore, one component can appear in several diagrams.**

These diagrams are managed by the active study case, and specifically by an object called the *Desktop*. If a reference to a network diagram is stored in a study case's *Desktop*, when the study case is activated, the diagram is automatically opened. Diagrams can be easily added and deleted from the *Desktop*.

Each diagram is related to a specific Grid (*ElmNet*). When a grid is added to an active study case, the user is asked to select (among the diagrams pointing to that grid) the diagrams to display. References to the selected diagrams are then automatically created in the corresponding *Desktop*.

Chapter 10 (Network Graphics), explains how to define and work with single line graphics.

### 4.5.5 Network Data

The Network Data folder holds network data (Element data) in "Grid" folders and object Grouping information.

#### Grids

In *PowerFactory*, electrical network information is stored in "Grid" folders (*ElmNet*, ). A power system may have as many grids as defined by the user. These grids may or may not be interconnected. As long as they are active, they are considered by the calculations. Data may be sorted according to logical, organisational and/or geographical areas (discussed further in Section 4.6: Project Structure).

---

**Note:** A Grid (and in general any object comprising the data model) is active when it is referred to by the current study case. Only objects referred in the current (active) study case are considered for

calculation. In the Data Manager, the icon of an active Grid is shown in blue, to distinguish it from inactive Grids.

---

For details of how to define grids refer to Chapter 9.Basic Project Definition, Section 9.3 (Creating New Grids).

### Grouping Objects

In addition to Grid folders, the Network Data folder contains a set of objects that allow further grouping of network components. By default, when a new project is created, new empty folders to store these grouping objects is created inside the Network Model folder.

For details of how to define grouping objects, refer to Chapter 15: Grouping Objects.

### 4.5.6 Variations

During the planning and assessment of a power system, it is often necessary to analyse different variations and expansion alternatives of the base network. In *PowerFactory* these variations are modelled by means of “Variations”. These are objects that store and implement required changes to a network, and can be easily activated and deactivated. The use of Variations allows the user to conduct studies under different network configurations in an organised and simple way.

Variation objects (*IntScheme*,  ) are stored inside the Variations folder ( ) which resides in the Network Model folder. Variations are composed of “Expansion Stages” (*IntStage*), which store the changes made to the original network(s). The application of these changes depends on the current study time and the activation time of the Expansion Stages.

The study time is a parameter of the active study case, and is used to situate the current study within a time frame. The activation time is a parameter given to the Expansion Stages, to determine whether or not, according to the study time, the changes contained within the Expansion Stages are applied to the network. If the activation time precedes the study time, the changes are applied to the original network. The changes of a subsequent expansion stage add to the changes of its predecessors.

In order that changes to the network configuration are applied and can be viewed, a Variation must be activated. These changes are contained in the expansion stage(s) of this active Variation. Once the Variation is deactivated, the network returns to its original state. Changes contained in an Expansion Stage can be classified as:

- Modifications to network components.
- Components added to the network.
- Components deleted from the network.

---

**Note:** If there is no active Operation Scenario, modifications to operational data will be stored in the active Variation.

---

### 4.5.7 Operation Scenarios

Operation Scenarios may be used to store operational settings, a subset of Element data. Operational data includes data that relates to the operational point of a device but not to the device itself e.g. the tap position of a transformer or the active power dispatch of a generator. Operation Scenarios are stored in the Operation Scenarios folder.

### 4.5.8 Study Cases

The Study Cases folder holds study management information. Study cases are used to store information such as command settings, active Variations and Operations Scenarios, graphics to be displayed, and study results. See Chapter 13 (Study Cases) for details.

### 4.5.9 Settings

Project settings such as user-defined diagram styles for example, which differ from global settings, are stored inside the Settings folder. See Section 9.1.3 (Project Settings)

## 4.6 Project Structure

Most of the data referred to in the previous Section 4.5 is project data. The default high-level arrangement of this data in the project is as shown here:

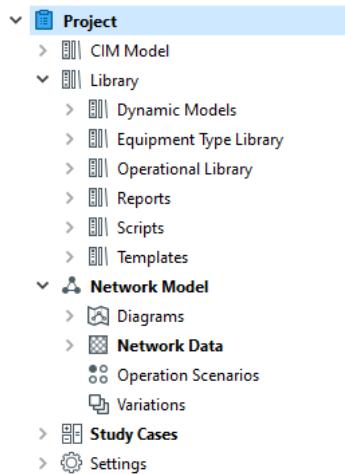


Figure 4.6.1: Basic Project Hierarchy

The structure of project data depends on the complexity of the network, use of the model, and user preferences. The user has the flexibility to define network components directly within the Grid, or to organise and group components in a way that simplifies management of project data.

Consider the example network data arrangement shown in Figure 4.6.2. In this case, two busbar systems (*ElmSubstat* in *PowerFactory*) have been defined, one at 132 kV, and one at 66 kV. The two busbar systems are grouped within a Site, which includes the 132 kV / 66 kV transformers (not shown in Figure 4.6.2). A Branch composed of two line sections and a node connects “132 kV Busbar” to “HV terminal”. Grouping of components in this way simplifies the arrangement of data within the Data Manager, facilitates the drawing overview diagrams, and facilitates storing of Substation switching configurations.

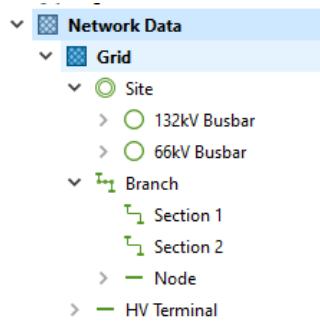


Figure 4.6.2: Example Project Structure

The following subsections provide further information regarding the *PowerFactory* representation of key network topological components.

#### 4.6.1 Nodes

In *PowerFactory*, nodes connecting lines, generators, loads, etc. to the network are generally called “Terminals” (*ElmTerm*). Depending on their usage within the power system, Terminals can be used to represent Busbars, Junctions, or Internal Nodes (their usage is defined by a drop down menu found in the Basic Data page of the terminal dialog). According to the selected usage, different calculation functions are enabled; for example the short-circuit calculation can be performed only for busbars, or for busbars and internal nodes, and so on.

#### 4.6.2 Edge elements

The term “Edge Element” refers to an element connected to a node or to more than one node. Includes single-port elements such as loads, and multi-port elements such as transformers.

#### 4.6.3 Branches

Elements with multiple connections are referred to “branches” (as distinct from a *Branch Element* (*ElmBranch*), which is a grouping of elements, discussed in Section 4.6.9). Branches include two-connection elements such as transmission lines and transformers, and three-connection elements such as three-winding transformers, AC/DC converters with two DC terminals, etc.

For information about how to define transmission lines (and cables) and sections refer to Chapter 12: Building Networks. Technical information about transmission line and cable models is provided in [Technical References Document](#) (Line (*ElmLne*)).

#### 4.6.4 Cubicles

When any edge element is directly connected to a Terminal, *PowerFactory* uses a “Cubicle” (*StaCubic*) to define the connection. Cubicles can be visualised as the panels on a switchgear board, or bays in a high voltage yard, to which the branch elements are connected. A Cubicle is generally created automatically when an element is connected to a node (note that Cubicles are not shown on the single line graphic).

## 4.6.5 Switches

To model complex busbar-substation configurations, switches (*ElmCoup*) can be used. Their usage can be set to Circuit-Breaker, Disconnector, Switch Disconnector, or Load Switch. The connection of an *ElmCoup* to a Terminal is carried out by means of an automatically generated Cubicle without any additional switch (*StaSwitch*) object.

## 4.6.6 Substations

Detailed busbar configurations are represented in *PowerFactory* as Substations (*ElmSubstat*). Separate single line diagrams of individual substations can be created. Substation objects allow the use of running arrangements to store/set station circuit breaker statuses (see Section [14.6.11: Running Arrangements](#)).

For information about how to define substations refer to Chapter [12: Building Networks](#).

## 4.6.7 Secondary Substations

Secondary Substations (*ElmTrfstat*) are smaller, simpler substations, typically used for single-transformer connections.

## 4.6.8 Sites

Network components including Substations and Branches can be grouped together within a “Site” (*ElmSite*). This may include Elements such as substations / busbars at different voltage levels. For information about how to define sites refer to Chapter [12: Building Networks](#).

## 4.6.9 Branch Elements

Similar to Substations, Terminal Elements and Line Elements can be stored within an object called a Branch Element (*ElmBranch*). Branches are “composite” two-port elements that may be connected to a Terminal at each end. They may contain multiple Terminals, Line sections (possibly including various line types), and Loads etc, but be represented as a single Branch on the single line graphic. As for Substations, separate diagrams for the detailed branch can be created with the graphical editor.

For information about how to define branches refer to Chapter [12: Building Networks](#), sections [12.2](#) and [12.5](#).

# 4.7 User Interface

An overview of the *PowerFactory* user interface is provided in this section, including general discussion of the functionality available to enter and manipulate data and graphics.

## 4.7.1 Overview

### 4.7.1.1 General handling

The *PowerFactory* user interface consists of a number of tool windows and tool bars, which can be freely moved around or docked, as the user chooses, and the Desktop, which is the term used to describe the main graphical window, together with other so-called “floating” windows.

#### Tool windows and tool bars

The tool windows are:

- **Project Overview**
- **Drawing Tools**
- **Output Window**

By default, these are “docked” into position, but can be resized or moved around as required. A window is moved by clicking on the top portion of the window and dragging to the required position. This can simply be an “undocked” location, but if the tool window approaches a position where it can be docked, this will be indicated by a blue background as shown here:

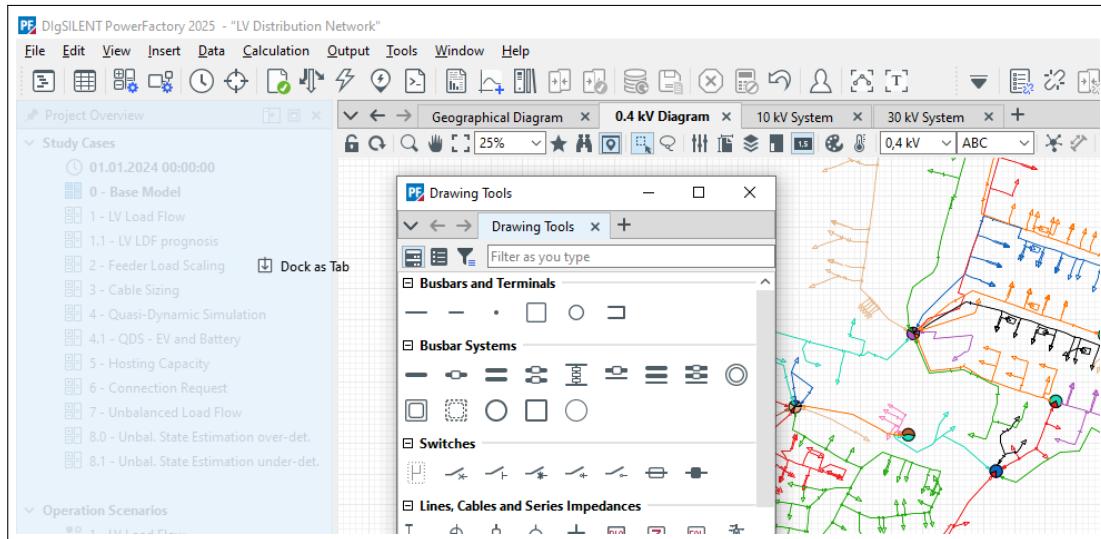


Figure 4.7.1: Docking a tool window

Toolbars can also be moved around. The icons are divided into sets separated by a barrier , and a set of icons can be moved by clicking on the barrier and dragging it to the required location.

A useful option to be aware of is *Reset windows layout*, which is found in the main Window menu. This will restore the various tool windows to their default positions.

Note that this does not affect the graphical window layout, split etc., as this is a study-case specific configuration.

The Desktop and the options available for working with the tabbed windows in *PowerFactory* are described in the next subsections.

### 4.7.2 Desktop (*SetDesktop*)

The term Desktop is used to describe the main graphical window, together with other so-called “floating” windows. A tabbed view concept is used, and in the main graphical window, the currently-selected graphic is often referred to as the active graphic.

If the user wishes to see more than one graphic at the same time, split-screen working can be used. The screen can be split vertically or horizontally or both at the same time, and graphical pages moved between the splits (or new splits created) by right-clicking on the graphic tab and choosing the relevant option. The options available are listed in the Network Graphics chapter, in Section 10.2.5.

The Desktop object, previously called the Graphics Board, contains information about the windows being displayed. It is found in the Study Case.

There are a number of DPL and Python scripting methods for *SetDesktop* objects. Please refer to the relevant scripting reference document for information.

### 4.7.3 Tab groups (*SetTabgroup*)

The term “Tab Group” refers to a window where multiple views are grouped together, each in its own tab. This can be a docked group, being the group of tabs seen in the main graphical window, or a floating group, which typically contains items such as the Data Manager, the Network Model Manager, or reports. Information about the tab groups is held within the Desktop object.

#### Working with tabs

Right-clicking on any tab offers a number of options, dependent on the context. Tabs can be moved from one group to another and new tabs can be created using the + sign. Renaming or duplication of tabs is possible in some cases.

With the exception of the Data Manager, tab groups will close when the Desktop is closed (when the study case or project as a whole is deactivated).

There are a number of DPL and Python scripting methods for *SetTabgroup* objects. Please refer to the relevant scripting reference document for information.

#### Working with tab groups

Any tab group, whether docked or in a floating window, offers a menu of options accessed via the (▼) icon.

The options presented depend upon the whether the user is working with the docked group in the main graphic window, or a floating group. Some options may be greyed out, if they are not currently applicable.

- *Open Recently Closed*: allows the user to open the recently closed diagrams and plots.
- *Open Diagram*: allows the user to open existing diagrams.
- *Open Plot Page*: used for opening the existing plot pages.
- *Open Report Document*: used for opening existing reports.
- *Export All Table Reports*: If the tab group contains more than one tabular report, this allows all reports to be exported at once.
- *Export All Report Documents*: If the tab group contains more than one PDF-format report, this allows all reports to be exported at once.

All open plots and diagrams in the tab group are then individually listed and can be selected as the active tab.

Further options allow all the tabs in a group to be moved or closed:

- For docked groups:
  - *Move All tabs to Left/Right* or *Move All tabs Up/Down*: this option can be used to simultaneously move all the tabs of one tab group to another tab group. (Only shown if the window is already split)
  - *Move All tabs to New Floating Group*: the selection of this option enables the user to simultaneously move all the tabs of a particular group to a floating window.
- For floating groups:
  - *Move All tabs to Main Window*: allows the user to simultaneously move all the tabs in a floating window to the main window.
- For all groups:
  - *Close All tabs*: closes all the tabs of a particular group.

#### 4.7.4 Key Features of the user interface

The main *PowerFactory* window is shown in Figure 4.7.2.

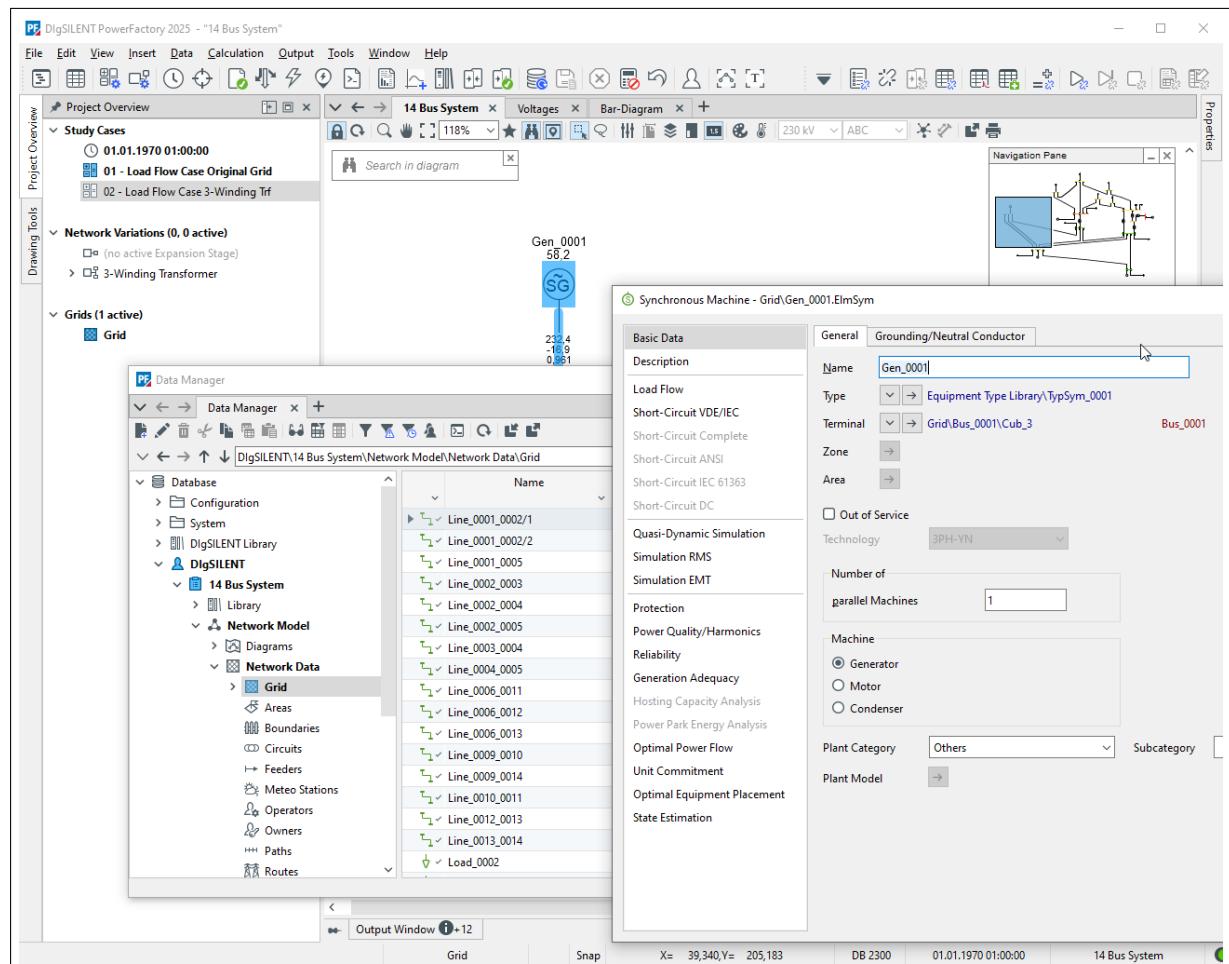


Figure 4.7.2: *PowerFactory* user interface

Key features of the main window are as follows:

1. The main window includes a description of the *PowerFactory* version, and standard icons to Minimise, Maximise/Restore, Resize, and Close the window.

2. The main menu bar includes drop-down menu selections. The main menu is discussed further in Section 4.7.5 (Menu Bar).
  3. The Main Toolbar includes commands and other icons. The Main Toolbar is discussed in further detail in Section 4.7.6 (Main Toolbar).
  4. The Graphical Editor displays single line diagrams, block diagrams and/or simulation plots of the active project. Studied networks and simulation models can be directly modified from the graphical editor by placing and connecting elements.
  5. When an object is right-clicked (in the graphical editor or in the Data Manager) a context menu with several possible actions appears.
  6. When an object is double clicked its edit dialog will be displayed. The edit dialog is the interface between an object and the user. The parameters defining the object are accessed through this edit dialog. Normally an edit dialog is composed of several “pages”. Each page groups parameters that are relevant to a certain function. In Figure 4.7.2 the Load Flow page of a generator is shown, where only generator parameters relevant to load flow calculations are shown.
  7. The Data Manager is the direct interface with the database. It is similar in appearance and functionality to a Windows Explorer window. The left pane displays a symbolic tree representation of the complete database. The right pane is a data browser that shows the content of the currently selected folder. The Data Manager can be accessed by pressing the *Data Manager* icon (☰) on the left of the main toolbar. It is always ‘floating’, and more than one can be active at a time. Depending on how the user navigates to the Database Manager, it may only show the database tree for selecting a database folder, or it may show the full database tree. The primary functionality of the Data Manager is to provide access to power system components/objects. The Data Manager can be used to edit a group of selected objects within the Data Manager in tabular format. Alternatively, objects may be individually edited by double clicking on an object (or *right-click → Edit*).
  8. The output window is shown at the bottom of the *PowerFactory* window. The output window is discussed in further detail in Section 4.7.7 (The Output Window).
  9. The Project Overview window is displayed by default on the left side of the main application window between the main toolbar and the output window. It displays an overview of the project allowing the user to assess the state of the project at a glance and facilitating easy interaction with the project data.
  10. The Drawing Tools window is by default also on the left-hand side of the user interface. In Figure 4.7.2 it is behind the Project Overview, but it can be selected via the tab, and will automatically come to the front if “freeze mode” is removed (see Section 10.2.7 for more information.)
  11. *PowerFactory* has a very flexible window moving and docking concept that aims to optimise work when using more than one monitor, allowing the user to:
    - create as many groups of tabs rightward or downward in the Main Window as required and freely move the tabs between these groups
    - create a new floating window by moving a tab from the Main Window or from an already existing floating window
    - move tabs from a floating window to the Main Window (docking) or to another already existing floating window
- Moving and docking options apply to both tabs displaying single-line diagrams and those containing plots. Even tabular reports can be moved or docked freely as described above. The different moving and docking options can be accessed by right-clicking on the tab. Another method, perhaps more intuitive, is to simply drag the tab to the centre of the screen to create a new floating window or to an existing window to add the tab to it.
12. *PowerFactory* dynamically adapts to any screen resolution and DPI scaling setting. The application perfectly respects DPI scaling when moving windows between two screens of different resolution settings, e.g. when working with a laptop and an additional monitor, or two monitors of different resolution settings.

#### 4.7.5 Menu Bar

The menu bar contains the main *PowerFactory* menus. Each menu entry has a drop down list of menu options and each menu option performs a specific action. To open a drop down list, either click on the menu entry with the left mouse button, or press the **Alt** key together with the underlined letter in the menu. Menu options that are shown in grey are not available, and only become available as the user activates projects or calculation modes, as required.

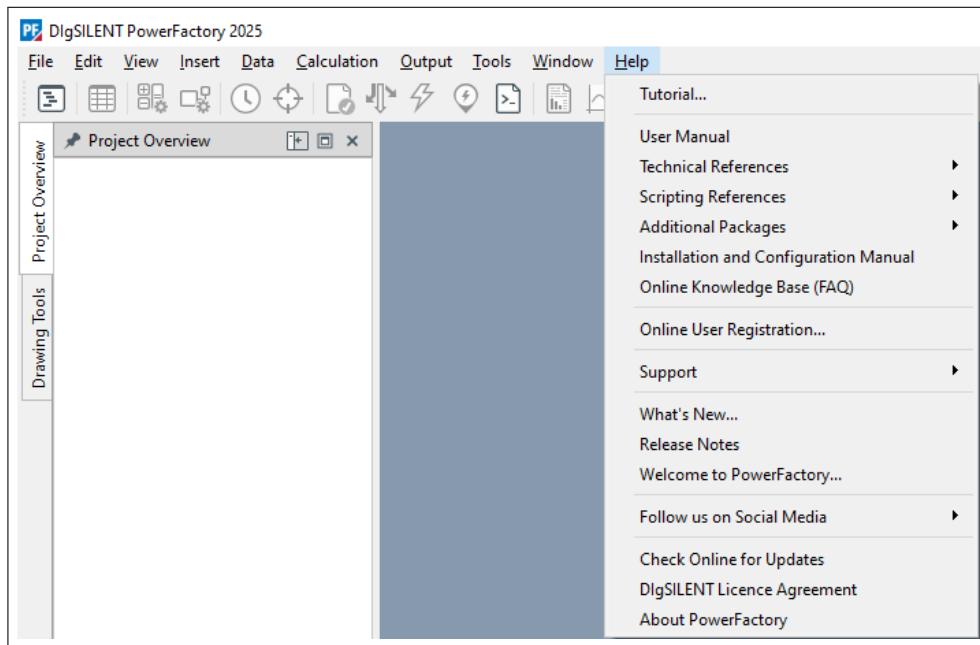


Figure 4.7.3: The help menu on the Menu bar

For example as show in Figure 4.7.3:

- To access *PowerFactory* tutorial: Press **Alt-H** to open the help menu. Use the keyboard to select Tutorial....
- To access the User Manual: Left click the *Help* menu. Left-click the option *User Manual* to open the electronic User Manual.

#### 4.7.6 Main Toolbar

The main *PowerFactory* toolbar provides the user with quick access to the main commands available in the program (see Figure 4.7.2). Buttons that appear in grey are only active when appropriate. All command icons are equipped with balloon help text which are displayed when the cursor is held still over the icon for a moment, and no key is pressed.

Note that the visibility of buttons to an individual user may depend on the User Profile. See Section 6.7.2 for more details about this.

To use a command icon, click on it with the left mouse button. Those icons that perform a task will automatically return to a non-depressed state when that task is finished. Some command icons will remain depressed, such as the button to *Maximise Output Window*. When pressed again, the button will return to the original (non-depressed) state.

This section provides a brief explanation of the purpose of the icons found on the upper part of the toolbar. Icons from the lower part of the toolbar are discussed in Chapter 10 (Network Graphics (Single

Line Diagrams)). Detailed explanations for each of the functions that the icons command are provided in the other sections of the manual.

### Open Data Manager

Opens a new instance of the Data Manager. If the option “Always open new Data Manager” is not enabled (default), using the  icon or Ctrl+D to open the Data Manager will present the existing Data Manager if one is already open; if enabled, a separate new Data Manager tab will be created.

For more information on the Data Manager refer to Chapter 11 ([Data Manager and Network Model Manager](#)).

### Open Network Model Manager

Opens the Network Model Manager, which is a browser for all calculation relevant objects. It provides a list of all elements (coloured in green) and types (coloured in red) that are in an active Grid: e.g. transformer types, line elements, composite models, etc.

If the option “Always open new Network Model Manager” is not enabled (default), using the  icon to open the Network Model Manager will present the existing Network Model Manager if one is already open; if enabled, a separate new Network Model Manager tab will be created.

For more information, see Chapter 11 ([Data Manager and Network Model Manager](#)).

### Study Case Manager

Deactivates the currently-active study case and opens the Study Case Manager window. See Section 13.2.1 for more details.

### Date/Time of Study Case

Displays the date and time for the case calculation. This option is used when parameter characteristics of specific elements (e.g. active and reactive power of loads) are set to change according to the study time, or a Variation status is set to change with the study time.

### Edit Trigger

Displays a list of all Triggers that are in the active study case. These Triggers can be edited in order to change the values for which one or more characteristics are defined. These values will be modified with reference to the new Trigger value. All Triggers for all relevant characteristics are automatically listed. If required, new Triggers will be created in the study case. For more information, see Chapter 18: Parameter Characteristics, Load States, and Tariffs. Section 18.2.

### Network Data Assessment

Activates the Network Data Assessment command dialog to generate selected reports on network data or to perform model data verification. For more information see Section 26.7: Capacitive Earth-Fault Current or Section 25.6: Troubleshooting Load Flow Calculation Problems.

### Calculate Load Flow

Activates the Load Flow Calculation command dialog. For more information about the specific settings, refer to Chapter 25: Load Flow Analysis.

### Calculate Short-Circuit

Activates the short-circuit calculation command dialog. For more information, refer to Chapter 26: Short-Circuit Analysis.

### Edit Short-Circuits

Edits Short-Circuit events. Events are used when a calculation requires more than one action or considers more than one object for the calculation. Multiple fault analysis is an example of this. If, for instance, the user multi-selects two busbars (using the cursor) and then clicks the right mouse button *Calculation* → *Multiple Faults...* a Short-circuit event list will be created with these two busbars in it.

#### **Execute Script**

Displays a list of scripts that are available. See Section 4.8.1 for a general description of DPL scripts, and Chapter 23: Scripting for detailed information.

#### **Generate Reports**

Used to generate inbuilt or customised reports, which can then be viewed using *PowerFactory's Report Viewer* or exported in a number of different formats. Depending on the calculation, one or more reports can be created. For more information, refer to Chapter 19: Reporting and Visualising Results, Section 19.5.

#### **Insert Plot**

Opens the Insert Plot dialog, where different types of plot can be selected. For more information refer to Chapter 19: Reporting and Visualising Results, Section 19.8.

#### **Documentation of Device Data**

Presents a listing of device data (a device is the model of any physical object that has been entered into the project for study). This output may be used in reports, and for checking data that has been entered. Depending on the element chosen for the report, the user has two options; generate a short listing, or a detailed report. For more information refer to Chapter 19: Reporting and Visualising Results, Section 19.4.

#### **Comparing of Results On/Off**

Turns on/off comparing of calculation results. Used to compare results where certain settings or designs options of a power system have been changed from one calculation to the next. For more information refer to Chapter 19: Reporting and Visualising Results, Section 19.6.

#### **Edit Comparing of Results**

Enables the user to select the cases/ calculation results that are to be compared to one another, or to set the colouring mode for the difference reporting. For more information refer to Chapter 19: Reporting and Visualising Results, Section 19.6.

#### **Update Database**

Utilises the current calculations results (i.e. the calculation 'output' data) to change input parameters (i.e. data the user has entered). An example is the transformer tap positions, where these have been calculated by the Load Flow command option "Automatic Tap Adjust of Tap Changers." For more information refer to Chapter 25: Load Flow Analysis, Section 25.5.7.

#### **Save Operation Scenario**

Saves the current operational data to an Operation Scenario (e.g. load values, switch statuses, etc.). See Chapter 16: Operation Scenarios.

#### **Break**

Stops a transient simulation or DPL script that is running.

#### **Reset Calculation**

Resets any calculation performed previously. This icon is only enabled after a calculation has been carried out.

**Note:** If *Retention of results after network change* is set to *Show last results* (User Settings, *Miscellaneous* page), results will appear in grey on the single line diagram and on the Flexible Data tab until the calculation is reset, or a new calculation performed.

## ↶ Undo

This is used to undo the last action which caused information to be written to the database.

## 👤 User Settings

User options for many global features of *PowerFactory* may be set from the dialog accessed by this icon. For more information refer to Chapter 7: User Settings.

## ☒ Maximise Graphic Window

Maximises the graphic window. Pressing this icon again will return the graphic window to its original state.

## [T] Maximise Output Window

Maximises the output window. Pressing this icon again will return the output window to its original state.

## ▼ Change Toolbox

In order to minimise the number of icons displayed on the taskbar, some icons are grouped based on the type of analysis, and are only displayed when the relevant category is selected from the *Change Toolbox* icon. In Figure 4.7.4, the user has selected *Simulation RMS/EMT*, and therefore only icons relevant for RMS and EMT studies are displayed to the right of the *Change Toolbox* icon. If, for example, *Reliability Analysis* were selected then icons to the right of the *Change Toolbox* icon would change to those suitable for a reliability analysis.

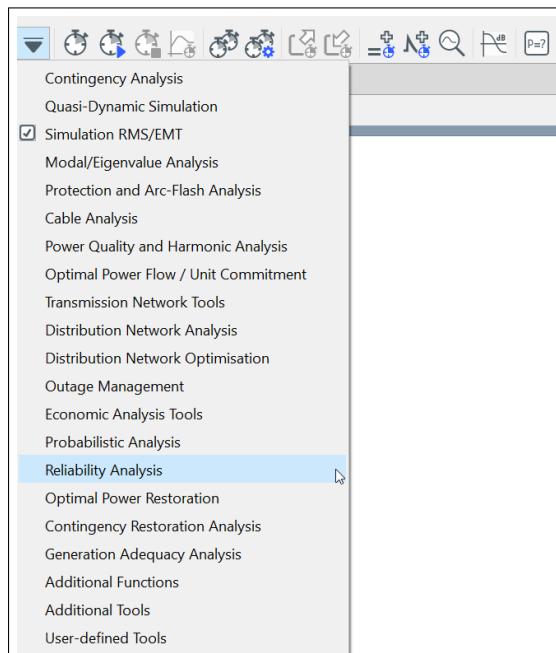


Figure 4.7.4: Change Toolbox selection

## 4.7.7 The Output Window

In addition to results presented in the single line graphics and / or Data Manager, the output window displays other textual output, such as error messages, warnings, command messages, device documentation, results of calculations, and generated reports, etc. This section describes output window use and functionality.

### 4.7.7.1 Output Window Visibility

By default, the output window is at the bottom of the main window, but it is not “pinned”, meaning that it is collapsed so as to leave more space for other windows.

There are several ways in which the visibility of the output window can be managed:

- The pin icon on the title bar can be used to pin/unpin the window; but even when unpinned, the output window can be viewed at any time by clicking on its tab.
- Alternatively, it can be left pinned but collapsed (using an icon toward the right-hand side of the title bar)
- It can be moved into another tab group or a separate floating group by dragging the tab to the new location.
- The output window visibility can be changed via the Window menu on the main toolbar.

If the output window is not pinned, an indication of the unread messages can be seen in its tab. By default, error messages will cause the output window to pop up, so that the errors are not overlooked. This default behaviour can be changed by the user, via the User Settings (Output Window page).

Two additional options for the output window are:

- Pressing the *Maximise Graphic Window* icon ( ) on the main toolbar to enlarge the graphical window by hiding the output window.
- Pressing the *Maximise Output Window* icon ( ) icons on the main toolbar to enlarge the output window.

### 4.7.7.2 Output Window Options

The contents of the output window may be stored, edited, printed, etc., using the icons shown on the right-hand pane of the output window. Some commands are also available from the context menu by right-clicking the mouse in the output window pane.

-  Opens the User Settings dialog on the *Output Window* page.
-  Turns the auto scrolling function on and off. When on, the the last line of the output window is displayed after each action, otherwise the output window remains at the last position viewed.
-  Saves the selected text, or the complete contents of the output window if no selection was made, to a text, html or csv file.
-  Copies the selected text to the Windows Clipboard. Text may then be pasted in other programs.
-  The contents of the output window are displayed and immediately saved in a file.
-  Redirects the output window to be printed directly.
-  Searches the text in the output window for the occurrences of a given text.

-  Clears the output window by deleting all messages. Note that when the user scrolls back and clicks on previous messages in the output window, the output window will no longer automatically scroll with new output messages. The *Clear All* icon will “reset” scrolling of the output window. **Ctrl\End** can also be used to “reset” scrolling.

#### 4.7.7.3 Using the Output Window

The output window facilitates preparation of data for calculations, and identification of network data errors. Objects which appear blue in the output window generally have a hyperlink so that they can be clicked with the left mouse button to open an edit dialog for the object. Alternatively, the object can be right-clicked and then *Edit*, *Show in Data Manager*, or *Mark in Graphic* selected. This simplifies the task of locating objects in the single line diagrams.

Additionally, options to jump between message types are available when selecting the option *Go to → Next/previous message*

#### 4.7.7.4 Output Window Filters

In the output window, shown in Figure 4.7.5, the messages are not only coloured, but icons are also used to indicate the category (error, warning, info, events,...); these categories can be filtered using the predefined filtering tabs. There is also a text filter, to find specific text strings in the output messages.

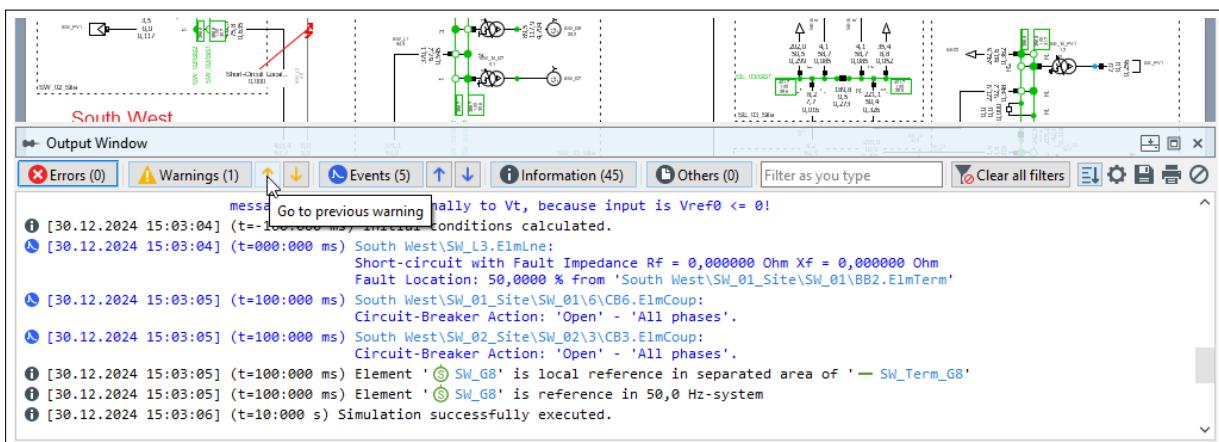


Figure 4.7.5: Output window

The messages in the output window are classified as following:

-  Error message: red.
-  Warning message: dark yellow.
-  Information messages: black.
-  Events messages: blue.
-  Output text: black.

The button  *Clear all filters* can be used to remove all the selected filters.

These settings can also be changed in the User Settings dialog (*User Settings → Data/Network Model Manager*) as described in Chapter 7.

#### 4.7.7.5 Copying from the Output Window

The contents of the output window, or parts of its contents, may be copied to the built-in editor of *PowerFactory*, or to other programs. The lines that are to be copied are determined by the output window settings; by default what is shown in the output window is copied. The filters can be used to show only the messages of interest.

### 4.7.8 Use of Colouring in *PowerFactory*

Colouring is used widely in *PowerFactory* for the presentation of data and visualisation of results. Generally speaking, the user can either configure the colour scheme, or simply let *PowerFactory* apply default settings. In this section, the general approach and options are described.

It should be noted that in some contexts, in particular the colouring of curves in plots, colour palettes are offered. In other contexts, the user simply configures the chosen colour directly.

#### 4.7.8.1 Colour palettes

A number of colour palettes are available. The use of palettes is particularly useful when configuring colours for plots, because by selecting a palette, the user automatically assigns a harmonious set of colours to the curves on the plot, according to the desired appearance.

The palettes available include *PowerFactory Standard*, which is the default palette, and *PowerFactory Classic*, which consists of the colours that were formerly used as a default colours in *PowerFactory*.

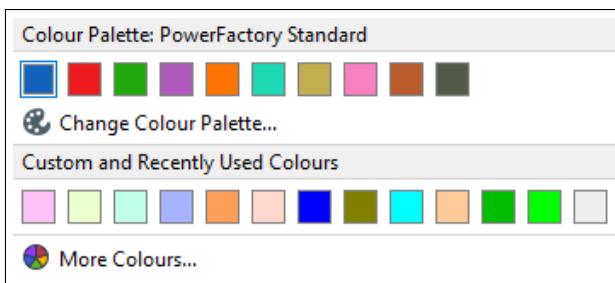


Figure 4.7.6: Choosing colours from the *PowerFactory Standard* palette

In addition, users can create, store and use their own colour palettes within their projects, using individual colour definition.

#### 4.7.8.2 Individual colour definition

The configuration of individual colours is done via the *Select Colour* dialog, shown in Fig 4.7.7 (this dialog is also reached when clicking on *More colours ...* in the dialog shown in Figure 4.7.6). Colours can be defined using the standard HSV, RGBA or HTML formats, or simply by clicking on the colour panel. Another option is to use the *Pick Screen Colour* button to reproduce any colour visible on the user's screen.

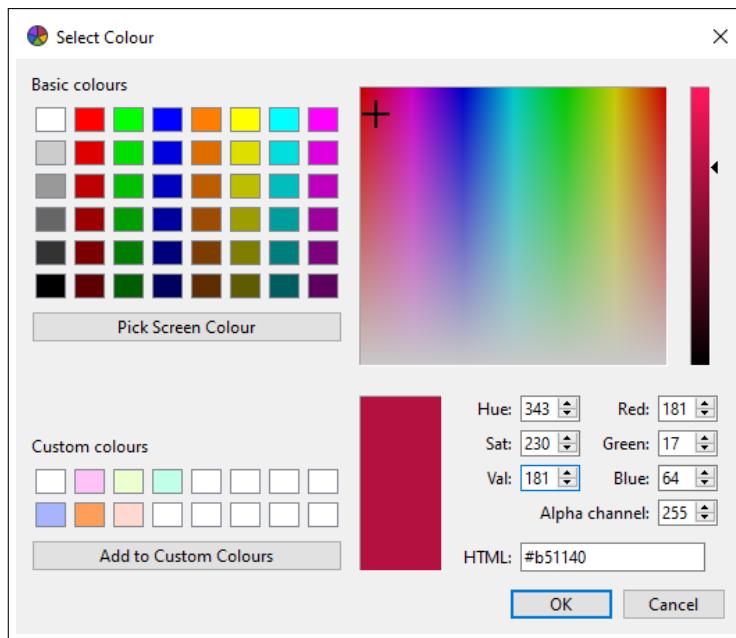


Figure 4.7.7: The *Select Colour* dialog

This dialog also allows the user to add a colour which they have defined to the list of custom colours, which will then be available for future use.

#### 4.7.8.3 Colouring of plots

For many of the functions in *PowerFactory*, the results can be presented in plots. The various plot options are described in Section 19.8, but the general principle is that the user can select a palette and then the colours will be automatically assigned from that palette. Nevertheless, the user still has the option to configure the colouring of individual curves.

#### 4.7.8.4 Colouring of elements in diagrams

There are wide-ranging options available for the colouring of single line diagrams, substation diagrams and geographic diagrams, which are all described in Section 10.3.10.1. These colouring options are based around network elements, and the user has the freedom to configure the colours used, via the *Colour Settings* buttons found in the *Diagram Colouring Scheme* dialog, which open the colour settings dialog.

On the General page of the colour settings dialog, the user will find an option to select a colour palette for diagrams. Selecting this palette will not automatically change any colour settings, but it will mean that the colours from the palette are offered as a default range when colours are subsequently defined for elements.

#### 4.7.8.5 Colouring in the Network Model Manager and Data Manager

Some grouping objects such as Zones or Boundaries have colours associated with them. These colours are individually configurable, with the same option as above to assign a default palette for colour selection.

Another use of colour in the Network Model Manager and Data Manager is the colouring of fields in the flexible data page. These colours can be changed via the user settings (see Section 7.8), using the

Select Colour dialog.

## 4.8 Scripting in *PowerFactory*

For automating tasks in *PowerFactory*, two scripting options are available: use of the inbuilt *DlgSILENT Programming Language DPL*, or scripting using Python.

### 4.8.1 *DlgSILENT Programming Language (DPL) Scripts*

**DPL** offers an interface to the user for the automation of tasks in *PowerFactory*. By means of a simple programming language and in-built editor, the user can define automation commands (scripts) to perform iterative or repetitive calculations on target networks, and post-process the results.

To find the name of an object parameter to be used in a DPL script, simply hover the mouse pointer over the relevant field in an object dialog. For example, for a general load, on the *Load Flow* page, hover the mouse pointer over the *Active Power* field to show the parameter name *plini*.

User-defined DPL scripts can be used in all areas of power system analysis, for example:

- Network optimisation
- Cable-sizing
- Protection coordination
- Stability analysis
- Parametric sweep analysis
- Contingency analysis

DPL scripts may include the following:

- Program flow commands such as 'if-else' and 'do-while'
- *PowerFactory* commands (i.e. load-flow or short-circuit commands: *ComLdf*, *ComShc*)
- Input and output routines
- Mathematical expressions
- *PowerFactory* object procedure calls
- Subroutine calls

DPL command objects provide an interface for the configuration, preparation, and use of DPL scripts. These objects may take input parameters, variables and/or objects, pass these to functions or subroutines, and then output results. By default, DPL commands are stored inside the Scripts folder of the project.

Consider the following simple example shown in Figure 4.8.1 to illustrate the DPL interface, and the versatility of DPL scripts to take a user-selection from the single line graphic. The example DPL script takes a load selection from the single line graphic, and implements a while loop to output the Load name(s) to the output window. Note that there is also a check to see if any loads have been selected by the user.

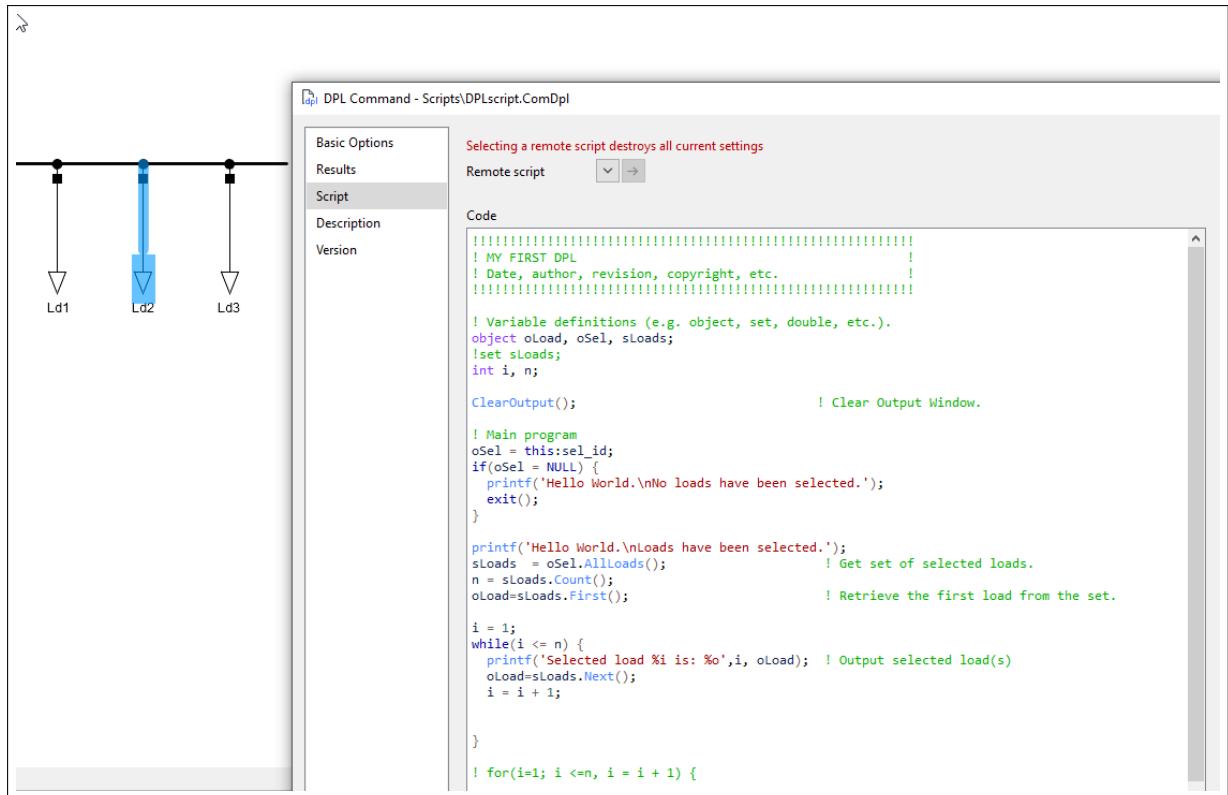


Figure 4.8.1: Example DPL Script

For further information about DPL commands and how to write and execute DPL scripts refer to Chapter 23 (Scripting), and the [DPL Reference](#).

## 4.8.2 Python Scripts

In addition to DPL it is also possible to use the Python language to write scripts to be executed in *PowerFactory*.

This can be done in one of two ways:

- The script can be written directly in a Python object (*ComPython*) in *PowerFactory*, or
- For more complex scripts, a Python script can be written in an external editor and linked to the Python command (*ComPython*) inside *PowerFactory*.

For further information about the Python command and how to write and execute Python scripts refer to Chapter 23 (Scripting), and the [Python Reference](#).

## **Part II**

# **Administration**

# Chapter 5

# ***PowerFactory Installation and Configuration***

This chapter provides information on how to configure *PowerFactory*, and how to log on. More detailed descriptions of the installation, database settings and additional information can be found in the [Advanced Installation and Configuration Manual](#).

---

**Note:** Windows administrator rights are necessary to install and configure *PowerFactory*.

---

## **5.1 Program Installation**

In general there are 3 primary questions to consider before installing *PowerFactory* software, which will determine the installation settings:

- Licence: Where should the licence key(s) reside?
- Installation: Where should *PowerFactory* be installed?
- Database: Where should the database reside?

Once *PowerFactory* has been installed, it can be started by clicking either on the desktop or by selecting *PowerFactory* in the Windows Start menu. *PowerFactory* will then start and create a user account upon the initial user log-in. If the user is working in a single-user-database environment, *PowerFactory* will take the username from Windows by default.

The user will then be automatically logged on. More information about users and login options is available in chapter [PowerFactory Administration](#).

## **5.2 PowerFactory Configuration**

The configuration of the application is stored in an “ini” file located with the executable. Changes to the application settings can be made using the *PowerFactory* Configuration dialog. Once *PowerFactory* is started, the Configuration dialog can be accessed via *Tools* → *Configuration* in *PowerFactory*’s main menu. Depending on where the “ini” file is stored, Windows administrator rights might be required to change these settings.

### 5.2.1 General Page

On this page the user can select the application language for the session.

### 5.2.2 Database Page

This page allows the user to specify what kind of database will be used. The options are:

- A single-user database which resides locally on each computer
- A multi-user database (used in conjunction with the appropriate licence) which resides on a remote server. Here all users have access to the same data simultaneously. In this case, user accounts are created and administrated exclusively by the Administrator.

*PowerFactory* provides drivers for the following multi-user database systems:

- Oracle
- Microsoft SQL Server
- PostgreSQL

For further information regarding the database configuration refer to the [Advanced Installation and Configuration Manual](#).

### 5.2.3 Workspace Page

The *Workspace* page allows the user to set the workspace directory and the workspace backup directory. The workspace is used to store the local database, results files and log files. For further information regarding options for configuring and using the workspace, refer to Section [5.4](#).

### 5.2.4 External Applications Page

The *External Applications* page is used to configure the external programs.

#### Python

- **Interpreter:** the Python Interpreter to be used can either be selected by version or by directory.
- **Version:** a number of Python versions are supported.
- **Used Editor:** there are three options to set the Python editor:
  - **internal:** uses the internal editor provided by *PowerFactory*. This editor is the same used when writing DPL scripts. More information about this editor can be found in Section [23.1.3](#).
  - **system default:** uses the system's default editor for Python files (\*.py); if no editor is defined as default for Python files, then the default editor for text files (\*.txt) is used. This is the default option.
  - **custom:** here the user can customise which editor should be used to open Python files.

#### C\C++ Compiler

A *C\C++ Compiler* can be configured here in order to enable the compilation functions of *PowerFactory* native dynamic DSL and Modelica models (see Section [30.1.5](#) for more information on compiling user defined dynamic models)

The following *Version* options are available:

- MinGW
- Visual Studio 20xx

The *MinGW* compiler is installed by default with *PowerFactory* and hence no other third party installation packages are required in order to make use of a C/C++ compiler. Detailed licence information for MinGW itself, the MinGW runtime and additional libraries can be found in the installation directory of *PowerFactory*, subfolder *MinGW → licences*.

If *Visual Studio* is installed, then it can be used for dynamic model compilation purposes by setting in *PowerFactory* the specific *Visual Studio* version (selected in the *Version* field).

---

**Note:** Make sure that the selected *Visual Studio* compiler version is already installed on the host computer in order for *PowerFactory* to correctly operate.

---

If a *Visual Studio* version is selected, then the *Shell Extension* of Visual Studio must be set.

#### PDF Viewer

Here the User can select which program should be used to open “.pdf” files. There are three options to set the PDF viewer:

- **system viewer:** uses the system’s default editor for pdf files (\*.pdf). This is the default option.
- **Sumatra PDF:** uses “Sumatra PDF” which is included in the *PowerFactory* installation.
- **custom:** here the user can customise which viewer should be used to open pdf files (\*.pdf).

#### PyTorch

In the *Directory* field should be specified the path to the folder where the copies of the corresponding \*.dll files are stored. For further information regarding PyTorch is available in the [Advanced Installation and Configuration Manual](#).

### 5.2.5 Network Page

The *Network* page is used to specify an HTTP proxy in the case where the user’s computer connects to the internet via a proxy server.

#### Proxy Host

Three options are available for specifying the proxy configuration, including options to change the proxy settings externally to *PowerFactory* if required.

- Use the system proxy settings.
- Configure the proxy manually, supplying the host name and port number.
- Provide a path to a proxy auto-configuration file (PAC).

#### Proxy Authentication

If it is necessary to provide authentication details to the proxy, this option is also checked, and the relevant protocol selected from the drop-down list. Unless the username and password are to be taken from the Windows user authentication details, they are entered here.

There is also a “Check internet Connection” button to check whether the configuration has been set up successfully.

## 5.2.6 Geographic Maps Page

On the *Geographic Maps* page, the default settings for background maps can be changed. The following parameters can be set:

- **Map Tile Cache**
  - **Directory:** map cache directory where downloaded map tiles are stored (default: workspace directory). A custom directory can be specified if the cache should be shared across different *PowerFactory* installations.
  - **RAM cache limit:** this setting can be used to limit the amount of application memory that should be used.
- **Network Settings**
  - **Preferred tile size [pixels]:** pixel dimensions of map tiles.
  - **Max server connections:** maximum number of map tiles that are downloaded simultaneously.
  - **Download time-out:** timespan after which a non-finished tile download is cancelled. This value may need to be increased for slow/unstable internet connections.
- **Google Maps account**

If Google Maps is to be used as the map provider, the “Google Maps account” data must be set on this page as well. To acquire a licence, please contact Google sales: (<http://www.google.com/enterprise/mapsearch>).

Similarly, the licence keys for other map providers can be entered.

## 5.2.7 Advanced Page

### General tab

- Paths in the **Additional Directories in PATH** field are used to extend the Windows path search. Typically this is required for the Oracle client.
- **Directories for external digex\* libraries (DLL):** In each row, set one directory path where legacy “digexdyn” or “digexfun” compiled DLL dynamic models may be located. First row is set by default to “\$(InstallationDir)”, representing the installation directory of *PowerFactory* (for more information on *PowerFactory* installation, refer to the [Advanced Installation and Configuration Manual](#)). The use of dynamic models based on legacy “digexdyn” and “digexfun” interfaces is recommended to be avoided, whenever possible. Refer to Section [30.14](#) for information on recommended practice of interfacing external dynamic models with *PowerFactory*.

---

**Note:** *PowerFactory* is supplied based on a 64 bit architecture, hence 64 bit compiled DLL files of all interfaced external models are required. Therefore, make sure that all DLL files within the assigned path(s) for external digex\* libraries do support 64 bit architecture.

---

### Advanced tab

Settings on the *Advanced* tab should only be changed under the guidance of the *PowerFactory* support (see Chapter [2 Contact](#)).

## 5.3 Licence

### 5.3.1 Select Licence

In order to run *PowerFactory*, the user is required to define licence settings in the DLgSILENT *PowerFactory Licence Manager*, its dialog can be accessed via *Tools* → *Licence* → *Select Licence*...

---

**Note:** The *PowerFactory Licence Manager* can be started externally using the corresponding shortcut in the main installation folder of *PowerFactory* or in the *Windows* start menu.

---

The *Licence Access* defines the type of licence, which can be a local licence (either a licence file or a USB dongle) or a network licence.

#### Automatic search

This option searches automatically local and network licences via a broadcast and chooses the first one found without further input.

#### Local Softkey / USB dongle

If local softkey / USB dongle is chosen, the *Local Licence Settings* require the selection of a *Licence Container*. The locally found containers are available in the drop-down-list.

#### Network licence

If network licence is chosen, the server name has to be selected from the drop-down-list or entered manually in the *Network Licence Settings*. Pressing  will refresh the list of available licence servers in the network. For the specified server the *Licence container* can be chosen from a drop-down-list or entered manually.

#### Selected Licence

The field on the right side of the dialog shows various details relating to the selected licence. This includes the order ID (useful for any contact with the sales department), the customer ID (useful for contact with technical support), the maximum number of concurrent users for a multi user environment and a list of the licensed additional modules. Note that the expiry date of the maintenance period for the licence is also shown.

If problems with the licence occur, the button *Create Licence Support Package* creates a zipped file with the needed information for the support to identify the cause of the problems.

### 5.3.2 Activate / Update / Deactivate / Move Licence

These options are relevant for local licences, where the user has to manage the licence. In a network licence environment, this is done by the network administrator.

For the activation, the update and the deactivation process the licence related *Activation Key* has to be entered into the upcoming dialog.

A *PowerFactory* software licence softkey can be moved between computers a limited number of times per year. The licence move is a two-stage process:

1. An activated licence needs to be transferred back to the *DIGSILENT* server via the Deactivate Licence feature of the Licence Manager.
2. The deactivated licence can be activated again on any computer.

More information regarding licence types and their management is available in the [Advanced Installation and Configuration Manual](#).

### 5.3.3 Licence and Maintenance Status

Via *Help* → *About PowerFactory*, users can get a summary of their *PowerFactory* installation, including version number, available calculation functions, and licence and maintenance information.

Left-clicking on the traffic light icon located in the lower right corner of the main window status bar gives quick access to this Help/About dialog, in which detailed information about licence and maintenance status can be found, as shown in Figure 5.3.1.



Figure 5.3.1: Licence and Maintenance Status

Two states are monitored and categorised according to a traffic light colour system.

#### State of time-limited licences

- Green: licence can be used for at least 6 days
- Yellow: licence expires in less than 6 days
- If the licence has expired, *PowerFactory* will not start

#### State of maintenance contract

- Green: maintenance contract is valid for at least 46 days
- Yellow: maintenance contract expires in less than 46 days
- Red: maintenance contract has expired

---

**Note:** The traffic light icon shows the more severe of the two states with the time-limited licence state having priority if both states have equal severity.

---

## 5.4 Workspace Options

By selecting *Tools* → *Workspace* from the main menu, the options described below are available.

### 5.4.1 Show Workspace Directory

The workspace directory can be seen by clicking *Tools* → *Workspace* → *Show workspace directory*.

### 5.4.2 Import and Export Workspace

The ability to export and import the workspace can be a convenient way of transferring settings and local databases from one installation to another. The location of the directory can be configured via the *PowerFactory Configuration* menu.

To import the workspace, select *Tools* → *Workspace* → *Import Workspace*. . . . This is a convenient way to import the entire workspace after a new installation.

To export the workspace, select *Tools* → *Workspace* → *Export Workspace*. . . . The package will be saved as a .zip file.

### 5.4.3 Show Default Export Directory

The selection *Tools* → *Workspace* → *Show Default Export Directory* from the main menu shows the user the directory that is used for the export. In particular, this directory is used for automated backups, e.g. before migration. The location of the directory can be configured via the *PowerFactory Configuration* menu.

## 5.5 Database Migration

When updating *PowerFactory* to a newer version, the database (local or multiuser) is migrated. This migration is done “on-the-fly”, that is, the projects are migrated only as needed.

However, it is possible to perform a complete migration on demand from the context menu of the Database root entry (*right-click* → *Migrate all projects*) and via scripting. For multi-user databases, the option is only available for the Administrator. For single-user databases, any user can execute the complete migration.

The migration status for each project is shown on the project dialog, *Storage* page as described in Section [9.1.2](#).

# Chapter 6

## ***PowerFactory Administration***

This chapter provides details of the *PowerFactory* database and the tasks that are executed by the user “Administrator”, e.g. creating and managing user accounts, user groups and profiles.

The information in this chapter is particularly relevant for a multi-user database (i.e. *Team Edition*), but it can also be of interest to a user with a single-use installation. Key objectives of the user account managing system are to:

- Protect the ‘system’ parts of the database from changes by normal (non-administrator) users.
- Configure and manage access to the database, via options for the authentication mode to be used and options for password management.
- Manage settings relating to data security and privacy.
- Facilitate both the sharing of user data and the restriction of data visibility between one user group and another.

The user account managing system provides each user with their own “private” database space. The user is nevertheless able to use shared data, either from the common system database or from other users, and may enable other users to use data from their private database.

The user account managing system manages this whilst using only one single database in the background, which allows for simple backup and management of the overall database.

The following sections provide the information necessary for the administration of the program’s database, including:

- [6.1: PowerFactory Database Overview](#)
- [6.2: The Administrator Account](#)
- [6.3: Administration Menu](#)
- [6.7: User Interface Customisation](#)
  - [6.7.1: User-defined Tools Toolbar](#)
  - [6.7.2: Profiles](#)

### **6.1 *PowerFactory Database Overview***

A brief introduction to the top level structure of the *PowerFactory* database is convenient before presenting the user accounts and their functionality.

The data in *PowerFactory* is stored inside a set of hierarchical directories. The top level structure is constituted by the following folders:

- **Configuration:** contains company specific customising for user groups, user default settings, project templates and class templates for objects. The configuration folder can only be edited by the Administrator and is read only for normal users.
- **System:** contains all objects that are used internally by *PowerFactory*. The system folder contains default settings provided by *DlgSILENT* and these should not be changed. They are automatically updated upon migration to a new *PowerFactory* version.
- ***DlgSILENT Library:*** contains all standard types and models provided with *PowerFactory*. The main library folder is read only for normal users.
- **User accounts:** contain user project folders and associated objects and settings.

The structure described above is illustrated in Figure 6.1.1

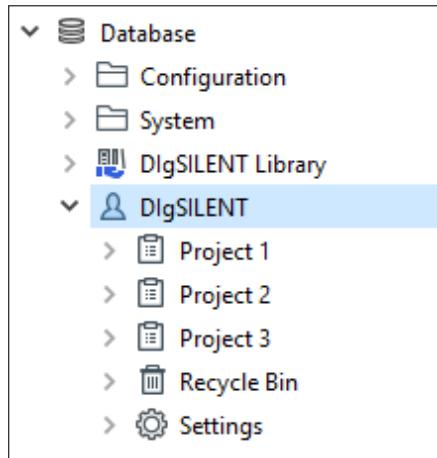


Figure 6.1.1: Basic database structure

An additional library, which will be visible for all the users, can be added by the Administrator to the global area. See section [Custom Global Library](#) for further details.

## 6.2 The Administrator Account

An “Administrator” account is created with the *PowerFactory* installation. Windows administrator rights are **not** necessary to switch to the Administrator user. The main functions of the Administrator are:

- Creation and management of user accounts.
- System database maintenance under the guidance of the *DlgSILENT* customer support.

The Administrator is the only user with permissions to:

- Add and delete users.
- Define users groups.
- Set individual user rights.
- Restrict or allow calculation functions.
- Set/reset user passwords.
- Create and edit profiles (see Section 6.7.2 for details).

- Configure various settings such as housekeeping processes, parallel processing configuration, password security options etc.

The Administrator is also the only user that can modify the folders in the global area. Although the Administrator has access to all the projects of all the users, it does not have the right to perform any calculations.

To log on as Administrator, there are two options:

- Select the shortcut in the Windows Start Menu PowerFactory 20nn →  Administration.
- Log into *PowerFactory* and select via the main menu *Tools* → *Switch User*. Select **Administrator**.

For further information about the Administrator role, refer to the [Advanced Installation and Configuration Manual](#).

Many of the activities carried out by the Administrator are easily accessed via the *Administration* menu; this is found on the main toolbar but is only visible if the user is logged on as Administrator. See Section [6.3](#) for further details.

## 6.3 Administration Menu

To assist the Administrator, an Administration menu is provided to give easy access to the more important settings; this is found on the main toolbar but is only visible if the user is logged on as Administrator. These are the options available from this menu:

### 6.3.1 User Management

The options available here are:

- Show Users...
- Show Groups...
- User Manager...

These are used to manage User accounts and User Groups as described in Sections [6.4](#) and [6.4.2](#).

### 6.3.2 Security and Privacy

The options available here are:

- Audit Log...
- Login Policy...
- Idle Session Timeout...
- External Data Access...
- File Exchange Settings...
- Privacy...

These are described in Section [6.5](#) below.

### 6.3.3 Calculation Settings

The options available here are:

- Parallelisation... : This allows the Administrator to configure the Parallel Computing Manager. See Section [6.6.1](#).
- Linear Programming... : This brings up a dialog which enables the Administrator to configure which internal and external linear programming solvers should be made available to users who are using the Unit Commitment and Dispatch Optimisation module. See Section [40.3.7.2](#).

### 6.3.4 Housekeeping

This gives access to the dialog for setting up Housekeeping tasks. Details can be found in Section [8.1](#).

## 6.4 Creating and Managing User Accounts and Groups

In the case of an installation with a local database, the default name for a *PowerFactory* user is the Windows user name, which is automatically created when *PowerFactory* is started for the first time. In this case the program will automatically create and activate the new account, without using the Administrator account. But if additional new users are required, or when working with multi-user database installations, the management of the users is done by the Administrator (see Section [6.2](#)), using the object *User Manager*.

The User Manager is located in the Configuration folder and can also be accessed from the Administration menu on the main toolbar: *Administration* → *User Management* → *User Manager*. . . .

### 6.4.1 Users

Clicking on the **Add User...** button in the *User Manager* will result in a new user being created, and the User edit dialog will be displayed. The settings are the following:

- *General page*
  - User name: user name that will be used in *PowerFactory*.
  - Full name: full name of the appropriate user. In case, that the parameter *User name* is set to be an abbreviation.
  - Change Password: the Administrator can change the user password here, without knowing the previous password. If this button is clicked by the user itself, the current password has to be entered as well.
  - Force password change: can be selected by the Administrator.  
**Note:** The last two options are only available for some authentication modes (see Section [6.5.2](#))
  - User sharing: by adding different users into the list of permitted users, access for these users can be granted to login to the appropriate user account. In a multiuser database, if User A is in the list of permitted user, they can access the user account.
- *Account page*:
  - Publishing user: by setting this flag, the user can be defined to be a publishing user. This means, that the user is visible to other users within the database and marked with a different symbol within the data manager. This option can be used to provide an user within the multiuser database, who publishes projects.

- User account enabled: this setting can be used to enable/disable the user account.
  - Enable user-specific idle settings: if selected, the global idle settings, described in Section 6.5.3, will be overridden and set for the user separately.
  - Force Authentication server usage: setting this option also requires the definition of an authentication server within the *PowerFactory* Configuration as explained in the [Advanced Installation and Configuration Manual](#).
- *Password Policy page*: on this page, the default password policy (see Section 6.5.2) can be customised by the Administrator for the user.
  - *Licence page*: if a licensed version with a restricted number of functions is used (i.e. you may have 4 licences with basic functionality, but only 2 stability licences), the *Licence* tab may be used to define the functions that a user can access. The *Multi-User Database* option should be checked for all users that will access the multi user database.  
As an alternative to allocating access to certain licence functions to individual users, it is possible to allocate access via User Groups instead. See Section 6.4.2 below.
  - *Parallel Computing page*: here it can be defined whether the user is allowed to use parallel processing possibilities within *PowerFactory*. The “User defined” setting allows the individual user to customise the globally-defined allowed processes number to a lower number if required, for example to free up resources for other applications.
  - *Optimisation page*: the *Unit Commitment* module (see Chapter 40) offers the possibility to use in-built or external linear problem solvers, the latter requiring an additional licence module. Here, the Administrator enables access to the preferred solver(s).

Existing users can be viewed via the Administration menu on the main toolbar: *Administration* → *User Management* → *Show Users*. . . .

The Administrator can edit any user account to change the user name, set new calculation rights or change the password by right-clicking on the user and selecting *Edit* from the context menu.

Any user can edit their own account by means of the User edit dialog. In this case only the full name and the password (depending the authentication mode) can be changed.

---

**Note:** The Administrator is the only one who may delete a user account. Although users can delete all projects inside their account folder, they cannot delete the account folder itself or the standard folders that belong to it (i.e. the *Recycle Bin* or the *Settings* folder).

---

## 6.4.2 User Groups

User groups are a useful way for managing various access rights and permissions within a multi-user database environment. For example, any project or folder in a user account may be shared, either with everybody or with specific user groups. User groups can also be used in conjunction with Profiles (see Section 6.7.2) and for controlling access to licence modules.

When installing *PowerFactory*, the user group “Everybody” is automatically created; additional user groups are created by the Administrator via the User Manager (*Administration* → *User Management* → *User Manager*. . . ) as follows:

- Press the **Add Group...** button.
- Enter the name of the new group, optionally a description and press **Ok**.
- The new group is automatically created in the User Groups directory of the Configuration folder.

Existing groups can be viewed via the Administration menu on the main toolbar: *Administration* → *User Management* → *Show Groups*. . . . The Administrator can change the name of an existing group by

means of the corresponding edit dialog (right-clicking on it and selecting *Edit* from the context menu). Via the context menu, groups can also be deleted.

The Administrator can add users to a group by:

- Copying the user in the Data Manager (right-click on the user and select *Copy* from the context menu).
- Selecting a user group in the left pane of the Data Manager.
- Pasting a shortcut of the copied user inside the group (right-click the user group and select *Paste Shortcut* from the context menu).

Users are taken out of a group by deleting their shortcut from the corresponding group.

The Administrator can also set the Groups *Available Profiles* on the *Profile* tab of the Group dialog.

In addition, the *Licence* page of the User Group can be used to configure which licence modules members of the group will have access to. For any individual user, the licence modules available to that user will be all those selected in that individual user's account set-up, plus any additional licence modules made available to the group(s) to which the user belongs.

For information about sharing projects, refer to Section [21.6 \(Sharing Projects\)](#).

## 6.5 Security and Privacy

This section gives an overview of the security and privacy features which can be managed by the Administrator. Note that more detail is provided in the [Advanced Installation and Configuration Manual](#).

### 6.5.1 Audit Log

The Audit Log is a log of key activities on the database, and is useful for the Administrator of a multi-user database.

#### 6.5.1.1 Enabling the Audit Log

The log can be enabled by the Administrator via the Administration Menu (*Administration* → *Security and Privacy* → *Audit Log...*), where a retention period is also set. By default the log is not enabled, and it should be noted that if the log is enabled then later disabled, all records will be lost. The information in the Audit log is securely held in the database itself.

#### 6.5.1.2 Using the Audit Log

An Audit Log command *ComAuditlog* can be created by the Administrator user and used to access the information in the log. The command options are:

- **Report**, to generate a high-level report to the output window
- **Export**, to export a detailed report
- **Check Integrity** As an additional assurance, it is possible to carry out a data integrity check on the Audit Log data to detect any data manipulation.

A more detailed description of the Audit Log and what it contains can be found in the [Advanced Installation and Configuration Manual](#).

## 6.5.2 Login Policy Options

### 6.5.2.1 Authentication Mode

Here the Administrator determines what authentication (username and password) mode will be used. The options are:

- **PowerFactory authentication** provides built-in user management, where the users must enter their *PowerFactory* usernames and passwords
- **Active Directory authentication** uses the external Windows Active Directory for user authentication
- **No authentication**

If the *Active Directory* authentication is selected, then the user can click on the “...” to the right of each group to select the Active Directory group, then give it an appropriate name within *PowerFactory*. All groups must be within the same domain. It should be noted that only one user in the second group (Administrators) may be logged in at any one time.

### 6.5.2.2 Password Policy

If the authentication mode *PowerFactory authentication* is selected, further options appear, allowing the Administrator to impose rules to enforce regular password changes and/or put in place rules on password quality, such as length and character diversity.

## 6.5.3 Idle Session Timeout

In this option, the Administrator can set a time-limit after which any idle *PowerFactory* session will be terminated. Such a session will be closed down in an ordered way, but it should be noted that unsaved scenario changes will be lost. A session will only be considered as idle if there has been no activity for the prescribed time, where “activity” means user activity such as mouse-clicks or keyboard actions. However, if a command is being executed (for example a long-running simulation, or a script), *PowerFactory* will wait until the command has been completed before terminating the session.

As well as being a good security measure, setting an Idle Session Timeout is useful in a multi-user environment because the licences are released back to the licence server.

## 6.5.4 External Data Access

The External Data Access dialog allows the Administrator to specify permitted addresses in order to manage access to data outside *PowerFactory*. This control is related to *IntUrl* and *IntDocument* objects which are being used to access external data.

## 6.5.5 File Exchange Settings

*PowerFactory* projects may make use of external files to supply data to the project. To facilitate the exchange of such files between users, it is possible to export/import projects in \*.pfds format. This is a zip archive that contains not only the project itself but also the external files that go with it. The file exchange settings allow the Administrator to block any undesirable files either for export or import or both. With wildcards (\*) allowed, the file pattern can block files by name and/or file type.

### 6.5.6 Privacy

This feature is available to allow the database Administrator to manage the visibility of user names. There are two options, which by default are not enabled:

- **Enable recording of modifying user in object**, which if checked will mean that the “Object modified by” information will include the *PowerFactory* user name.
- **Display system account in user object**, which if checked will mean that the Windows username will appear in the *IntUser* object when the user has an active session.

## 6.6 Calculation Settings

### 6.6.1 Parallel Computing Manager

To speed up calculations, various calculation commands offer the option of parallelisation. The settings for the parallel computation of tasks are configured within the *Parallel Computing Manager*. It can be accessed via *Administration* → *Calculation Settings* → *Parallelisation*.

The Parallel Computing Manager has the configuration options as summarised below:

- Basic Options page
  - Master host name or IP
  - Parallel computing method
  - Max. number of processes on local machine
- Communication page
  - Communication method

#### 6.6.1.1 Basic Options Page

**Master host name or IP:** The machine name or IP address of the master host. If only the local multi-core machine is used, the name can be “localhost”.

**Parallel computing method:**

- Local machine with multiple cores: all the parallel processes will be started in the local machine.

**Max. number of processes on local machine:**

- Number of cores: all cores available in the machine will be used for parallel computing.
- Number of cores minus 1: use N-1 cores (N is the number of cores available in this machine).
- User defined: the number of parallel processes as specified by the given table will be started in the local machine. The first column of the table is the number of cores available in the local machine and the second column is the number of parallel processes to be started. For a specific machine, the corresponding row in this table is found according to the number of available cores and then the number of parallel processes in the second column is used. If the row is not found (not specified in this table), all cores are used by default.

### 6.6.1.2 Communication page

**Communication method:** The network data can be transferred to parallel processes either via file or TCP/IP protocol.

## 6.7 User Interface Customisation

This section describe the two different things that the Administrator can do in *PowerFactory* to customise the user interface:

- Configure the User-defined Tools toolbar (Section 6.7.1)
- Configure profiles (Section 6.7.2)

### 6.7.1 User-defined Tools Toolbar

The User-defined Tools toolbar, accessible using the *Change Toolbox* icon ▾, can be configured by the Administrator using the Tool Configuration.

The Tool Configuration (*SetToolconfig*) is accessible via the Administration menu: *Administration* → *Tool Configuration*....

The User-defined Tools toolbar can be used to make commonly-used tools such as scripts and Add On Modules available to users. The commands to be displayed in this toolbar are defined on the *Command* page of the Tool Configuration, the fields described below can be edited.

- **Command:** in this field, the relevant command or script is selected from the location where it has been stored.
  - **Scripts:** scripts may be stored within the Tool Configuration itself or in the *Configuration, Scripts* folder
  - **Com\* objects:** generally, commands *Com\** are stored within the Tool Configuration itself.
  - **Add On Modules:** add on module commands can be stored in the *Configuration, Add On* folder.
- **Edit:** if selected, the DPL command dialog will appear when a Command is executed. If de-selected, the DPL command dialog will not appear when a Command is executed.
- **Temp:** if not selected, a local copy of the associated script will be created in the active study case when the script is run; if selected, this local copy will instead be generated in the user's folder *Settings/Temp/Calculation Cases*, and will be deleted when *PowerFactory* is closed.
- **Icon:** previously created icons can be selected, which will be shown on the menu where the command is placed. If no icon is selected, a default icon will appear (a Hammer, DPL symbol, or default Com\* icon, depending on the Class type).

#### Definition of Icons

Icons can be defined in the *Configuration* → *Icons* folder by selecting the *New Object* icon and then *Icon*. From the Icon dialog, icon images can be imported and exported. Icons should be 24 pixels by 24 pixels in Bitmap format (recommended to be 24-bit format).

## 6.7.2 Profiles

Profiles can be used to configure aspects of the Graphical User Interface, such as toolbars, menus, dialog pages, and dialog parameters.

By default, *PowerFactory* includes “Base Package” and “Standard” profiles, selectable from the main menu under *Tools* → *Profiles*. Selecting the “Base Package” profile limits icons shown on the Main Toolbar to those that are used with the Base Package of the software. The “Standard” profile includes all available *PowerFactory* icons.

Profiles are created in the *Configuration* → *Profiles* folder by selecting the *New Object* icon and then *Profile*. An Administrator can create and customise profiles, and control User/User Group selection of profiles from the *Profile* tab of each group.

Figure 6.7.1 shows the Profile dialog for a new profile, *CustomProfile*, and Figure 6.7.2 illustrates aspects of the GUI that may be customised using this profile. This section describes the customisation procedure.

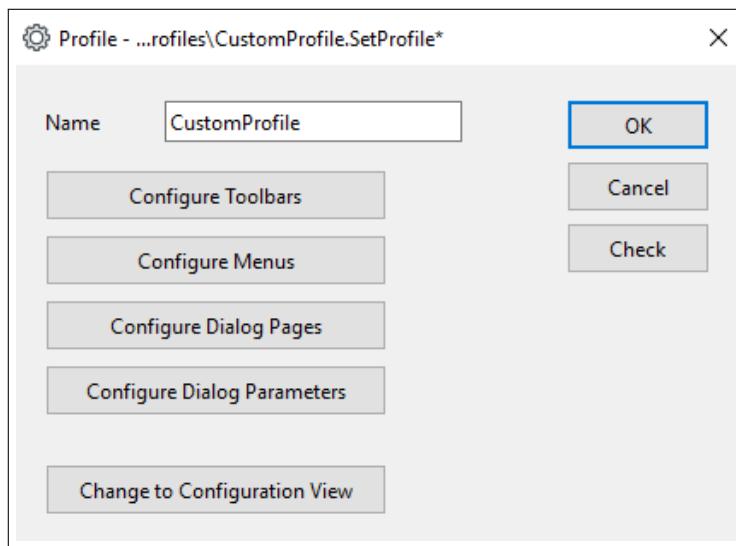


Figure 6.7.1: Profile dialog

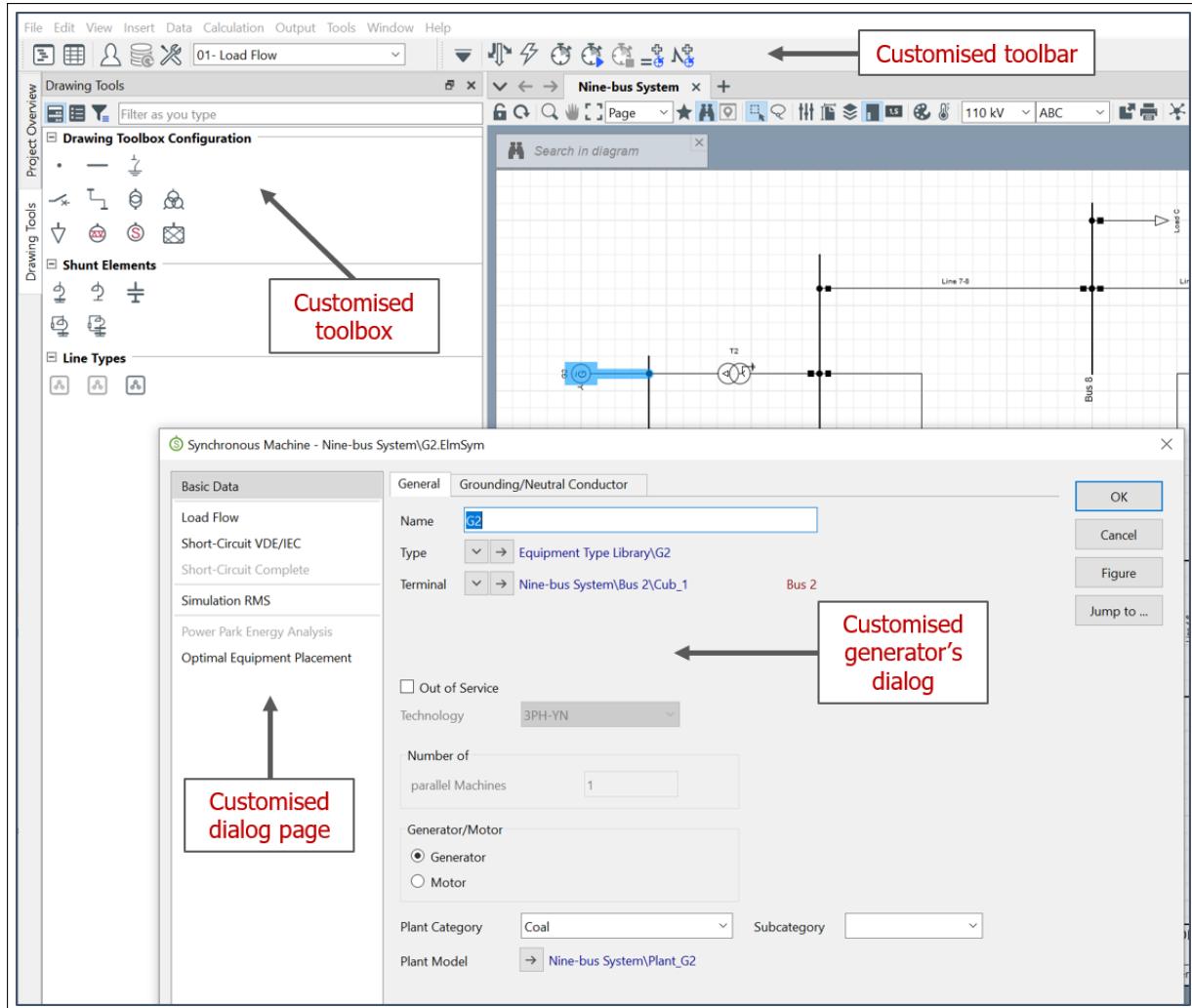


Figure 6.7.2: GUI Customisation using Profiles

**Note:** Note that when a new profile is created, the list of profiles assigned to the groups is cleared (including the group “Everybody”). A user needs at least one profile to log in to *PowerFactory*. Ensure that all users/groups have a profile before closing the program.

### 6.7.2.1 Configuration of Toolbars

The Toolbar Configuration is used to customise the displayed toolbars in the profile and are created as follows:

- Click on the **Configure Toolbars** button
- Click on the icon *New Object* , a new Toolbar Configuration will be created
- Select the toolbar to configure (e.g. Main toolbar, drawing toolbar, etc.)
- Double-click on the field *Toolboxes*
- Click on the icon *New Object*  and select either a *Toolbox Configuration (SetTboxconfig)* or a *Toolbox Group Configuration (SetTboxgrconfig)*

**Note:** The *Toolbox Group Configuration* is used to create switchable toolboxes by the use of several *Toolbox Configurations*.

Figure 6.7.3 shows an example where there is a main toolbox, and a toolbox group. The toolbox group adds a *Change Toolbox* ▾ icon to the menu, which allows selection of the one of the customised Toolbox Configurations stored inside the group: *Base Package* and *Dynamic Simulation* as shown in Figure 6.7.4.

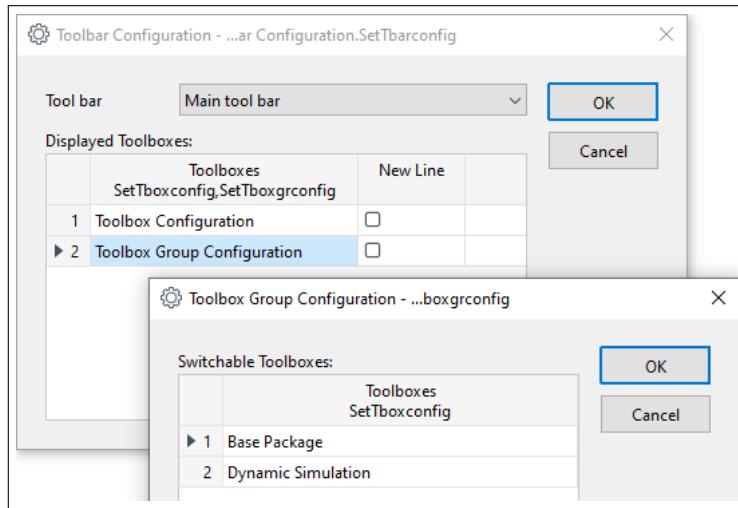


Figure 6.7.3: Toolbar Configuration

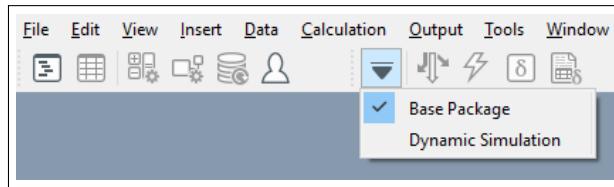


Figure 6.7.4: Toolbox Group

Each toolbox can be customised to display the desired buttons, as illustrated in Figure 6.7.5 for two different configurations.

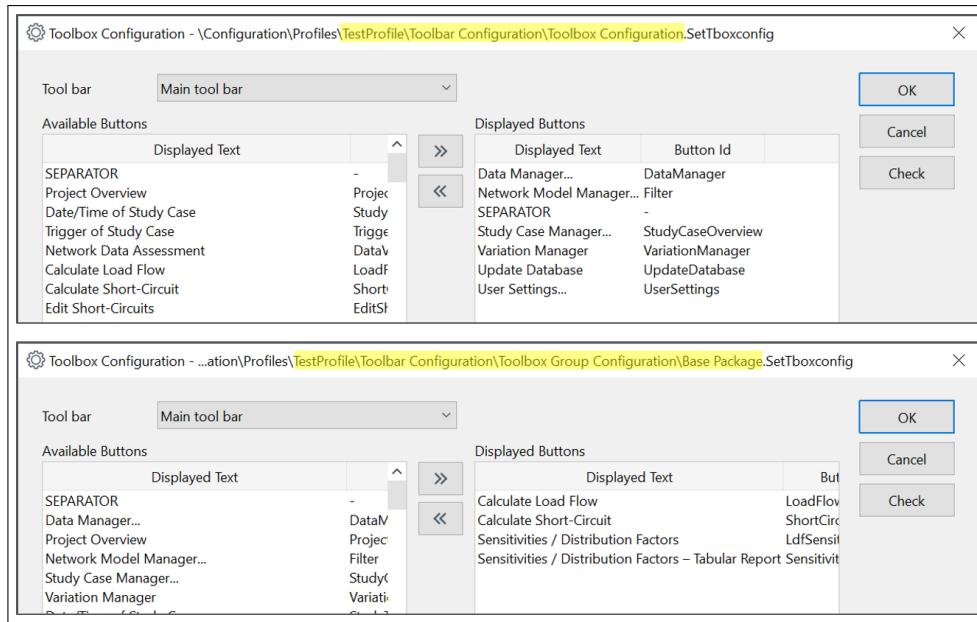


Figure 6.7.5: Toolbox Configurations

Customised commands should be defined using *Tool Configuration*, as described in Section 6.7.1.

### 6.7.2.2 Configuration of Menus

The *Main Menu* and the *Data Manager*, *Graphic*, *Plots*, and *Output Window* menus can be customised from the *Menu Configuration* dialog (shown in Figure 6.7.6), which is created by clicking on the **Configure Menus** button on the Profile dialog and then on the icon *New Object*

The **Change to Configuration View** button of the Profile dialog is used to display description identifiers for configurable items. The Menu Configuration includes a list of entries to be removed from the specified menu. Note that a profile may include multiple menu configurations (e.g. one for each type of menu to be customised).

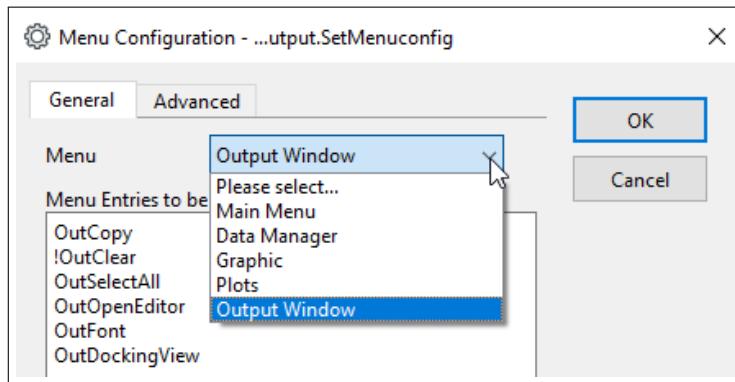


Figure 6.7.6: Menu Configuration

### 6.7.2.3 Configuration of Dialog Pages

The *Dialog Page Configuration*, created by clicking on the **Configure Dialog Pages** button on the Profile dialog, may be used to specify the “Available” and “Unavailable” dialog pages shown when editing

elements, such as illustrated in Figure 6.7.7. Note that Users can further customise the displayed dialog pages from the *Functions* tab of their *User Settings*.

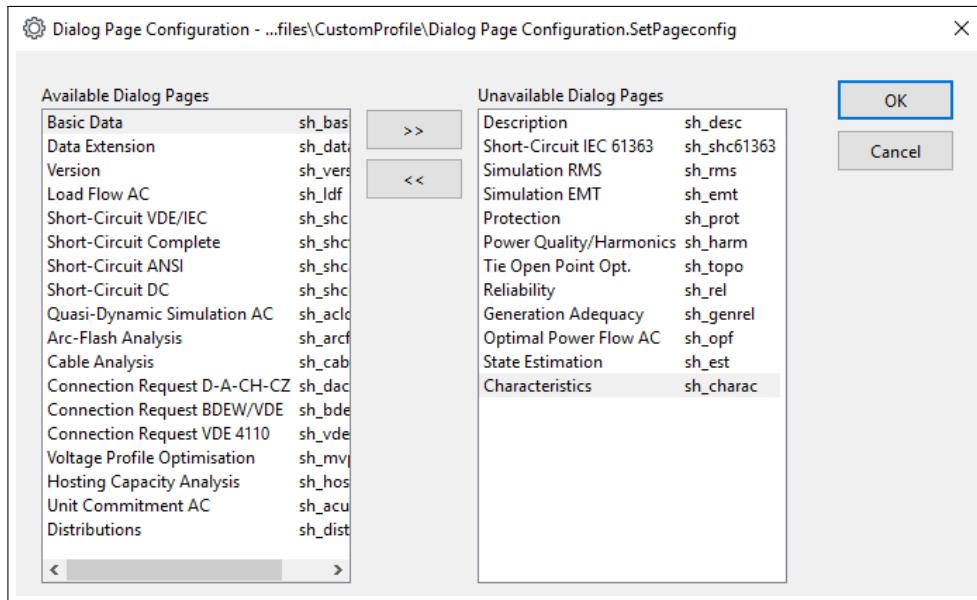


Figure 6.7.7: Dialog Page Configuration

#### 6.7.2.4 Configuration of Dialog Parameters

The *Dialog Configuration*, created by clicking on the **Configure Dialog Parameters** button on the Profile dialog and then on the icon *New Object* , may be used to customise element dialog pages, such as illustrated for a Synchronous Machine element in Figure 6.7.8. “Hidden Parameters” are removed from the element dialog page, whereas “Disabled Parameters” are shown but cannot be modified by the user. A Profile may include multiple dialog configurations (e.g. one for each class to be customised).

Note that if there is a Dialog Configuration for say, *Elm\** (or similarly for *ElmLne*,*ElmLod*), as well as a dialog Configuration for *ElmLne* (for example), the configuration settings will be merged.

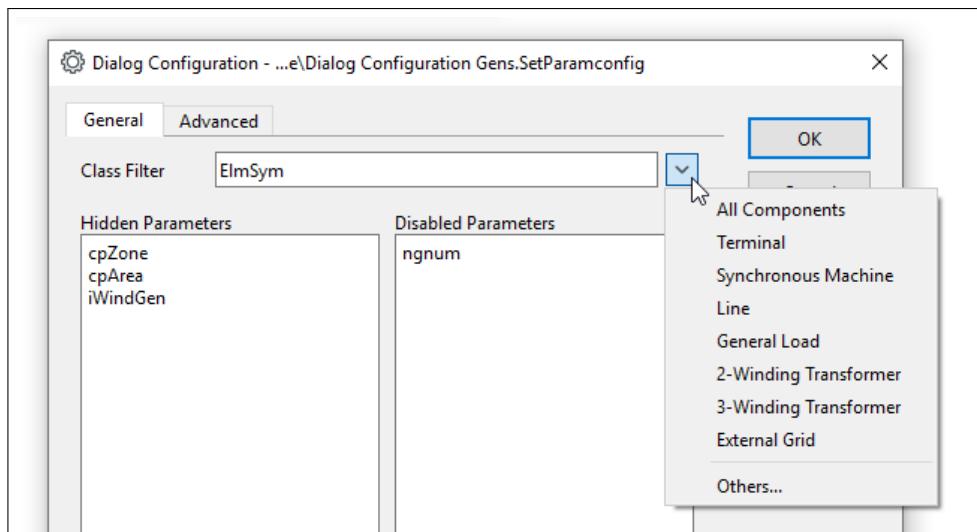


Figure 6.7.8: Dialog Configuration

**Note:** Configuration of Dialog parameters is an advanced feature of *PowerFactory*, and the user should be cautious not to hide or disable parameters that might need to be edited or corrected in order to pass validity checks. Seek assistance from *DlgSILENT* support if required.

---

#### 6.7.2.5 References

Profiles can also contain references to configurations. This allows several profiles to use the same configurations. These referenced configurations can either be stored in another profile or in a subfolder of the “Profiles” folder (e.g. a user-defined profile can use configurations from a pre-defined profile).

# Chapter 7

## User Settings

The User Settings dialog offers general settings which can be configured by the user individually. The dialog may be opened either by clicking the *User Settings* button (👤) on the main tool bar, or by selecting the *Tools → User Settings...* menu item from the main menu.

### 7.1 Window Settings

Here the user has some options for customising the default appearance and layout of the windows.

#### Application colour scheme

This setting offers users the possibility to work in dark mode if they wish. Dark or light mode can be explicitly selected, or taken from the system settings.

#### Tabbed Document Interface

The graphical pages are displayed as tabbed documents, with the tabs by default at the top of each page.

- **Show tab icons:** icons can be shown on the tabs, to help the user distinguish between different pages. With the option turned off, the tabs take up less space.
- **Show confirmation dialog when closing diagrams:** users may prefer not to have the confirmation dialog. Closed (but not deleted) diagrams can of course be re-opened.
- **Tab position:** if the user prefers, the tabs can be shown at the bottom of the graphics.

#### Drawing Toolbox

Here the user can select whether group headers and/or element labels in the drawing toolbox should be shown or not. These options can also be changed within the toolbox itself.

### 7.2 Data/Network Model Manager Settings

The Data/Network Model Manager settings include:

### Browser

- **Save data automatically (tabular input):** the Data Manager and the Network Model Manager will not ask for confirmation every time a value is changed in the data browser when this option is selected.
- **Confirm delete action:** causes a confirmation dialog to appear whenever something is about to be deleted.

### Data Manager

- **Sort automatically:** specifies that objects are automatically sorted (by name) in the data browser.
- **Remember last selected object:** The last selected object will be remembered when a new Data Manager window is opened.
- **Always open new Data Manager:** if not enabled (default), using the  icon or Ctrl+D to open the Data Manager will present the existing Data Manager if one is already open; if enabled, a separate new Data Manager tab will be created.
- **Always open new Network Model Manager:** if not enabled (default), using the  icon to open the Network Model Manager will present the existing Network Model Manager if one is already open; if enabled, a separate new Network Model Manager tab will be created.

### Operation Scenario

- If **Save active Operation Scenario automatically** is enabled, the period for automatic saving must be defined.
- **Automatically migrate to current configuration during activation:** the default is for this option to be selected. This means that after migration to a new version of *PowerFactory*, the operation scenarios will be migrated when a project is activated, and any new scenario data attributes will be assigned values based on the current status of the object. If the option is not selected, the scenario will not be migrated (and values will not be assigned) until that scenario is saved, and this may be the user preference. Note however that the scenarios have to be migrated to the current *PowerFactory* version if the Operation Scenario Manager (see Section 16.4) is used.

### Export/Import data (DZ/DZS)

Configures the export and import as follows:

- **Binary data:** saves binary data, such as results in the result folders, to the 'DZ' export files according to selection.
- **Export references to deleted objects:** will also export references to objects which reside in the recycle bin. Normally, connections to these objects are deleted on export.
- **Export 'Modified by':** enables the export of information about who last changed an object (attribute 'modified by'). This information could conflict with data privacy rules and is therefore configurable.

### Custom library

In the *Used library* field, the user can specify the additional global library to be displayed when assigning a type.

More information about creating a custom global library is available in Section 4.5.2: [Custom Global Library](#).

### Reset column widths

When this button is pressed, the width of the columns in the Data Manager and Network Model Manager are set to their default value.

### **Hotkeys**

This button brings up a filtered list of hotkey assignments, showing just those context-dependent hotkeys applicable to the Data Manager or Network Model Manager.

## **7.3 Graphic Windows Settings**

### **7.3.1 General tab**

#### **Open desktop on study case activation**

Causes the graphics windows to re-appear automatically when a study case is activated. When not checked, the graphics window must be opened manually via *Window → Desktop*.

#### **Grid representation**

The style and colour options allow the user to configure the appearance of the background grid, which is visible when freeze mode is turned off. The colour can be set back to default using a button at the bottom of the page.

#### **Mark in Graphic**

- The **colour** and **opacity** used when the objects are marked in the graphics can be defined.
- **Highlight small elements using additional markers:** sometimes small objects such as terminals are hard to spot even when highlighted. If this option is selected, the position of such objects will be indicated using a marker.
- **Zoom-in on marked elements:** if this is selected, the graphic where the object is to be shown will be zoomed in so that the object can be more easily seen. The level of zoom for both schematic and geographic diagrams can be configured by the user.

#### **Background colours**

To change the configuration of the background colour for:

- **Window:** graphic windows
- **Graphic page:** diagrams
- **Model:** dynamic model elements
- **Model type:** dynamic model types

The colours can be set back to default using the button **Reset colours** at the bottom of the page.

#### **Limit number of open site and substation diagrams**

The user may set a limit to restrict the number of opened graphic pages.

### 7.3.2 Advanced tab

#### Cursor

Defines the cursor shape:

- **Arrow:** a normal, arrow shaped cursor.
- **Tracking Cross:** a small cross.

#### Acceleration of Zooming

The higher the Acceleration factor, the more zoom there will be for a given mouse operation.

#### Update Graphic while Simulation is running

The graphic will be updated during the simulation.

#### General Options

- **Show Text only if height will be least:** text smaller than the selected size will not be shown
- **Diagram window margin:** the graphical pages are by default shown against a dark background, with a small margin. The size of that margin can be adjusted here.
- **Open diagrams in freeze mode on study case activation:** if selected (default option), diagrams cannot be modified until the freeze button is clicked.
- **Allow resizing of branch objects:** if the option is enabled, the user can left click an edge element within the single line diagram and then resize it.
- **Show “Edit Graphic Object” in context menu:** if the option is enabled, when the user right-clicks on an element within the single line graphic, the option *Edit Graphic Object* will be offered.
- **Snap textboxes:** by default, this option is not enabled, allowing the user to position text-boxes precisely. However, selecting the option makes it easier to align text boxes with each other.

## 7.4 Output Window Settings

The output window is described in detail in Chapter 4, section [The Output Window](#).

#### Pop up collapsed Output Window

By default, the output window is “collapsed”; although it can be viewed at any time, there are also options to have it automatically pop up under certain conditions. The default selection is just “on new error messages”.

#### Enable Message filter

When un-checking this box, the filter buttons are removed from the output window.

#### Displayed Messages

This is where the filters used in the output window are defined. This, however, can be directly done in the output window.

### Message format

- **No date and time:** the messages in the output window will be printed without a time stamp.
- **Date and time to the second:** the date and time of the system up to the second will be shown in every line of the output window.
- **Date and time to the millisecond:** the date and time of the system up to the millisecond will be shown in every line of the output window.
- **Full object names:** when an object is printed, the complete name (including the location path) is printed.

### Font

The font used in the output window is set by clicking on the icon .

### Show confirmation dialog before clearing messages

This option is normally checked, to avoid users accidentally losing messages that they need, but deselecting it allows users to clear the output more quickly.

## 7.5 Profile Settings

*PowerFactory* provides standard profiles which define the configurations of the toolbars seen by the users. It is also possible for the Administrator to set up additional profiles, in order to provide customisation for different users (see Section 6.7.2 for details).

Here, the user can select the required profile and see the configuration details.

Also on the profile page, are settings relating to Hotkeys and the Quick Access menu (see Section 7.13):

- **Edit Hotkeys...** to bring up the *Edit Hotkeys...* dialog, also accessible via the main *Tools* menu.
- **Confirm reassignment of key sequence:** selected by default, this ensures that the user has to confirm reassigning a key sequence if another hotkey already uses it.
- **Confirm hotkey reset if it leads to reassignment:** selected by default, this ensures that the user has to confirm if resetting a hotkey to default will affect another hotkey.
- **Confirm reset of multiple hotkeys:** selected by default, this ensures that the user has to confirm if resetting multiple hotkeys even if confirmation for resetting individual hotkeys is not requested.
- **Edit Quick Access...** to bring up the *Edit Quick Access...* dialog, also accessible via the main *Tools* menu.

## 7.6 Functions Settings

The functions settings page provides check boxes for the function modules that are accessible from the Data Manager or from the object edit dialogs. The user may choose to see only certain modules in order to “unclutter” dialogs.

This may also be used to protect data by allowing only certain calculation functionality to be seen by certain users. This is particularly useful in a multi-user environment or when inexperienced users utilise *PowerFactory*.

## 7.7 Editor Settings

The editor used for DPL scripts, DSL equations and, if selected, Python scripts, can be configured on this page.

### Options

- **Enable Auto Indent:** automatically indents the next line.
- **Enable Backspace at Start of Line:** will not stop the backspace at the left-most position, but will continue at the end of the previous line.
- **View Blanks and Tabs:** shows these spaces.
- **Show Selection Margin:** provides a column on the left side where bookmarks and other markings are shown.
- **Show Line Numbers:** shows line numbers at the beginning of each line. This option is very useful when locating errors in the script.
- **Enable Autocomplete:** a list of possible functions will be shown when writing a word inside the editor.
- **Tab Size:** defines the width of a single tab.

### Tabs

Toggles between the use of standard tabs, or to insert spaces when the tab-key is used.

### Colours...

Pressing this button opens the *Editor colour settings* dialog from which it is possible to specify a different colour scheme for each programming language used in the software as well as a separate colour scheme for use in the descriptive text fields of *PowerFactory* database objects. Different colours can be assigned for each different predefined class of data. By default a preview mode is shown where the impact of the chosen colouring scheme on an editable sample of text or code can be seen. An overall summary of the selected colour schemes can be shown by pressing the *Overview* button and from this view, colouring schemes can also be adjusted as well as copied and pasted from one column (programming language) to another. The default colouring scheme can be restored for each programming language independently by selecting the *reset* button, when the relevant tab is selected.

The button  can be used to specify the appearance of the text used within the *PowerFactory* editor. The font, font style, size, effects and writing system can all be defined.

### Hotkeys...

This button brings up a filtered list of hotkey assignments, showing just those context-dependent hotkeys applicable when using editors.

## 7.8 Colours Settings

To make it easier for users to identify the different sources of data easily, background colouring of the data fields is used, both in the network model manager and in the element dialogs.

As can be seen in Figure 7.8.1, the user can select different colours. See Section 4.7.8 for information about configuring colours. In addition, because data might belong to more than one category (e.g. operation scenario data which also has associated characteristics), the user can set priorities according

to which information is considered more important. A **Reset to default** button allows the user to reset all these settings to their default values.

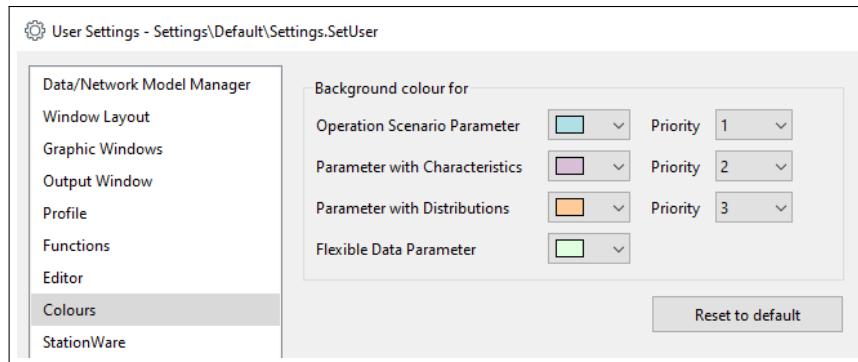


Figure 7.8.1: Data colouring options

## 7.9 StationWare Settings

When working with *DIGSILENT's StationWare* connection options are stored in the user settings. The connection options are as follows:

### Service Endpoint

Denotes the *StationWare* server name. This name resembles a web page URL and must have the form:

- `https://the.server.name/psmsws/PSMSService.asmx` or
- `https://192.168.1.53/psmsws/PSMSService.asmx`

`https` denotes the protocol, `the.server.name` is the computer name (or DNS) of the server computer and `psmsws/PSMSService.asmx` is the name of the *StationWare* application.

---

**Note:** The default *StationWare* configuration requires SSL for the *StationWare* applications (web GUI and web services). Use `http` instead of `https`, if SSL is not enabled for your *StationWare* applications.

---

### Username/Password

Username and Password have to be valid user account in *StationWare*. A *StationWare* user account has nothing to do with the *StationFactory* user account. The very same *StationWare* account can be used by two different *PowerFactory* users. The privileges of the *StationWare* account actually restrict the functionality. For device import the user requires read-access rights. For exporting additionally write-access rights are required.

## 7.10 Offline Settings

These settings are only relevant if the installation has been configured to enable Offline mode to be used (see Section 8.2). Users will normally leave these settings at their default values.

- **Id contingent size:** it is necessary, when starting an Offline session, to reserve object IDs in the main database so as to avoid any conflicts. This parameter specifies the number of IDs to be reserved, and therefore the number of objects that could be created.
- **Id contingent warning threshold:** once the user reaches this percentage of the above number of IDs, a warning is issued. This can act as a prompt for timely resynchronisation of the user's changes.

## 7.11 Parallel Computing

The settings for parallel computing are centrally defined by the Administrator, as described in Section [6.6.1](#).

However, an individual user may wish to modify the settings and it is possible to do so on this page. A typical reason for this would be that the user wishes to make use of parallel computation but does not want to use the maximum allowed number of cores, because of a need to work on other applications outside *PowerFactory* at the same time. Here the user can opt to use fewer (but not more) cores than the maximum set by the Administrator.

On the Advanced tab, there are two more options:

- **Max. waiting time for process response:** the default setting here is 50 s. A shorter time risks stopping processes which are still executing tasks, and a longer time could slow down the overall execution time. This setting should be changed only with care.
- **Transfer complete project to all processes:** the default is for this *not* to be selected. This is because normally *PowerFactory* transfers all data required. Only in exceptional circumstances should it be set, and the user should be aware that there is likely to be an adverse effect on performance.

## 7.12 Miscellaneous Settings

### 7.12.0.1 General

#### Localisation

- **Decimal Symbol:** selects the symbol selected to be used for the decimal point. It can be either taken from the system, or defined by the user.

#### Retention of results after network change

When the option *Show last results* is selected, modifications to network data or switch status etc. will retain the results, these will be shown on the single line diagram and on flexible data pages in grey until the user reset the results (e.g. by selecting Reset Calculation, or conducting a new calculation).

#### Check for application updates

*PowerFactory* will remind the User if there are new updates available for the software. In this field is defined how often *PowerFactory* shall check for available updates. By default there will be a reminder every 14 days. The possible options are:

- **Manually:** the User will check for updates manually, no reminder will be shown.
- **On each application start:** a reminder will be shown every time *PowerFactory* is started.

- **According to interval:** a reminder will be shown according to the time defined in this field.

#### Edit Filter before Execute

If this is selected, when the user uses a filter, a dialog box appears and the user may first change something or immediately press **Apply**; if this option is not checked then filters are just applied straightaway.

#### Allow housekeeping task to operate when user is connected

This option is only active if housekeeping is enabled.

#### 7.12.0.2 Confirmation Dialogs

- **Show 'Remove Contingencies' confirmation dialog in Contingency Analysis:** when existing contingencies will be removed because they will be overwritten by new ones (e.g. when using the Contingency Definition tool), the default behaviour is to ask the user to confirm, in case of error. If the user prefers not to be asked, this option should be deselected.
- **Show 'Reset Calculation' confirmation dialog:** this option can be deselected if the user does not want to be asked for confirmation when using the *Reset Calculation* button.
- **Show 'Exit' dialog:** when the user closes *PowerFactory*, a confirmation dialog normally appears. In addition to confirming that the user wishes to exit, it offers options relating to project purging and recycle bin emptying. There is also a *Show on exit* option which can be un-checked. This user setting is another way of disabling the confirmation dialog, or of course re-enabling it.
- **Show backup reminder dialog:** if this option is set, the user will receive a reminder about making a database backup when logging in.
- **Show 'Variation is recording' notification:** if selected (default value), a small window with the message "Variation is recording" will be displayed if an expansion stage is active and a modification is made that will be stored in the expansion stage.
- **Confirm delete action:** if this option is set, a confirmation dialog will appear whenever something is about to be deleted.
- **Confirm when closing groups with multiple tabs:** if this option is set, the user will be asked for confirmation before a group of multiple tabs (for example a set of tabular reports) is closed.
- **Confirm when closing merge browser:** if this option is set, the user will be asked for confirmation before the merge tool browser is closed. This avoids inadvertent loss of assignments.
- **Confirm reassignment of key sequence:** selected by default, this ensures that the user has to confirm reassigning a key sequence if another hotkey already uses it.
- **Confirm hotkey reset if it leads to reassignment:** selected by default, this ensures that the user has to confirm if resetting a hotkey to default will affect another hotkey.
- **Confirm reset of multiple hotkeys:** selected by default, this ensures that the user has to confirm if resetting multiple hotkeys even if confirmation for resetting individual hotkeys is not requested.

## 7.13 Hotkeys and Quick Access

### 7.13.1 Hotkeys

Hotkeys are shortcuts used to speed up work within *PowerFactory*. Fixed standard hotkeys such as F5 for refresh or Ctrl+C for copying are supplemented by user-definable hotkeys for actions such as running

a load-flow calculation (default: F10), opening the User Settings dialog (default: Ctrl+U) or marking a selected object in a graphic (default: Ctrl+M).

There are two categories for actions that the user can assign key sequences to:

- **Global actions:** These are actions triggered via an icon in one of the main toolbars or selecting an item from a main menu. They do not depend on the currently active window or control but can only be executed if no modal dialog is open.
- **Context-dependent actions:** These actions can only be executed if a specific window or control is currently active.

It is worth noting that for commands, it is possible to assign two different key sequences, one to open the command dialog and another to directly execute the command. An example of this is the existing default settings of Ctrl+F10 to just open the Load Flow command dialog, and F10 to execute the Load Flow command without opening the command dialog.

A particular hotkey sequence can be used for more than one context-dependent action (in different contexts), but a hotkey assigned to any global action must be unique.

### 7.13.1.1 Assigning hotkeys (*Edit Hotkeys...* dialog)

The *Edit Hotkeys...* dialog can be opened from the main menu using *Tools* → *Edit Hotkeys...*. It can also be accessed from any main toolbar icon via right-click, or from the User Settings dialog, Profile page. (Note that these settings are not part of the User Profile as such, but are just part of the more general user customisation.)

There are two drop-down menus at the top of the dialog:

- **Category** enables the user to show the list of Global Hotkeys, Context-dependent Hotkeys or Fixed Hotkeys.
- **View** is only available if Global Hotkeys is selected and offers the choice between a list of actions from the main toolbars (including all the calculation toolboxes) or a list of actions from the main *PowerFactory* menu.

In the panel below, available actions are listed, and hotkeys can be assigned in one of two ways:

- Click once in the relevant Key Sequence field and type the key sequence directly; or
- Use right-click or double-click to bring up the *Set Key Sequence* dialog, which provides more options and information.

At the bottom of the dialog there is an option to *create a hotkey sheet*, which enables the user to generate a PDF that lists all the hotkeys currently defined.

There are also options to **Export** the current hotkey configuration to file, or **Import** a new hotkey configuration, enabling users to share a common configuration of hotkeys. Note that an import will completely replace the existing hotkey configuration.

Finally, a **Reset** button allows the user to reset all hotkeys to default.

### 7.13.1.2 Allowed key sequences and potential conflicts

The key sequences that can be allocated by the users are subject to some constraints, but typical key sequences that can be used are function keys (with some exceptions) and many combinations consisting of one or more modifiers (e.g. Ctrl, Alt) and a main key.

To avoid unexpected and unwanted behaviour, some key sequences are prohibited:

- Key sequences commonly used in Windows, such as Ctrl+C and Ctrl+Alt+Del
- Key sequences used to access entries in the main *PowerFactory* menu (such as Alt+F to open the File menu)
- Key sequences used by the fixed hotkeys

If hotkeys are defined using the *Set Key Sequence* dialog rather than simply typing into the field, the user will be informed if the sequence entered is invalid.

Users should also be aware that some key sequences (such as Window+L) are deliberately *not* captured by *PowerFactory*, so as not to disrupt their normal function. Similarly, if the user is running another application that captures and recognises certain keystrokes, these combinations may never reach *PowerFactory*.

Another consideration is that the user could define a hotkey using a sequence that has already been assigned to a different action. By default, there are checks in place to prevent accidental overwriting. This is configurable; see Section [7.5](#).

#### 7.13.1.3 Assigning hotkey to toolbar icons

As an alternative to using the *Edit Hotkeys...* dialog for assigning hotkeys to actions from the toolbars, it is possible to simply right-click on any active icon and *Assign key sequence...*. This brings up the *Set Key Sequence* dialog directly.

#### 7.13.1.4 Assigning hotkey to user-defined scripts

To execute user-defined scripts via a hotkey, scripts can be added to the User-defined Tools toolbar, then a hotkey can be assigned to the relevant command.

### 7.13.2 Quick Access

*PowerFactory* offers a Quick Access menu, which - as long as no modal dialog is open - can be called up from anywhere in the GUI via the hotkey Ctrl+Q (this is configurable).

#### 7.13.2.1 Managing Quick Access (*Quick Access...* dialog)

The Quick Access... dialog can be opened from the main menu using *Tools* → *Edit Quick Access...*. It can also be accessed from any main toolbar icon via right-click, or from the User Settings dialog, Profile page. (Note that these settings are not part of the User Profile as such, but are just part of the more general user customisation.)

At the top of the dialog are:

- A **View** drop-down menu, which offers the choice between a list of actions from the main toolbars (including all the calculation toolboxes) or a list of actions from the main *PowerFactory* menu.
- A **Filter** field, which helps the user to find the required action.

Selected items are added or removed using the double arrows between the two panes, and the order of the list of menu items can be managed using the **Move up** and **Move down** buttons.

There are also options to **Export** the current Quick Access configuration to file, or **Import** a new Quick Access configuration, enabling users to share a common configuration of Quick Access items. Note that an import will completely replace the existing Quick Access configuration.

### 7.13.2.2 Adding/removing toolbar actions

As an alternative to using the *Edit Quick Access...* dialog for adding actions to the Quick Access menu, it is possible to simply right-click on any active icon and *Add <command> to Quick Access ....*

### 7.13.2.3 Assigning user-defined scripts to the Quick Access menu

User-defined scripts can be added to the User-defined Tools toolbar, where they are then available for inclusion in the Quick Access menu.

# Chapter 8

# Additional Features for a Multi-User Database

The information in this chapter is relevant for a multi-user database (i.e. *Team Edition*) since it provides information on how to configure features that are only available for this type of database; these are:

- Housekeeping - see Section [8.1](#)
- Archiving - see Section [8.1.4](#)
- Offline Mode - see Section [8.2](#)

The general information about the use of a multi-user database, i.e. selecting the type of database and user administration is available in chapters [PowerFactory Installation and Configuration](#) and [PowerFactory Administration](#).

More detailed descriptions of the installation, database settings and additional information can be found in the [Advanced Installation and Configuration Manual](#).

## 8.1 Housekeeping

### 8.1.1 Introduction

Housekeeping automates the administration of certain aspects of the database; in particular purging projects, emptying user recycle bins and the deletion of old projects. Housekeeping is triggered by the execution of a Windows Scheduled Task; this can be set up to run at night, thus improving performance during the day by moving regular data processing to off-peak times. An additional benefit to housekeeping is that users will need to spend less time purging projects and emptying recycle bins, something that can slow down the process of exiting *PowerFactory*.

Housekeeping is only available for multi-user databases (e.g. Oracle, SQL Server). For details on scheduling housekeeping, see the [Advanced Installation and Configuration Manual](#).

### 8.1.2 Configuring Permanently Logged-On Users

Normally, housekeeping will not process data belonging to logged-on users; however, some user accounts (e.g. those for a control room) may be connected to *PowerFactory* permanently. These users can be configured to allow housekeeping to process their data while they are logged on.

This is done from the User Settings dialog, Miscellaneous page, by selecting *Allow housekeeping task to operate when user is connected*. When logged in as Administrator, the User Settings dialog of the users can be accessed in folder *User → Settings → Default*.

Regardless of this setting, housekeeping will not operate on a user's active project.

### 8.1.3 Configuring Housekeeping Tasks

The Housekeeping command (*SetHousekeeping*) is used to control which housekeeping tasks are enabled. A Housekeeping command can be created and modified by the user "Administrator" via the menu *Administration → Housekeeping*. . . . The following sections discuss the different housekeeping tasks available in the Housekeeping dialog.

### 8.1.4 Project Archiving

Project archiving provides the following options:

- **Disable:** Archiving is not used.
- **Immediate archiving by the user:** by selecting "Archive" from the context menu, the project will be immediately archived and placed in the archive directory.
- **Deferred archiving by Housekeeping job:** by selecting "Archive" from the context menu, the project will be immediately archived, but not placed in the archive directory. This will happen automatically depending on the Housekeeping settings.

**Important:** The archive directory can be defined under "Tools\Configuration\Database\Archive"

A project cannot be archived unless it is deactivated. By right-clicking on the project a context menu will appear. By selecting "Archive", the project will be moved to the *Archived Projects* folder of the user (*IntUser*). If specified in the Housekeeping archiving options, the project will be immediately placed in the archive directory.

Conversely, archived projects may also be restored. To restore an archived project, the user must select "Restore" from the context menu which appears after right-clicking on a deactivated project.

### 8.1.5 Configuring Deletion of Old Projects

If the option *Remove projects based on last activation date* has been selected in the Housekeeping dialog, when the Housekeeping is executed, for each user, each project will be handled according to the selected *Action*.

The *Action* options are:

- **Delete project:** deletes the project
- **Archive project:** archives the project

The project properties determine whether a project can be automatically deleted or archived.

The settings are found on the *Storage* page of the project dialog, and by default the option "Housekeeping project deletion" is not selected. If it is selected, a retention period can be specified, by default 60 days.

These defaults can be changed for new projects by the Administrator using a template project (folder Configuration/Default).

The settings for multiple projects can be changed in a Data Manager on the Storage tab, as shown below in Figure 8.1.1. A value of '1' is equivalent to the Housekeeping option *Delete project* being selected.

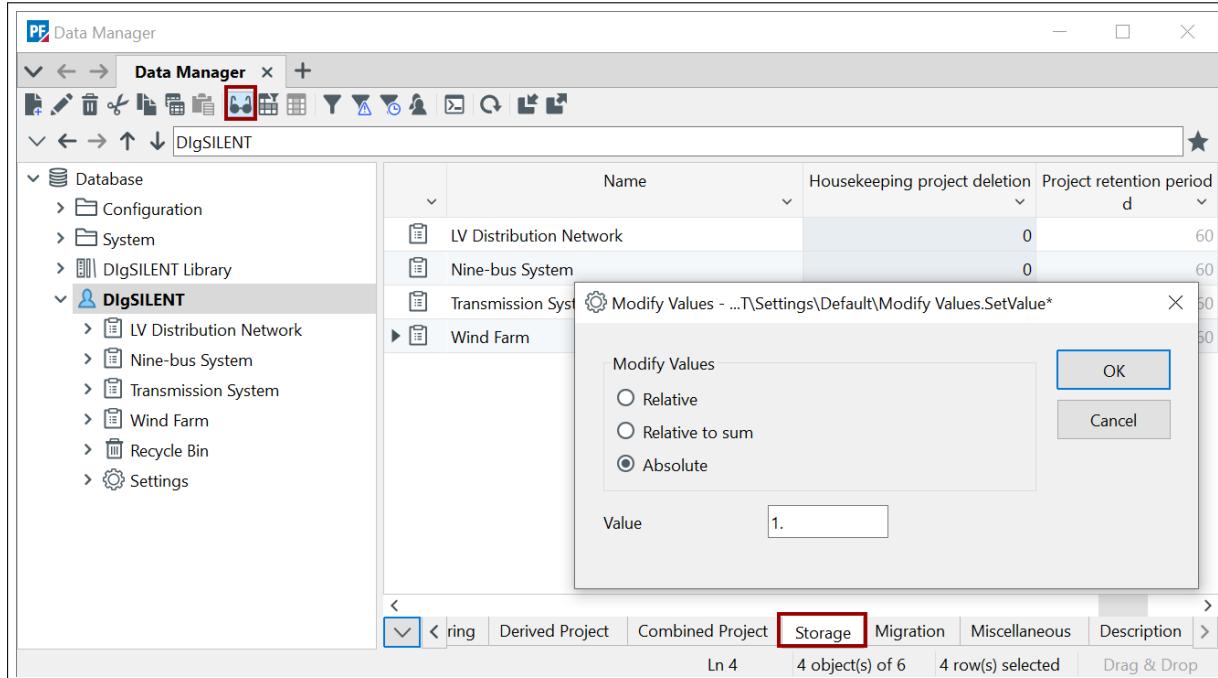


Figure 8.1.1: Setting parameters for multiple projects

A project will be deleted/archived by the housekeeping task if it meets the following criteria:

1. The project is configured for automatic deletion/archiving on the *Storage* page of the project properties.
2. The last activation of the project is older than the retention setting on the project.
3. It is not a base project with existing derived projects.
4. It is not a special project (e.g. User Settings, or anything under the System or Configuration trees).
5. The project is not locked (e.g. active).
6. The owner of the project is not connected, unless that user is configured to allow concurrent housekeeping (see Section 8.1.2).

### 8.1.6 Configuring Purging of Projects

A *PowerFactory* project contains records of changes to data, which makes it possible to roll back the project to an earlier state using versions (see Section 21.2). However, as the user works with the project and makes changes to it, the number of records increases and it is useful to remove older, unwanted records in a process known as “purging”.

If *Purge projects* has been ticked in the Housekeeping dialog, when the Housekeeping is executed, each project will be considered for purging. A project that is already locked (e.g. an active project) will not be purged.

The criteria used by Housekeeping to purge a project are:

- If the project has been activated since its last purge.

- If it is now more than a day past the object retention period since last activation, and the project has not been purged since then.
- If the project is considered to have invalid metadata (e.g. is a pre-14.0 legacy project, or a PFD import without undo information).

Once housekeeping has been configured to purge projects, the automatic purging of projects on activation may be disabled by the user, thus preventing the confirmation dialog popping up. To do this, the option *Automatic Purging* should be to *Off* on the *Storage* page in the Project Properties dialog. This parameter can also be set to *Off* for multiple projects (see Section 8.1.5 for details).

### 8.1.7 Configuring Emptying of Recycle Bins

If *Delete recycle bin objects* is set in the Housekeeping dialog, when Housekeeping is executed, each user's recycle bin will be examined.

Entries older than the number of days specified in the Housekeeping dialog will be deleted.

There is also an option *Prevent manual clearing of recycle bin*. If the Administrator sets this, individual users will not be able to clear their own recycle bins, or delete individual objects therein.

### 8.1.8 Delete orphaned archive files

This option is provided to allow the housekeeping of archived projects. If the reference to an archived project is subsequently removed from the database, that archived project is “orphaned”: there is no point in retaining it in the archive as it could no longer be restored to the database. This option allows the housekeeping process to delete such orphaned archive files.

### 8.1.9 Purge of object keys

Object keys are unique identifiers that are created whenever projects are exported. Their purpose is to facilitate the “marrying-up” of a derived project with its base project when both are imported into another database. It is possible to manually purge object keys for an individual project, and this will delete object keys relating to objects which no longer exist. This option in the housekeeping settings allows that process to be automated, but as the analysis of all projects in this way is resource-intensive, this option should only be selected following *DlgSILENT* advice.

### 8.1.10 Monitoring Housekeeping

In order to ensure that housekeeping is working correctly, it should be regularly verified by an administrator. This is done by inspecting the HOUSEKEEPING\_LOG table via SQL or the data browsing tools of the multi-user database. For each run, housekeeping will insert a new row to this table showing the start and end date/time and the completion status (success or failure). Other statistics such as the number of deleted projects are kept. Note that absence of a row in this table for a given scheduled day indicates that the task failed before it could connect to the database. In addition to the HOUSEKEEPING\_LOG table, a detailed log of each housekeeping run is stored in the log file of the housekeeping user.

### 8.1.11 Summary of Housekeeping Deployment

The basic steps to implement housekeeping are:

1. Set up a Windows Scheduled Task, as described in the *PowerFactory* Advanced Installation and Configuration Manual.
2. Configure those users expected to be active during housekeeping, as described in Section [8.1.2](#).
3. Configure the Housekeeping dialog as described in Section [8.1.3](#).
4. If using the project deletion/archiving task, configure automatic deletion/archiving properties for new projects, as described in Section [8.1.5](#).
5. If using the project deletion/archiving task, configure automatic deletion/archiving properties for existing projects, as described in Section [8.1.5](#).
6. Regularly monitor the HOUSEKEEPING\_LOG table to verify the status of housekeeping runs, as described in Section [8.1.10](#).

## 8.2 Offline Mode User Guide

This section describes working in offline mode. Installation of the offline mode is described in the [Advanced Installation and Configuration Manual](#).

The Offline Mode concept was introduced with users of multi-user databases in mind. Users who have a Team Edition licence make use of a multi-user database because of the benefits it brings in terms of sharing data. Sometimes, however, users wish to work detached from the main database. The following terms are used in this section:

- **Online:** Connected to, and working in, the multi-user database
- **Offline:** Disconnected from the multi-user database and working in a local database cache.

### 8.2.1 Functionality in Offline Mode

#### 8.2.1.1 Start Offline Session

Preconditions:

- A *PowerFactory* user account must already exist in the online database. The *PowerFactory* “Administrator” user is able to create user accounts.
- A user can only start an offline session if they are not currently logged on.

---

**Note:** The Administrator user is only allowed to work in online mode (not in offline mode).

---

To create an offline session, follow these steps:

- Start *PowerFactory*. In the Log-on dialog enter the user name and password.
- On the *Database* page, enter the *Offline Proxy Server* settings (see Figure [8.2.1](#))

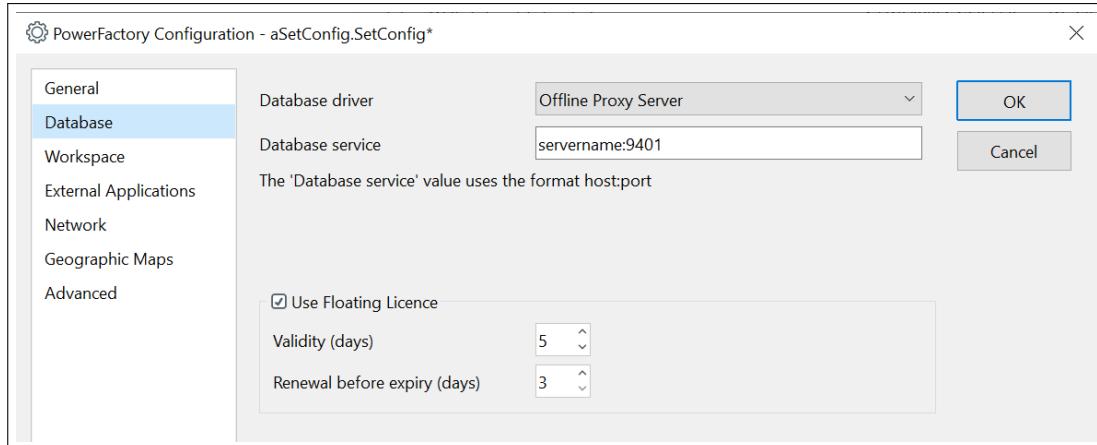


Figure 8.2.1: Log-on dialog, *Database* page

---

**Note:** Using a floating licence with the offline mode allows working with *PowerFactory* without connection to the licence server. Please note, that the usage of floating licences has to be included in the network licence and activated in the user settings.

---

- Press **OK**
- If the usage of a floating licence is configured, *PowerFactory* will generate the floating licence and adapt the licence settings. *PowerFactory* has to be started again afterwards.
- An information dialog appears, saying “Offline database isn’t initialised yet. The initialisation process may take several minutes”.
- Press **OK**
- Following initialisation, the usual *PowerFactory* application window is shown.

### 8.2.1.2 Release Offline Session

- From the main menu, select *File* → *Offline* → *Terminate Offline session*
- A warning message is shown to confirm the synchronisation
- Press **Yes**
- All unsynchronised local changes will then be transferred to the server and the local offline database is removed.
- If a floating licence has been used in offline mode, this licence will be returned to the licence server.

### 8.2.1.3 Synchronise All

Synchronises global data (new users, projects added, projects removed, projects moved) and all subscribed projects.

- Open the main menu *File* → *Offline* → *Synchronise all*

#### 8.2.1.4 Subscribe Project for Reading Only

- Open the Data Manager and navigate to the project.
- Right-click on the project stub. A context menu is shown.
- Select *Subscribe project in offline mode for reading only*.

The project will then be retrieved from the *Offline Proxy Server* and stored in the local *Offline DB cache*.

#### 8.2.1.5 Subscribe Project for Reading and Writing

Write access to the project is required.

- Open the Data Manager and navigate to the project.
- Right-click on the project stub. A context menu is shown.
- Select *Subscribe project in Offline mode for reading and writing*.

#### 8.2.1.6 Unsubscribe Project

- Open the Data Manager and navigate to the project.
- Right-click on the project. A context menu is shown.
- Select *Unsubscribe project in Offline mode*.

#### 8.2.1.7 Add a New Project

A new project is created in offline mode. It is available only in this offline session. Later this project should be published to other users and synchronised to the online database.

- Create a new project or import a PFD project file.
- Open the Data Manager and navigate to the project.
- Right-click on the project stub. A context menu is shown.
- Select *Subscribe project in Offline mode for reading and writing*.

#### 8.2.1.8 Synchronise Project

Synchronises a subscribed project. If the project is subscribed for reading only, the local project will be updated from the online database. If the project is subscribed for reading and writing, the changes from the local offline database will be transferred to the online database.

- Open the Data Manager and navigate to the project
- Right-click on the project stub. A context menu is shown.
- Select *Synchronise*

## 8.2.2 Functionality in Online Mode

### 8.2.2.1 Show Current Online/Offline Sessions

The session status for each user is shown in the Data Manager. Users who are working online appear like this , and those working offline like this .

## 8.2.3 Terminate Offline Session

There may occasionally be cases which require that an offline session be terminated by the Administrator; e.g. if the computer on which the offline session was initialised has been damaged and can no longer be used, and the user wants to start a new offline session on a different computer.

The Administrator is able terminate a session as follows:

- Right-click on the user; the context menu is shown.
- Select *Terminate session*
- A warning message is shown to confirm the synchronisation.
- Press **Yes**

# **Part III**

# **Handling**

# Chapter 9

## Basic Project Definition

The basic database structure in *PowerFactory* and the data model used to define and study a power system is explained in Chapter 4 (*PowerFactory* Overview). It is recommended that users become familiar with this chapter before commencing project definition and analysis in *PowerFactory*. This chapter describes how to define and configure projects, and how to create grids.

### 9.1 Defining and Configuring a Project

There are three methods to create a new project. Two of them employ the Data Manager window and the third employs the main menu. Whichever method is used the end result will be the same: a new project in the database.

#### Method 1: Using the main menu:

- On the main menu choose *File* → *New* → *Project*.
- Enter the name of the project. Make sure that the *Target Folder* is set to the folder in which the project should be created. By default it is set to the active user account folder.
- Press **Execute**.

#### Method 2: Using the element selection dialog from the Data Manager:

- In the Data Manager click on the *New Object* button (- In the *Filter as you type* field, type project or *IntPrj*. Select  Project.
- Press **Ok**. The project folder dialog will then open. Press **Ok**.

#### Method 3: Directly via the Data Manager:

- Locate the active user in the left-hand pane of the Data Manager.
- Place the cursor on the active user's icon or a folder within the active user account and right-click.
- From the context menu choose *New* → *Project*.... Press **Ok**. The project folder dialog will then open. Press **Ok**.

---

**Note:** The *ComNew* command is used to create objects of several classes. To create a new project it must be ensured that the *Project* option is selected.

---

In order to define and analyse a power system, a project must contain at least one grid and one study case. After the new project is created (by any of the methods described), a new study case is automatically created and activated. A dialog used to specify the name and nominal frequency of a new, automatically-created grid pops up. When the button **OK** is pressed in the grid dialog:

- The new grid folder is created in the newly-created project folder.
- An empty single line diagram associated with the grid is opened.

The newly-created project has the default folder structure shown in Figure 9.1.1. Although a grid folder and a study case are enough to define a system and perform calculations, the new project may be expanded by creating library folders, extra grids, Variations, Operation Scenarios, Operational Data objects, extra Study Cases, graphic windows, etc.

Projects can be deleted by right-clicking on the project name in the Data Manager and selecting *Delete* from the context menu. Only inactive projects can be deleted.

---

**Note:** The default structure of the project folder is arranged to take advantage of the data model structure and the user is therefore advised to adhere to it. Experienced users may prefer to create, within certain limits, their own project structure for specific advanced studies.

The default structure of the project, can be modified from the Administrator account. The default structure is defined in a project held the folder: *System → Configuration→ Default*.

---

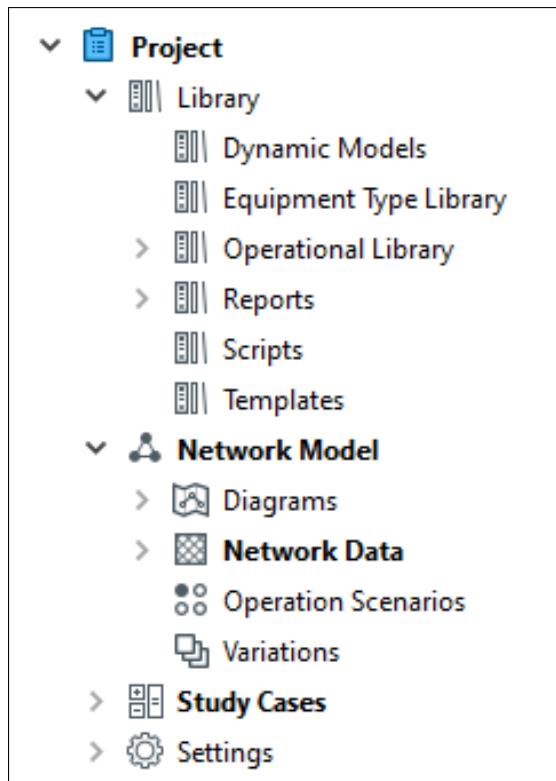


Figure 9.1.1: Default project structure

### 9.1.1 Project Directory

Whenever a project is created, an area outside *PowerFactory*, called the project directory, is also created. By default, this is created as a folder within the user's workspace, named using the object ID of the project. See Section 5.4 to find out more about the workspace.

The project directory is the default place for calculation results to be stored, and is also used as the default location for external files, for example for parameter characteristics.

---

**Note:** In earlier versions of *PowerFactory*, there was no such standard location for external files, so the user had the option to define an external data location `$(ExtDataDir)`). Now, the project directory becomes the standard external data location, although users still have the option to reference files in other locations.

---

### 9.1.2 Project Dialog

The project (*IntPrj*) dialog can be accessed by selecting *Edit* → *Project Data* → *Project...* on the main menu or by right-clicking the project folder in the Data Manager and selecting *Edit* from the context menu.

The *Basic Data* page contains basic project settings:

- Pressing the  button in the *Project Settings* field opens the Project Settings dialog (*SetProj*). See Section 9.1.3 for more information regarding the settings of the project.
- When a project is created, its settings (i.e. result box definitions, report definitions, flexible page selectors, etc.) are defined by the default settings from the system library. If these settings are changed, the changes are stored in the folder “Settings” of the project. The settings from another project or the original (default) ones can be taken by using the buttons **Take from existing project** or **Set to default**. The settings can only be changed when a project is inactive.
- The *Project directory* panel shows the filepath of the project directory, together with some options:
  - The deletion behaviour options determine what happens to the project directory when a project is deleted (and removed from the recycle bin).
  - The **Open...** button will open the project directory in Windows.
  - The **Consolidate** button is useful if a project uses external files in locations other than the project directory. It is used to create copies of all these files in the project directory. A subfolder in the project directory will be created for each folder where such files are found.
  - **Relocate...** allows the user to choose an alternative place for the project directory.
- Pressing the **Contents** button on the dialog will open a new data browser displaying the main folders included in the current project directory.

The *Sharing* page of the dialog allows the definition of the project sharing rules. These rules are particularly useful when working in a multi-user database environment. Further information is given in Chapter 21 (Data Management).

The *Derived Project* page provides information if the project is derived from a master project. The list of derived project can also be printed in the output window using the context menu: *right-click* → *Show Derived Projects* → . Further information is given in Chapter 21 (Data Management).

The *Combined Project* panel enables the user to combine additional projects with the current project if it is active. See Section 21.7.1.3 for more details.

The *Storage* page provides information about the records stored in the project. A *PowerFactory* project contains records of changes to data, which makes it possible to roll back the project to an earlier state using versions (see Section 21.2). However, as the user works with the project and makes changes to it, the number of records increases and it is useful to remove older, unwanted records in a process known as “purging”. By default all changes within the last 7 days will be retained. The context menu, accessible by right-clicking on the project, provides the additional options related to the storage:

- Update Storage Statistics: the data stored in the project is checked and updated. It helps to identify “purgable objects”, i.e. objects that are no longer used in the projects and whose last modification is older than the retention period.
- Purge Store: delete the “purgable objects”.
- Purge Object Keys: each object in the project has an internal key that makes it linkable to external objects. This command deletes this internal key.

The *Migration* page provides information about the migration status of the project. The significance of the information displayed on the *Migration* page is described as follows:

- **Migration status:** when the entire database is migrated, the projects are migrated “on-the-fly”, that is, they are migrated only as needed, and it may be that only some parts of the project are migrated. This status flag indicates to the user whether the project has been partially or fully migrated to the relevant *PowerFactory* major version. The objects that still need to be migrated can be printed in the output window using the context menu, accessible by right-clicking on the project, *Storage* → *Show Migration Statistic*.
- **Project ID, Object ID and Timestamp ID:** these IDs represent the respective database IDs and would be relevant for the user while raising a Support request for Migration related queries.
- **Migrated to Build:** this represents the last build ID of the database currently being used to which the project is migrated. This information can be provided while raising Migration related Support queries.
- **Migrate Variations and Operation Scenarios folder:** applying this option will update the project’s folder structure to match the latest default structure. The basic project structure was modified with *PowerFactory* version 2020 so that the *Variations* and *Operation Scenarios* folders are now situated directly under the *Network Model* folder. Existing projects are not migrated to the new structure by default when databases are migrated to the new structure, as this could cause problems with user-defined scripts. Instead, a dedicated option to “Migrate Variations and Operational Scenario folder” is provided on the *Migration* page in order for the user to manually adapt the project structure.
- **Down Migration of Plot Pages:** since *PowerFactory* 2021, a new plot framework has been introduced. However, if there is a requirement to export a project consisting of plots based on the new framework and to subsequently import it in a *PowerFactory* version which only supports the old plot framework, then the user can choose to migrate down the plot pages to the old framework using **Migrate down** before exporting the project. This operation will create additional plot pages in the project based on the old framework. Once the project has been exported then there is a possibility for the user to delete these additionally created plot pages using the **Purge down migrated pages** option.

The *Miscellaneous* page contains the following:

- The *Tags* field is a free-text field, which can be used to add user-specific labels to the project.
- The button **Calculate**, in the *Licence Relevant Nodes* field, calculates the number of nodes relevant to the *PowerFactory* licence; this number is the number of equipotential nodes in the network.

The *Description* page is used to add user comments and the approval status.

### 9.1.3 Project Settings

The project settings dialog (*SetPrj*) can be accessed by selecting *Edit* → *Project Data* → *Project Settings...* on the main menu or by pressing the → button in the *Project Settings* field of the project’s dialog. The pages of the dialog are described in the following sections.

### 9.1.3.1 Validity Period

*PowerFactory* projects may span a period of months or even years, taking into account network expansions, planned outages and other system events. The period of validity of a project specifies the time span the network model is valid for.

The validity period is defined by the *Start Time* and *End Time* of the project. The study case has study time, which must fall inside the validity period of the project.

**Start Time:** Start of validity period.

**End Time:** End of validity period.

**Status:** This flag enables the user to label a project as Draft or Issued. It does not have any effect on functionality.

### 9.1.3.2 Formats and Units

On this page, units for the data entry and display of variables can be selected.

#### Input Variables

- **Units:** this parameter is used to change the system of measurement used when displaying element parameters. By default, results are entered and displayed in metric (SI) units. British Imperial units can be used instead, by selecting one of the options English-Transmission or English-Industry. The Transmission option uses larger units where appropriate (i.e. miles rather than feet for line length).
- **Lines/Cables Length unit, m:** for metric length units, this parameter allows the user to select the preferred prefix (such as k to use kilometers rather than meters).
- **Loads/Asyn. Machines P,Q, S unit VA, W, var:** this parameter allows the user to select the preferred prefix (such as M to use megawatts rather than watts).
- **Static Generators/Synchr. Machines P,Q, S unit VA, W, var:** this parameter allows the user to select the preferred prefix (such as M to use megawatts rather than watts).
- **Currency Unit:** for displaying values relevant for cost-related calculations, this parameter allows selection from a range of currency units (abbreviated in accordance with ISO 4217). In case none of the units from the list corresponds to the one requested, it is possible to write the text directly in the field (up to 6 characters).
- **Default frequency:** the default frequency can be configured in this field. All new objects that use a nominal frequency are then initialised accordingly with the default nominal frequency. (Existing objects are not affected by any change to this setting).

#### Output Variables

Use the drop-down menus to adjust the preferred prefix for the output variables. It is possible to define different settings for *Load Flow and Simulation* and for *Short-Circuit* calculation types.

The number of decimals can be defined independently for each of the following output variable:

- Voltage
- Current
- Power
- Percent (%)

- Degree
- Per unit (p.u.)
- Other units

### 9.1.3.3 Calculation Options

On the *Calculation Options* page, additional parameters used during the calculation are defined. The following options are available:

#### General

- **Base Apparent Power:** this is the value used during the calculation; the base power of each element is defined by its rated value.
- **HV voltage level:** this describes the voltage threshold for network elements to be considered as high voltage elements. This information is used in various commands when defining acceptable load flow errors.
- **MV voltage level:** this describes the voltage threshold for network elements to be considered as medium voltage elements. The upper range is defined by the threshold given in Min. voltage for HV voltage level. This information is used in various commands when defining acceptable load flow errors.
- **Min. Resistance, Min. Conductance:** the minimum resistance and conductance that will be assigned to the elements if none is defined.
- **Threshold Impedance for Z-model:** this parameter is used to control the modelling of currents (Y- and Z-models are used internally and this parameter controls the small-impedance threshold for switching from one to the other). This control is designed to enhance the robustness of the algorithm and the user is recommended to leave the parameter at its default setting.

The **Set to default** button can be used to restore all the values to the default ones.

#### Topology

- **Settings for slack assignment:** this option only influences the automatic slack assignment (e.g. if no machine, or more than one machine, is marked as “Reference Machine”)
  - **Auto slack assignment:**
    - \* **Method 1:** all synchronous machines can be selected as slack (reference machine);
    - \* **Method 2:** a synchronous machine is not automatically selected as slack if, for that machine, the option on its *Load Flow* page: *Spinning if circuit-breaker is open* is disabled.
    - \* **Off:** auto slack assignment is switched off; the grid will be considered as de-energised if no reference machine is defined.
  - **Priority for Reference Machine:** the criteria used for automatic selection of a reference machine are described under the Load Flow options, in Section 25.3.2. However, the way in which the machines are prioritised for selection can be influenced using this setting. The options are:
    - \* **Rated Power:** Snom is used as the criterion, as described in page 509
    - \* **Active Power Capability:** Pmax-Pmin is used instead of Snom
    - \* **Active Power Reserve:** Pmax-Pgini is used instead of Snom
  - **Auto slack assignment for areas without connection to fictitious border grid:** for large network models covering multiple systems connected by so-called “fictitious border grids”, this option can be used in conjunction with a *Fictitious border grid* flag on the *ElmNet* object, to have more control over which areas will be considered in the calculation, so that remote parts of the network that are not of interest for a particular study can be excluded. The default is for this option to be active.

- **Automatic Out of Service Detection:** when calculations are executed, if the *Automatic Out of Service Detection* parameter is selected, the calculation will treat de-energised elements as though they have been made Out of Service (using the Out of Service flag). This means that they will not be considered in the calculations and no results will be displayed for them. It should be noted that this parameter does not affect elements which are isolated only by open circuit breakers (as opposed to other switch types). These are retained in the calculation because they could be energised via switch close actions at some point.
- **Determination of supplying transformers:** for a distribution network, this flag alters the way in which calculations determine which elements are supplied by which substations. If only voltage controlling transformers are considered, step-up transformers will not be considered as supplying transformers. This affects some calculations such as reliability analysis and the colouring mode *Topology, Supplied by Substation*.

The **Set to default** button can be used to restore all the values to the default ones.

### Lines/Branches

- **Calculation of symmetrical components for untransposed lines:** the selection of one of these methods defines how the sequence components of lines in *PowerFactory* will be calculated:
  - **Method 1:** apply the 012-transformation (irrespective of line transposition). This is the standard method used;
  - **Method 2:** first calculate a symmetrical transposition for untransposed lines, and then apply the 012-transformation.
- **Earth wire reduction of towers:** when overhead lines are modelled using towers, the earth-wires are reduced before matrix transposition is carried out. In older versions the matrix was transposed first, so this setting allows users to use the older method if this is required for compatibility reasons.
- **Consider line compensation current for line loading calculation:** If selected, the total line current (sum of the line current plus the line compensation current) is considered when calculating the line loading. Otherwise, only the line current (without the line compensation current) is used to calculate the line loading.

The **Set to default** button can be used to restore all the values to the default ones.

### Result file

For many calculations, the results are recorded in a result file, which by default uses a proprietary binary format. The *PowerFactory* result object (*ElmRes*) refers to this result file, which is typically stored in the workspace directory.

On this tab, the user can opt to have result files written instead in an open database format (SQLite or ODBC). This allows post-processing based on direct file access to be more easily implemented. Details of the internal layout of the database-based result files can be found in Section 19.7.2.

If ODBC is selected, the user must also select an ODBC Database Configuration object (\*.SetDatabase), using Select... from the drop-down menu. If the ODBC Database Configuration has not already been prepared, the new object icon  can be used to create and configure it.

#### 9.1.3.4 Graphic

On the *Graphic* page, several parameters relevant for the graphical interface can be adjusted. The following options are available:

## General

- **When connecting component to busbar create**
  - **Circuit-Breaker:** when a component is connected to a busbar in a single line graphic, a switch (class *StaSwitch*) is automatically created. By default this switch will be a circuit breaker.
  - **Switch type depending on nominal voltage:** if this option is selected instead, there is the option to have different switch types created for two voltage levels (e.g. low and high voltage systems), with the threshold specified by the user.
- **Variations**
  - **Show inactive elements from other variations:** by default, inactive elements (for example elements created in a expansion stage which is not yet active) are displayed on network diagrams if they are not in freeze mode. They are shown in the colour selected on the first option of the colouring scheme, Energising Status, even if that option is deselected. If the *Show inactive elements from other variations* project setting is deselected, inactive elements will not be visible.
- **Insertion of new substations**
  - **Insert substations with bays:** when new substations are created, it is possible to include Bay elements (*ElmBay*). These group together the network elements that normally constitute a standard bay connection of a circuit to a busbar within the substation. This grouping is useful for visualisation but is also used by the Load Flow Calculation option *Calculate max. current at busbars*: see Section 25.3.3.

## Text Boxes

Customise fonts used for Single Line and Block diagrams. Text boxes used for *Labels* and *Results*

- **Default fonts for Single Line Diagrams:** to customise fonts for *Labels*, *Results* and *Title and Legends*.
- **Default fonts for Block Diagrams:** to customise fonts for *Blocks/slots*, *Signals* and *Title and Legends*.
- **Show jump-to labels at graphically half-connected lines:** to show additional labels for connections going from one diagram into another diagram.

## Plots

- **Use legacy protection plots:** to enable use of the older plot framework.
- **Default colour palette:** to select a different colour palette than the default provided by *PowerFactory*.
- **Default plot style:** to select a different plot style than the default provided by *PowerFactory*.
- **Names in plot legends:** for element names shown in plot legends, the user has the option to include additional information to show where the element is located, e.g. site or substation (short name).

## Geographic

- **Coordinate system:** the setting determines in which coordinate space the geographic coordinates of net elements in the project are stored/interpreted. The user can select from a range of coordinate systems (identified by their EPSG codes) or define their own customised system.
- **WGS-84 bounds:** these fields show the bounds of the selected system in WGS-84 coordinates. They can be modified if the selected option is *custom*.

### 9.1.3.5 Miscellaneous

#### Display name

This option modifies the way the names of the elements (attribute `e:cDisplayName`) are displayed, it affects graphics and reports. The options are:

- **Show element name only:** only the name of the element will be displayed, e.g. “BB” for a busbar within a substation and site.
- **Prepend parent element:** the name of the parent element will be displayed in front of the element name, e.g. “Substation/BB” for a busbar within a substation and site. This is the default option.
- **Prepend site and/or substation name:** it can be decided if the name of the site and/or the substation should be displayed in front of the name of the element, e.g. “Site/Substation/BB” for a busbar within a substation and site.

#### Switching Actions

- **Isolate with earthing:** if this option is selected, when a planned outage is applied, the equipment will not only be isolated but earths will be applied, for example on the terminals at either end of a line. Similarly, when using the *right-click → Isolate (with earthing)* option, earths will be applied.
- **Isolate opens circuit-breakers only:** when equipment is isolated using a planned outage, this option determines whether it is switched out using just circuit breakers or whether isolators adjacent to the breakers are also opened.

#### Planned Outages

- **Consider automatically upon study case activation:** if this option is selected, relevant *IntPlannedout* outages are automatically applied when a study case is activated.
- **Creation of Planned Outages:** outages can be represented using *IntPlannedout* objects (see Chapter 43), and the default is that any new planned outage created will be of this class. If the user wishes instead to create the older *IntOutage* objects, this project setting should be changed to “Create *IntOutage* (obsolete)”.
- **Application sequence of Planned Outages:** when users apply planned outages (*IntPlannedout*), the outcome can depend on the order in which they are applied, and there are default rules based on outage start time and the outage *Priority* flags (see Section 43.3.2 for details). This project setting allows the user to modify those rules. The default setting is “Time-based application (time before priority)”

### 9.1.3.6 Data Verification

#### Nominal Voltage Check

- **Max. allowed difference over Lines/Switches/Fuses:** these are the maximum permitted percentage differences between the nominal voltages at the two ends of a line or the two sides of a switch/fuse (as a percentage of the higher voltage). With differences above these thresholds, a load flow will fail with an error message; for smaller differences (but > 0,5%) a warning is given.
- **Max. allowed deviation from Terminal Voltage, for transformers and for other elements:** there is a check when running a load flow that the rated voltage of a transformer or other element is not too low compared with the nominal voltage of the terminal to which it is connected. With differences above these thresholds, a load flow will fail with an error message; for smaller differences (but > 10%) a warning is given.

### Line couplings

- **Allowable difference in lengths of lines:** this is the maximum permitted percentage difference in the length of those lines which are part of the same line coupling.

#### 9.1.3.7 Substation/Site Types

Includes a list for substation and site types, configurable by the user, which can be used in geographic diagrams to distinguish graphically between different types of substation/sites by assigning different symbols to each type in the list. In a substation itself, the Substation Type is selected from the same list, in the Description page.

#### 9.1.4 Activating and Deactivating Projects

To activate a project use the option *File* → *Activate Project* from the main menu. This shows a tree with all the projects in the current user's account. Select the project that should be activated. Alternatively, a project may be activated by right-clicking on it in the Data Manager and using the context menu.

The last 5 active projects are listed under *File* in the main menu. The currently active project is the first entry in this list. To deactivate the currently active project, select it in the list. Alternatively, you may choose the option *File* → *Deactivate Project* from the main menu. To activate another project, select it in the list of the 5 last active projects.

Upon project activation, the user may see a message about project purging. The purge function is described above in Section 9.1.2, *Storage* page.

---

**Note:** Only one project can be activated at a time.

---

#### 9.1.5 Exporting and Importing Projects

This section describes the import and export of *PowerFactory* projects or data. The import and export of information in other data formats is described in Chapter 24.

Projects (or any folder in the database) can be exported using the \*.pdf (*PowerFactory* Data) file format, or \*.pdfx format. The \*.pdfx format is used when the user wishes to include external files together with the \*.pdf file; the benefit of this is when sharing projects with other people who may not have access to the original files.

The Snapshot Export method, described in Section 9.1.5.2, creates a file in a \*.dzs format.

##### 9.1.5.1 Exporting a Project

To export a project, select *File* → *Export...* → *Data...* from the main menu or click on the  icon of the Data Manager. Alternatively projects can be exported by selecting the option *Export...* on the project context menu (only available for inactive projects).

The export process consists of two or three dialogs, depending on the user selections.

## Initial dialog

In the initial dialog, the user selects the location for the export and the type of export:

- \*.pdf is used when just the project itself is to be exported.
- \*.pdfx is used if the project contains references to external files (for example for parameter characteristics) and the user wants to include these in the export. The resulting \*.pdfx file is a zipped file that contains both the project and the referenced data files.

## Main export dialog

Once the user has clicked on Save, the main export dialog is presented. This has three pages.

### Basic Options

- **Objects to export:** The top-level object(s) to be exported is shown. These are normally projects but could be a folders containing projects, or entire user accounts.
- **Export current state:** this option is visible if the project (or, object in the *Objects to export* table) has Versions defined. If enabled (default), the current state of the project will be exported. Otherwise, only the state of the selected Version/s will be exported.
- **Versions to export:** this table shows all Versions of the *Objects to export*, if any are available. By disabling the checkbox for specific Versions, the user can define which Version should or should not be exported. For base projects, the corresponding Version for the derived project must be selected. See Section [21.3.1](#) for further details.

### Project Files

This page is only relevant if the project makes use of external data files.

- **Export project files:** This setting is by default checked if an export to \*.pdfx is to be made.
- **Files in project directory:** This panel shows all referenced files found in the project directory (as opposed to referenced files stored elsewhere).
- **Ask for referenced files before adding them to the archive:** If this is selected, the user will be presented, after clicking on Execute, with a further dialog to confirm which data files are to be included.

Note that the *PowerFactory* Administrator account has the capability to block the import or export of certain file types via security settings.

### Advanced

On this page, the selected file path and name will be shown, together with some further options.

- **Export data in retention period:** if enabled, data changes from within the retention period will be exported. See Section [9.1.2](#) for further details.
- **Export 'Modified by':** if enabled, the information who last changed an object is exported (attribute 'modified by'). This information could conflict with data privacy rules and is therefore configurable.
- **Export external data files (e.g. results files):** if enabled, calculation results (i.e. results files, plot data, etc.) will be exported. Otherwise, the calculation must be repeated after importing.
- **Export derived project as regular project:** this option is only available for derived projects; see Section [21.3.1](#). If enabled, a derived project will be exported as a full project, meaning that it will contain all the data required from the base project and will no longer need the base project, or retain any link to it.
- **Export to former PowerFactory version:** if the project is intended to be imported into a former *PowerFactory* version, this flag must be activated and the version specified. The button

**Check compatibility for target version** can be used to check for possible problems when exporting to the selected version

### File selection dialog

This last dialog will only appear for a \*.pfpx export, where external files are found, and also only if the option “Ask for referenced files before adding them to the archive” was selected in the previous step. All the files to be exported, whether from the project directory or elsewhere, are listed, and the user may deselect any that are not required. Clicking on OK completes the process.

### The structure of a \*.pfpx file

As mentioned above, a \*.pfpx file is in a zip archive format, containing a *PowerFactory* project and any associated external files that have been included in the file export. The contents of the \*.pfpx file are:

- A metadata \*.json file, which provides internal information needed for correctly managing file paths etc. for the export and subsequent import.
- The *PowerFactory* project.
- A folder with the same name as the project directory, containing:
  - Any files referenced by the project that were stored in the project directory.
  - Additional folders created to store referenced files that were held outside the project directory.

This zip archive format means that the contents can be viewed by the user and can easily be examined by virus scanners or similar tools.

#### 9.1.5.2 Snapshot Export

The Snapshot Export function enables the currently active status of a project to be exported, such that only the relevant objects are included. A project exported in this way is potentially a much smaller file, which nevertheless when reimported into *PowerFactory* can be used to reproduce analysis carried out in the original project study case.

Unlike the existing Project Export, where the project must first be deactivated, the Snapshot Export is performed on an active project. This way, *PowerFactory* can determine exactly which objects are active and which data are applicable as a result of an active scenario or active variations.

To carry out a snapshot export from a project, the required study case and scenario (if used) should be activated. Then *File → Export → Project Snapshot (\*.dzs)...* from the main menu is selected.

When the Snapshot Export is executed, the resulting file outside *PowerFactory* has the file extension .dzs. It can be imported just like a \*.pfpx file and when activated can be used to perform the usual calculations such as load flow or simulations. Furthermore, it is possible for merge processes to be carried out between it and the source project, for example if there is a need to include additional data from the source project.

The Snapshot Export captures only the data required to reproduce the results of the active study case. Therefore the following objects, for example, will not appear in the resultant project:

- **Variations:** changes in active variation stages are consolidated. The exported project will contain therefore no variations.
- **Inactive study cases, scenarios and grids:** inactive study cases, scenarios and grids are not exported. The exported project will have one study case, and no scenarios; if a scenario had been active in the source project, the data will be represented in the network data.
- **Unused library objects:** only objects which are in use are exported, so unused information such as type data which are not referenced will not be exported.

- **Characteristics:** the parameters which are modified by Characteristics will be set at the values determined by the Characteristics, but the Characteristics themselves will not be exported.
- **Operational Library:** operational data such as Thermal Ratings, which may contain variations, will be reduced to just the currently active values.

### 9.1.5.3 Importing a Project

The process for importing a project is the same whether the project is stored as a \*.pf file or a \*.pfdf file. However, the latter will contain additional files; see the paragraph called **Project Files page** below.

There are several options for importing projects:

- By selecting *File → Import → Data...* from the main menu.
- By clicking on the  icon in the Data Manager.
- By selecting *Import...* on the project context menu (only available for inactive projects).
- *Drag and Drop* the \*.pf file onto the *PowerFactory* GUI.

The user can select the file type to be imported from the drop-down menu in the Open dialog which opens. Only when using the *Drag & Drop* option for the import, the type of file is automatically selected by *PowerFactory*.

#### Basic Options page

The Basic Options page has two parts:

The *File information* panel contains information about the file to be imported, including:

- **Contents:** list of all the projects included in the imported file.
- **Date and Time:** information about when the file was exported.
- **Exported by:** user that exported the file.
- **Target version:** the target version set when the project/s was exported.
- **Source version:** the *PowerFactory* version from which the project/s was exported.
- **Number of projects:** the number of projects included in the file.
- **Number of records:** number of data stored in the project. The number and type of records in a project are shown in the *Storage* page of the project's edit dialog.
- **Original Path:** the original location from where the file was exported.
- **Missing or unreadable referenced object:** if there are missing references, they are listed in this field.

In the *Import options* panel in the PFD import dialog contains the following options:

- **New Path:** by default, the project will be imported in the current location in the Data Manager. In this field, a different path can be specified.
- **Text encoding:** specifies the code used in the project. It should be only modified if the project code does not align with the coding of the computer, e.g. Cyrillic project with English MS Windows settings (Latin code).
- **Activate project after import:** automatically activates the project after importing it.

### Project Files page

This page is only relevant for the import of \*.pfmx files.

If a project has been exported as a \*.pfmx file, this means that external data files have probably been included in addition to the project itself. This process is described in Section 9.1.5.1 above. When the \*.pfmx file is imported, all the external files that come with it are stored in the new project directory. The folder structure for external files within the \*.pfmx file is reflected within the new project directory. Furthermore, all the references to external files within the newly imported project are automatically adjusted to their new location.

On this page, the user can see the external files that are to be imported, and modify the selection via check-boxes. Note that certain file types can be blocked by the Administrator (see Section 6.5.5). In addition, files of certain types (\*.dll, \*.fmu and \*.pfmu) are deselected by default. The user may be able to select these deselected files for import, but will see a warning about the risks of doing so.

### Advanced page

There are two panels on this page. In the first panel, missing external references are listed. The second provides additional information for derived projects, namely the base project and version from which the project was derived. See Section 21.3.1 for information about base and derived projects.

## 9.1.6 External References

In order to avoid problems when exporting/importing projects, it is recommended to check for external references before exporting the project. This can be done via the project context menu, selecting the *External References* sub-menu and then the option *Check for External References*. The user can then select the External Locations, such as the *DIGSILENT Library* or the “Configuration” folder. After pressing **OK**, a list of external project references is displayed in the output window.

If external references are found, these can be packed before the project is exported. This can be done through the context menu of the project, by accessing the *External References* sub-menu and selecting the option *Pack External References*. Similar to the option described above, the user can select the External Locations. By pressing **OK**, a new folder in the project structure called “External” is created, which contains all formerly external objects.

---

**Note:** The options described above are only selectable after the project has been deactivated.

---

To review the use of type objects, standard models, etc. from the *DIGSILENT Library* (and other external libraries) in a particular project, the option *Show External Types* from the *External References* sub-menu can be used. This generates a report in the output window, containing those types that are not stored inside the project library. Objects from external libraries can be opened up from the list in the output window. In addition, for each item listed, the number of objects using it is given. This acts as a hyperlink which can be used to bring up a list of objects in the project referencing this item.

As explained in Section 14.2.1, all the main objects in the global libraries are managed using versions. Via the option *Show Updates for External Types* from the *External References* sub-menu, the user can check whether later versions are available, and see what has been changed in the later versions.

## 9.1.7 Including Additional Documents

It may be useful sometimes to provide additional information such as a detailed documentation about a network element, to the user of a project. Such information can be provided by creating an *IntDocument* object within the project, for example within the project library.

This is done by using the *New Object* button (  ), and selecting *IntDocument* from the list.

In the *IntDocument* object, there is a *Filename* field, which is populated by selecting the file location. This is sufficient to provide access to a document which is outside the *PowerFactory* database, but there is also an *Import* option, which enables the user to import the document itself into the *PowerFactory* database.

Such *IntDocument* objects (whether simply links or containing the actual document) can also be created elsewhere in the database, for example in the Configuration area. Network elements have a field called “Additional Data” on the description page, which is a convenient place to hold a reference to an *IntDocument* object.

## 9.2 Project Overview

The Project Overview gives an overview of the project and allows easy interaction with the project data. It is a dockable window that by default is shown on the left side of the main application window, as can be seen in Figure 9.2.1. It is normally pinned so as to be always visible, but various options are available to the user to change this:

- The pin icon on the title bar can be used to pin/unpin the Project Overview and Drawing Tools; they can be temporarily seen by clicking on their respective tabs.
- Alternatively, these windows can be left pinned but collapsed (using an icon toward the right-hand side of the title bar)
- The Project Overview and/or Drawing Tools can be moved into another tab group or a separate floating group by dragging the tab to the new location.
- The visibility of the Project Overview and/or Drawing Tools can be changed via the Window menu on the main toolbar.

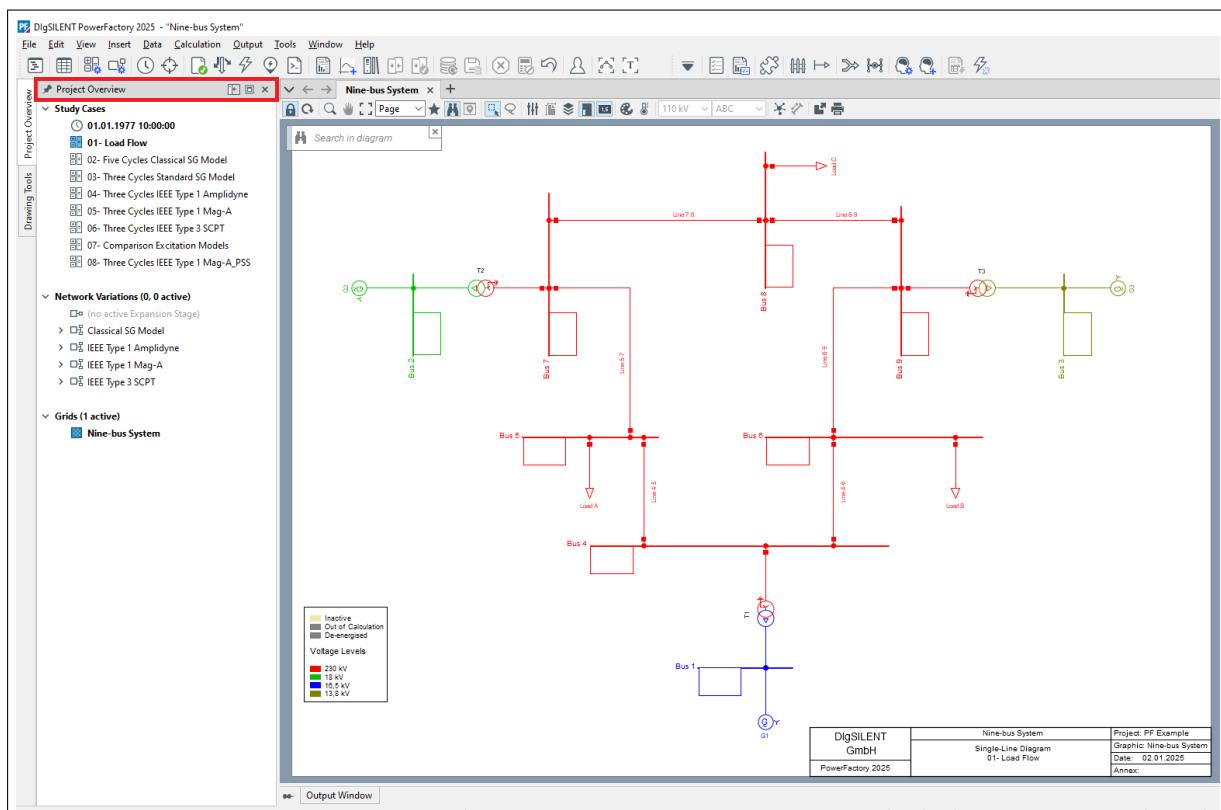


Figure 9.2.1: Project overview

The following objects and information can be accessed via the Project Overview.

- Study Cases
  - Active Study Case
  - Inactive Study Cases
  - Current Study Time
- Operation Scenarios
  - Active Scenario Schedulers
  - Active Scenarios
  - Inactive Scenarios
- Variations
  - Recording Expansion Stage
  - List of active Variations with active and inactive Expansion Stages as children
  - List of inactive Variations with inactive Expansion Stages as children
- Grid/System Stages
  - List of active Grids
  - List of inactive Grids
- Trigger
  - Active triggers

Entries for active objects are displayed with bold text, entries for inactive objects are displayed as disabled/grey.

Modifications to the objects and their status can be done directly in the Project Overview window.

## 9.3 Creating New Grids

When defining a new project, a grid is automatically created. If additional grids are required, various methods may be used to add a grid folder to the current network model:

1. Select *Insert → Grid with Diagram...* on the main menu.
2. Right-click on the Network Data folder (in the active project) in a Data Manager window and select *New → Grid...* from the context menu.
3. On the Project Overview window, right-click on the *Grids* and select *New → Grid...*

The dialog to create a new grid will then appear. There the grid name, the nominal frequency and a grid owner (optional) may be specified. After the **OK** button has been pressed, the new grid is created in the Network Data folder and a reference in the Summary Grid object of the selected active Study Case is created.

As indicated in Chapter 13 (Study Cases), grids can be later added or removed from the active Study Case by right-clicking on the grid and selecting *Activate/Deactivate*.

# Chapter 10

## Network Graphics

### 10.1 Introduction

*PowerFactory* works with three different classes of graphics which constitute the main tools used to design new power systems, controller block diagrams and displays of results:

- Single Line Diagrams (described in this chapter)
- Block Diagrams for User-defined Dynamic Models (UDM) (described in Section [30.2](#) (for High-level Control System Representation), Section [30.6](#) (for DSL Models) and Section [30.10](#) (for Modelica Models))
- Plots (described in Section [19.8](#): Plots)

Diagrams that are displayed are held in the Desktop object; (see Section [10.2.2](#) for more information).

When working with graphics, many hotkeys are available for common actions. These are listed in Section [10.9](#).

### 10.2 Graphic Windows and Database Objects

In the *PowerFactory* graphic windows, graphic objects associated with the active study case are displayed. Those graphics include single line diagrams, station diagrams, block diagrams and plots. Many commands and tools are available to edit and manipulate symbols in the graphics. The underlying data objects may also be accessed and edited from the graphics, and calculation results may be displayed and configured.

Many of the tools and commands are found in the drop down menus or as buttons in the toolbars, but by far the most convenient manner of accessing them is to use the right mouse button to display a “context menu”. The menu presented is determined primarily from the cursor position.

#### 10.2.1 Network Diagrams and other graphics

Four types of graphical pages are used in *PowerFactory*:

1. Single Line Diagrams (schematic and geographical network diagrams) for entering power grid definitions and for showing calculation results.

2. Detailed graphics of sites, substations or branches (similar to network diagrams) for showing busbar (nodes) topologies and calculation results.
3. Block Diagrams for designing dynamic models and relays.
4. Plot Pages for creating plots, e.g. for the results of a time domain simulation.

The icon *Diagrams* (Diagram icon) can be found inside the Data Manager. Grids, substations, branches, sites and dynamic models types (*Composite Model Frame*, *DSL Model Type* and *Modelica Model Type* in *PowerFactory* terminology) each have a graphical page. In order to see the graphic on the screen, open a Data Manager and locate the graphic page object you want to show, click on the icon next to it, right-click and select *Diagrams* → *Show Diagram*. This option is also available directly by right-clicking on the object. The graphic pages of grids and substations are to be found in the subfolder *Diagrams* (Diagram icon) under the *Network Model* folder.

Note that it is also possible to store Diagrams within the Grid, although this is generally not recommended.

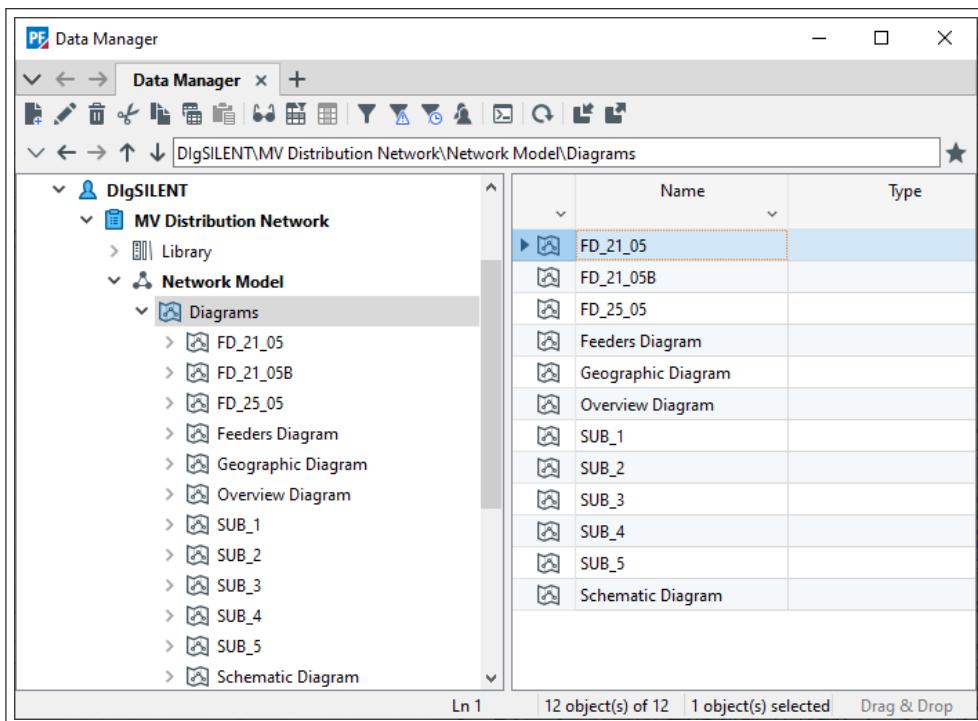


Figure 10.2.1: The Diagrams folder inside the Data Manager

## 10.2.2 Active Graphics, Study Case and Desktop

The graphics that are displayed in an active project are determined by the active study case. The study case has a folder called the *Desktop* (*SetDesktop*), which contains graphic page objects that are linked to the graphics to be displayed. The desktop is automatically created and maintained and should generally not be edited by the user.

Graphic pages and other pages (tabs), such as the Data Manager or reports, can be freely moved from one docked or floating window to another. The organisation of these tabbed windows is done automatically via Tab Group objects (*SetTabgroup*), also held within the desktop.

Consider the project shown in Figure 10.2.2. There are several diagrams in the *Diagrams* folder, but the desktop folder in the study case has a reference to only some of them and thus only these graphics will be shown when the study case is activated.

The page tab, which is normally at the top of the graphic, offers options for handling the graphics in the desktop. See Section 10.2.5 for details.

The study case and desktop folder will also contain references to any other graphics that have been created when the study case is active, such as plot pages.

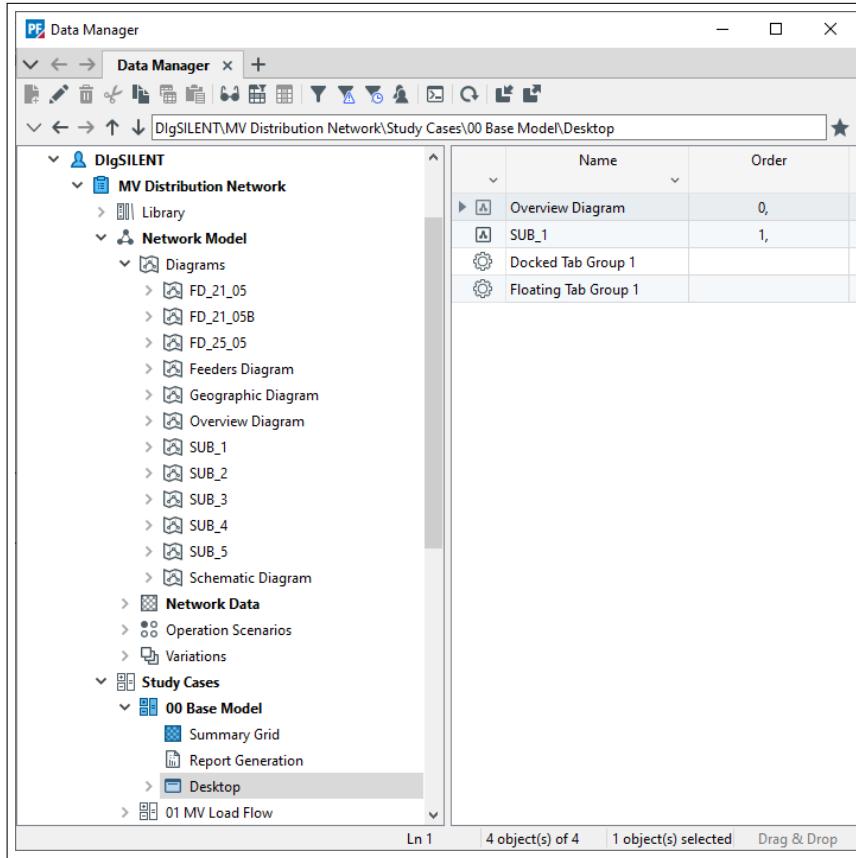


Figure 10.2.2: Relationship between the study case, desktop and single line diagrams

### 10.2.3 Single Line Graphics and Data Objects

When building a new network, it is usually recommended that this is done from a single-line diagram. As each objects is created, the associated graphical representation is created too. For more information about building a network, see Chapter 12.

In a simple network there may be a 1:1 relationship between data objects and their graphical representations, i.e. every load, generator, terminal and line is represented once in the graphics. However, *PowerFactory* provides additional flexibility in this regard. Data objects may be represented graphically on more than one graphic, but only once per graphic. Thus a data object for one terminal can be represented graphically on more than one graphic. All graphical representations contain the link to the same data object.

Furthermore, graphical symbols may be moved without losing the link to the data object they represent. Likewise, data objects may be moved without affecting the graphic.

The graphics themselves are saved in the database tree, by default in the Diagrams folder of the Network Model. This simplifies finding the correct representation of a particular grid, even in the case where there are several graphic representations for one grid.

When the drawing tools are used to place a new component (i.e. a line, transformer, etc.) a new data object is also created in the database tree. A graphic object therefore has a reference to a grid folder.

The new data objects are stored into the 'target' folders that the graphics page are associated with. This information may be determined by using the *Diagram Settings* button (☰).

Since data objects may have more than one graphic representation the deletion of a graphic object should not mean that the data object will also be deleted. Hence the user may choose to delete only the graphical object (*right-click → Delete Graphic Object only*). In this case the user is warned that the data object will not be deleted. This suggests that a user may delete all graphical objects related to a data object, with the data object still residing in the database and being considered for calculations.

When an element is deleted completely (right menu option *Delete*) a warning message will confirm the action. This warning may be switched off in the User Settings dialog, *Data/Network Model Manager* page, *Confirm Delete Action*.

#### 10.2.4 Creating New Graphic Windows

A new graphic window can be created by using *Insert* on the main menu. Two of the options in this menu concern the creation of elements with their respective graphics: *Insert → Grid with Diagram* and *Insert → Dynamic Model → Composite Model Frames/DSL Model Type/Modelica Model Type*. The other three other options allow the insertion of diagrams of existing elements or plots:

- **Schematic Diagram:** creates a schematic diagram and sets the target folder to the active grid. If there is more than one active grid, a dialog to select the target grid will be shown.
- **Geographic Diagram:** creates a geographic diagram of the network.
- **Plot:** presents a dialog box where the user selects the required plot type. Another dialog box enables parameters to be entered. Once these are confirmed, the plot is then displayed.

New diagrams can also be inserted using the *Create new tab* icon (+) in a fixed graphic window or floating group, and selecting the relevant options.

#### 10.2.5 Page Tab

The page tab of the graphic window shows the name of the graphic. By default the tab is at the top of the window, but a user setting (*Window Layout* page) enables the user to change this if required. Another user setting allows the user to show icons on the page tab, to indicate the type of graphic. The order of the page tabs is easily changed by using drag-and-drop.

A number of options are offered when right-clicking on the page tab. Some of these relate to split-screen working and the options offered will depend on the existing split configuration (if any), and also on how many tabs are already open in a group. The options offered to the user also depend on the type of information displayed on the tab, but these are the main options available:

- *Move to Group Above/Below* or *Move to Left/Right Group*: for split-screen working, move to an existing group. (See Section 4.7 for more details)
- *Move Upward/Downward to New Group*: or *Move Leftward/Rightward to New Group*: for split-screen working, start a new group (See Section 4.7 for more details)
- *Move Within Group...:* Allows tabs to be rearranged within the group. Tabs can also simply be dragged to a new position.
- *Move to New Floating Group*: Allows tabs to be moved out of the main graphic window into a floating group.
- *Rename Page...:* displays a dialog to rename the tab.
- *Rename Diagram...:* displays a dialog to rename the diagram.

- *Close Tab*: closes the graphic page, removing it from the desktop in the study case, but not deleting the graphic object itself, if one exists.
- *Delete Page* (for plots) will close the graphic page *and* delete the page from the study case.
- *Delete Diagram* (for single-line diagrams and geographic diagrams) will close the graphic page *and* delete the diagram itself.
- *Duplicate Page*: used for creating copies of plot pages.
- *Convert to Permanent Diagram*: if a new detailed diagram of a substation or site has been created, this option enables the user to save it as a permanent graphic.

The name of the active (i.e. currently-presented) graphic is shown in bold. But the above options are available whether the graphic tab is active or not. Tabs can be multi-selected by using the CTRL key (to select individual tabs) or the Shift key (to select a range of tabs), and the selected tabs will be coloured blue; then appropriate options can be applied to the selected tabs.

### 10.2.6 Tab Group

The tab group offers a drop-down menu consisting of a list of options that allows the opening of diagrams and plots (and other information such as reports). For details of the options available in this menu, please refer to Section [4.7.3](#).

### 10.2.7 Drawing Tools

The Drawing Tools tool-window appears by default on the left-hand side of the graphic window, where it shares the space with the Project Overview. The visibility of the Drawing Tools and Project Overview can be controlled by the user. By default, this area is “pinned”, which means that one or other window is always permanently visible, but the user can use the pin icon to hide them. Even if the windows are pinned, they can be temporarily collapsed via a Collapse Area icon on the toolbar. They can also be moved into a separate floating group.

The tool icons shown in the Drawing Tools window form a tool-box, which is specific to the type of graphic that is currently selected.

See Figure [10.2.3](#) for two examples.

The Drawing Tools are only available for use if the graphic is not in freeze-mode. A graphic may be unlocked from freeze mode using the  icon on the graphic tool bar or the similar icon at the top of the Drawing Toolbox itself.

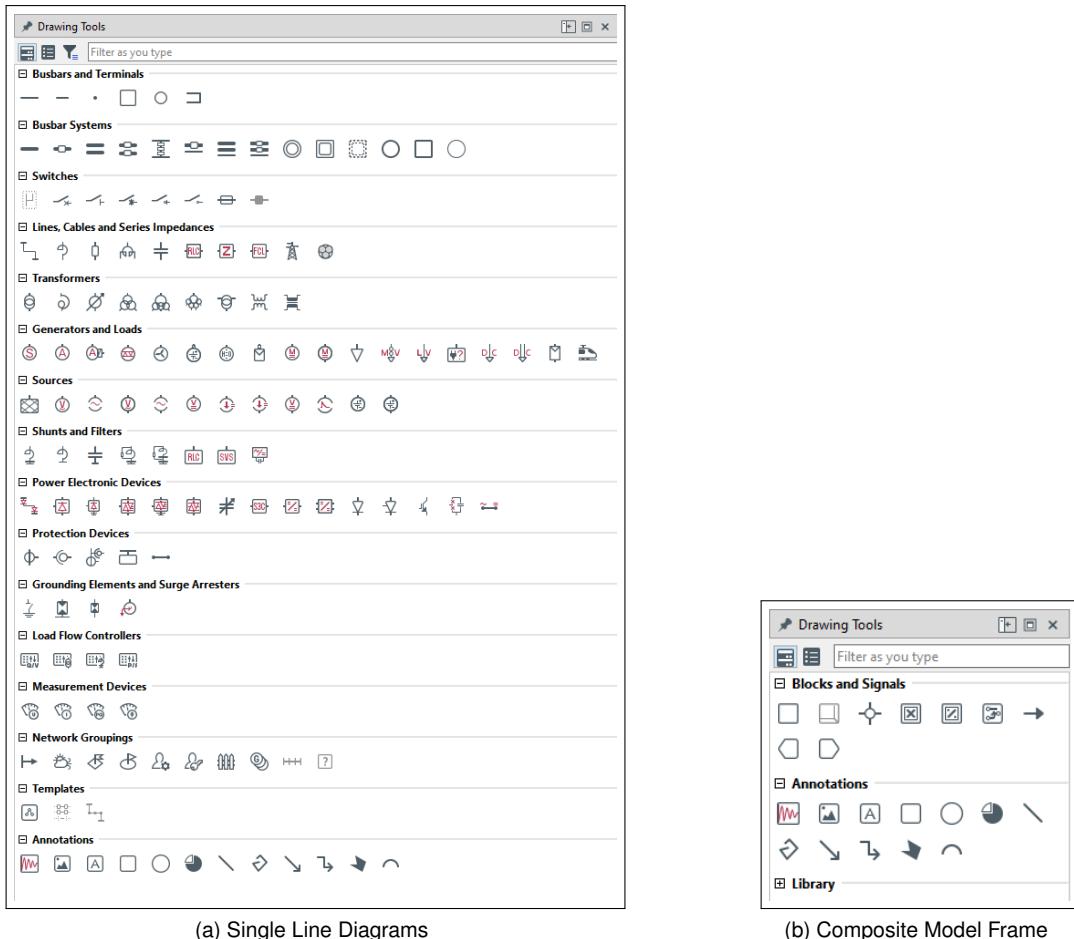


Figure 10.2.3: Drawing Toolbox examples

**Note:** The greyed out objects in the Drawing Tools window can only be used during assisted manual drawing. See Section 12.6.1.3 for more information.

The user can customise the Drawing Tools window in several ways:

- Group headers can be shown or not.
- Element labels can be shown or not.
- A group filter allows the user to show only certain groups. Alternatively, if the group headers are shown, the plus and minus icons can be used to expand or collapse individual groups.
- There is a text filter for filtering by element label. This is case-insensitive and will find any occurrence of the given text string in the element labels, but only in the groups which are currently visible.

In addition, the Drawing Tools tool-window features a dynamically generated category, called *Recently Used*, which contains the most newly used icons. Once produced, it is automatically situated at the top of the Drawing Toolbar tool-box and, as any other grouping category, can be collapsed (minimised) and/or hidden. To avoid cluttering, it only shows at most one full row of icons, i.e. the number of shown icons depends on the width of the Drawing Tools tool-window.

**Note:** An icon is considered as used when it is selected and unselected by the user.

### 10.2.8 Active Grid Folder (Target Folder)

On the status bar of *PowerFactory* (Figure 10.2.4), the active grid folder is displayed on the left-most field, indicating the target folder (grid) that will be modified when changes are made in the network diagram. The active target folder can be changed double-clicking this field and then selecting the desired target folder. This can be useful if the user intends to place new elements on a single line diagram, but have the element stored in a different grid folder in the Data Manager.



Figure 10.2.4: The Status Bar

## 10.3 Graphic Commands and Settings

In this section the commands and settings that are available in *PowerFactory* to configure and use the graphic windows are introduced. The sub-sections of this section are divided as illustrated in Figure 10.3.1. Some of these commands are also available from the main menu under *View*.

Further commands available from the context menus of elements are also listed towards the end of this section (10.3.15).

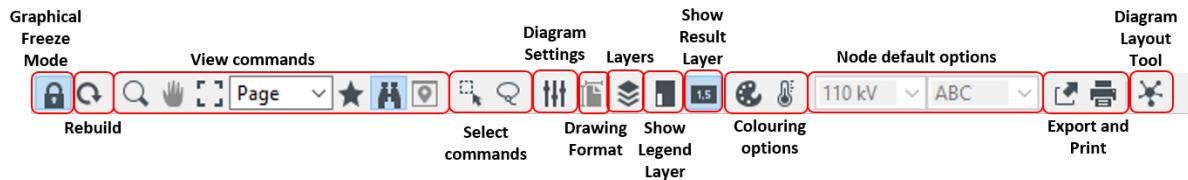


Figure 10.3.1: Graphics Toolbar

### 10.3.1 Freeze Mode

The *Graphical Freeze Mode* button locks the diagram from graphical changes, no network elements can be added or deleted. Note that the status of switches can still be modified when freeze mode is on.

### 10.3.2 Rebuild

The drawing may not be updated correctly under some circumstances. The rebuild function updates the currently visible page by updating the drawing from the database.

### 10.3.3 View commands

#### 10.3.3.1 Zoom In

When pressing the *Zoom In* button, the cursor changes to a magnifying glass. The mouse can then be clicked and dragged to select a rectangular area to be zoomed. When the frame encompasses the area you wish to zoom into release the mouse button.

Alternatively, by pressing **Ctrl** and using the mouse scroll wheel it can be zoomed in and out with the mouse cursor as reference point. Using the **Ctrl+-** and **Ctrl++** keys, zooming is also possible referenced to the centre of the visible area. If in addition **Shift** is pressed, the reference changes to the mouse cursor.

---

**Note:** The Acceleration Factor for zooming and panning can be changed on the Advanced tab of the *Graphic Window* page in the User Settings dialog.

---

### 10.3.3.2 Zoom All

The button *Zoom all*  zooms to the page extends.

### 10.3.3.3 Zoom Level

This button adjusts the zoom level based on a number of presets, or a user-defined zoom level (given in percentage).

### 10.3.3.4 Pan Tool

Use the *Pan tool*  to pan the single line diagram (when not at the page extends). The pan tool is activated with pressed middle mouse button, too. Alternatively, the mouse scroll wheel can be used to scroll vertically, and **Ctrl+→ / ↑ / ← / ↓** used to scroll vertically and horizontally.

### 10.3.3.5 View Bookmarks

The *View Bookmarks*  allows to save the current view and restore that view at a later date. The bookmarks may be used with different network diagrams (single line, geographic, detailed substation graphic) of the same or different grids. In big networks this feature allows to switch very quickly between diagram details to check e.g. the impact of operational changes in loads/feed-in at different places in the network.

By clicking *View → Bookmarks → Add Bookmark...* the name will be asked, under which the current view is stored and displayed in the list of the View Bookmarks. To edit, delete already existing or even create manually new bookmarks, click on *View → Bookmarks → Manage Bookmarks...*. An object browser with all existing bookmarks appears. They can directly be changed using the object browser or by opening the Edit-dialog for single bookmarks. The *IntViewBookmark*-objects contain the reference to the diagram, the position and size of the *View Area*. To further accelerate the workflow Hotkeys are set automatically for the bookmarks, which can be changed, too. If the current view should be assigned to the opened bookmark, the button **« From View** may be pressed.

### 10.3.3.6 Search in Diagram

It is possible to search for network elements within a graphic, using the  icon in the graphic toolbar. The search facility automatically lists possible objects as the user types.

On geographic diagrams (see Section 10.8), it is also possible to search for places such as towns or streets.

If the user wishes to restrict the search to geographic places rather than network elements, the prefix “geo:” can be used.

### 10.3.3.7 Navigation Pane

The *Show Navigation Pane*  icon is used to open or close the navigation pane.

The navigation pane provides the user an overview of the whole network in a small window. It is available for all graphics but plots. When zooming-in on a part of the grid, the navigation pane provides an overview of the whole network and highlights the part of the network that is currently being shown in the diagram.

The frame within the navigation pane can be moved around in order to see different parts of the network.

The navigation pane is enabled for every diagram by default, but can be disabled for specific diagrams. This is done by first clicking on the *Diagram Settings* icon , then the dialog, go to the *Advanced* tab within the *Basic Attributes* and disable the option *Allow Navigation Pane*.

## 10.3.4 Select commands

### 10.3.4.1 Rectangular Selection

The *Rectangular Selection*  icon is used to select a rectangular section of the diagram. This icon is by default pressed, however the *Pan Tool* or *Free-form Selection* may also be used.

### 10.3.4.2 Free-form Selection

The button *Free-form Selection*  is used to select a custom area of the diagram.

## 10.3.5 Diagram Settings

Each graphic window has its own settings, which may be changed using the *Diagram Settings* button . The available settings of the dialog are described in the following sections.

### 10.3.5.1 Basic Attributes page

The *General*-tab offers the following settings:

- **Name:** the name of the graphic
- **Target folder for network elements:** the reference to the database folder in which new power system elements created in this graphic will be stored.
- **Default view area:** when the user has zoomed into part of the graphic, the  icon can be used to zoom out again. By default, this will be to the full area of the graphic, but by selecting a bookmarked area here, as the default view area, this “zoom all” can be customised. This is useful for users of large networks who are only interested in a specific region. (See Section 10.3.3.5 for more information about bookmarks.)
- **Write protected:** if enabled, the single line graphic can not be modified. The drawing toolboxes are not displayed and the *freeze* icon becomes inactive.

- **Line style for cables:** is used to select a line style for all cables.
- **Line style for overhead lines:** is used to select a line style for all overhead lines.
- **Node width factor:** the width of points and lines for nodes and busbars.
- **Offset factor when drawing one-port devices:** defines the length of a connection when a one port device (e.g. load, shunt) is drawn by clicking on the busbar/terminal. This is the default distance from the busbar in grid points.
- **Allow individual line style:** permits the line style to be set for individual lines. The individual style may be set for any line in the graphic by right-clicking on the line and selecting *Graphic Object → Set Individual Line Style*. This may also be performed for a group of selected lines/cables in one action, by first multi selecting the elements.
- **Allow individual line width:** as for the individual line style, but may be used in combination with the “Line Style for Cables/Overhead Lines” option. The individual width is defined by selecting the corresponding option in the right mouse menu (may also be performed for a group of selected lines/cables in one action).
- **Diagram colouring:** if the *Default* option is selected, changes to the active colouring scheme will affect all diagrams. By setting the option to *Colouring scheme*, the scheme of the current diagram can be configured separately. Press **Manage...** to open an object browser with a list of the available colouring scheme settings. Copy the existing one or create a new one and change it to the desired scheme. Close the object browser and select the new colouring scheme from the drop down list.  
When a new diagram colouring scheme is created, there is an additional **Default** button in the edit dialog. Clicking on this button will apply the new scheme to all the diagrams with the *Default* option selected.

The *Advanced*-tab offers the following settings:

- **Allow navigation pane to be shown:** if checked, the *Navigation Pane* can be activated by clicking on the *Show Navigation Pane* button , otherwise the button will be grayed out.
- **Show tooltip on network elements:** if this is selected, information about network elements will be shown if the user hovers the mouse over the element.

The *Coordinates Space* page should generally only be configured with the assistance of *DIGSILENT* support staff. Note that if *Use Scaling Factor for Computation of Distances* is selected on the *Coordinates Space* page, it is possible to calculate the length of lines on the Single Line Diagram by right-clicking and selecting *Measure Length of Lines*. In geographic diagrams, this option is activated by default.

### 10.3.5.2 Schematic Diagram page

When a schematic diagram (overview, single line or detailed) is active, the *Schematic Diagram* page will be available with the following options:

#### Drawing aids

- **Snap elements to grid:** snaps the elements onto the drawing raster.
- **Line routing:** defines the orthogonalisation of the connecting lines between branch elements and terminals, the options are:
  - **Non-orthogonal:** connections will be drawn exactly as their line points were set.
  - **Orthogonal:** allows only right-angle connections between objects.
  - **Semi-orthogonal:** the first segment of a connection that leads away from a busbar or terminal will always be drawn orthogonally. The *Semi-orthogonal offset* defines the length of the first orthogonal segment.

### Size factors for

Defines the size of the symbols in the diagram for sites, substations, edge elements and line end symbols. The *connection circles on simplified substations* is a width factor used in single line diagrams: in single line diagrams multiple busbar substations are only represented by their main busbars. Connected elements may be connected to each of the busbars by changing the state of the internal switches. The currently active connection is shown in the diagram by a filled circle, the not connected ones by hollow circles. The width of the circles is defined in this field.

---

**Note:** The settings for the cursor type for the graphic windows (arrow or tracking cross) may be set in the User Settings dialog, see Section 7.3 (Graphic Windows Settings). This is because the cursor shape is a global setting, valid for all graphic windows, while all graphic settings described above are specific for each graphic window.

---

#### 10.3.5.3 Text Boxes page

The following options are available:

- **Allow individual fonts for text boxes:** self-explanatory.
- **Use fonts from project settings:** this option allows the user to use the font configuration (i.e. font type, style, size, effects) defined in the Project Settings. When activated, a button is available to access the corresponding Project Settings page.
- **Labels**
  - **Nodes:** in case the option *Use fonts from project settings* is not active, the user can customise the font characteristics. The option *Show frame* shows a frame in the text boxes corresponding to node labels.
  - **Branches:** in case the option *Use fonts from project settings* is not active, the user can customise the font characteristics. The option *Show frame* shows a frame in the text boxes corresponding to branch labels.
  - **Background:** specifies the transparency of label boxes:
    - \* **Opaque:** means that objects behind the label box cannot be seen through the label box.
    - \* **Transparent:** means that objects behind the label box can be seen through the label box.
- **Results**
  - **Nodes:** in case the option *Use fonts from project settings* is not active, the user can customise the font characteristics. The option *Show frame* shows a frame in the text boxes corresponding to node results.
  - **Branches:** in case the option *Use fonts from project settings* is not active, the user can customise the font characteristics. The option *Show frame* shows a frame in the text boxes corresponding to branch results.
  - **Background:** specifies the transparency of result boxes:
    - \* **Opaque:** means that objects behind the result box cannot be seen through the result box.
    - \* **Transparent:** means that objects behind the result box can be seen through the result box.
  - **Always show result boxes of detailed couplers:** self-explanatory.
  - **Space saving representation of result boxes on connection lines:** self-explanatory.
- **Title and legends:** in case the option *Use fonts from project settings* is not active, the user can customise the font characteristics corresponding to the title and legends.
- **Show line from general text boxes to referenced objects:** may be disabled to unclutter the graphic.

#### 10.3.5.4 Switches page

- **Switch state symbol at connection end:** selects the switch representation (see Figure 10.3.2):
  - **Permanent box:** shows a solid black square for a closed and an frame line for an open switch (10.3.2a).
  - **Old style switch:** shows the switches as the more conventional switch symbol (10.3.2b).

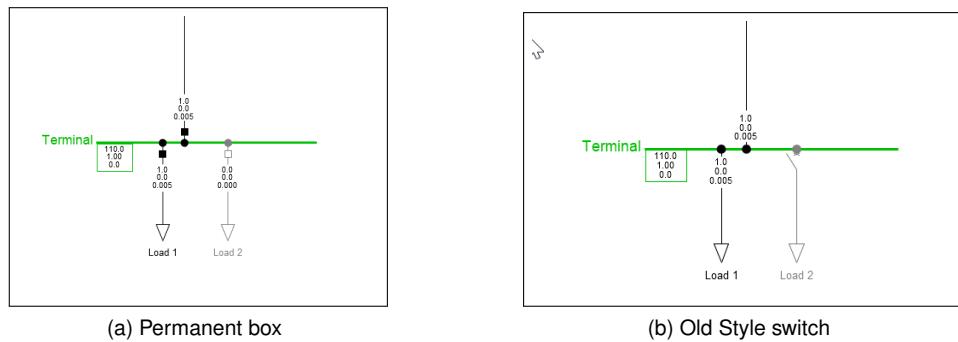


Figure 10.3.2: Different switch symbols

- **Display frame around switches:** draws a frame around the switch itself (breakers, disconnectors, etc.). This only applies to user-drawn breakers and disconnectors.
- **Create switches when connecting to busbar:** self-explanatory.
- **Show connected busbars as small dots in simplified substation representation:** defines how the connection points on busbars are represented in busbar systems.

#### 10.3.5.5 Geographic Diagram page

The settings on this page define the appearance of the graphical representation of network elements in the geographic diagrams. This page is only visible when a geographic diagram is active.

- **Size factors for:** defines the size of the symbols in the diagram for sites, substations, terminals, edge elements, text, line loads and section transitions and line end symbols.
- **Geographic scale threshold for visibility of:** in extensive networks with a high scale level, edge elements (except lines) and switch state boxes at line ends, are hidden at a specified scale level to improve the clarity of the diagram. The threshold for the visibility can be changed in this field.
- **Line width:** sets the width of all the lines in the geographic diagram.
- **Distance factor for one-port devices:** defines the distance of all drawn one-port-devices (e.g. load, shunt) to their connected nodes. This is the default distance from the busbar in grid points.
- **Margin at full zoom:** since in geographic diagrams there is no border, this value defines the margin shown if *Zoom All* [ ] is pressed.
- **Show coordinates in latitude/longitude:** shows the coordinates of the current cursor position in latitude and longitude in the Status Bar. Otherwise the position is displayed as X/Y values representing UTM-coordinates. The border values of the area represented by the diagram are listed in the tab *Coordinate Space* of the *Basic Attributes* page (10.3.5.1).
- **Prefer branch coordinates:** this option affects elements which are grouped to branches (*Elm-Branch*). If the branch itself has geographic coordinates, they will be used in the geographic diagram, otherwise the coordinates of the elements contained in the branch are taken into account.
- **Show Scale:** shows or hides the scale in the diagram.

### Substation Types tab

The settings in this tab are related to the graphical representation of substations in geographic diagrams. The dialog offers the possibility to distinguish graphically different types of substations and improve the clearness of the diagram by adding additional data through the substation symbol. A possible use of this feature can be seen in the geographic diagram 'Overview Diagram' of the Application Example 'MV Distribution Network' (*File → Examples*).

The column 'Substation Type' of the *Assignment Table* is the list, of which one element may be chosen in the drop down list *Type* in the *Description*-page of the Substation-dialog, opened by right-clicking on a graphical substation element and choosing *Edit Substation*. The list contains the Substation Types defined in the according page of the Project Settings dialog.

The column 'Symbol' is storing the name of the symbol, which is searched by default in the subfolder *Database/System/Library/Graphic/Symbols/SGL/Composites*. As additional source, the project's sub-folder *Settings/Additional Symbols* is taken into account.

### 10.3.6 Drawing Format

The drawing area for single line diagrams, block diagrams and plots is modified in the *Drawing Format* dialog, accessed using the  button. A predefined paper format can be selected as-is, edited, or a new format can be defined. The selected paper format has 'Landscape' orientation by default and can be rotated by 90 degrees by selecting 'Portrait'. The format definitions, which are shown when an existing format is edited or when a new format is defined, also show the landscape dimensions for the paper format.

It is not possible to draw outside the selected drawing area. If a drawing no longer fits to the selected drawing size, then a larger format should be selected. The existing graphs or diagrams are repositioned on the new format (use **Ctrl+A** to mark all objects and then grab and move the entire graphic by left clicking and holding the mouse key down on one of the marked objects; drag the graphic to a new position if required).

### 10.3.7 Layers

The single line, geographic and block diagrams use transparent layers of drawing sheets on which the graphical symbols are placed. Each of these layers may be set to be visible or not, and layer depth functionality is built in, meaning that the user can select the order in which layers sit on top of one another.

Which layers are visible and exactly what is shown on a layer is defined in the *Diagram Layer Configuration* dialog, accessed through the graphic toolbar  or selecting *View → Layers...* from the main menu.

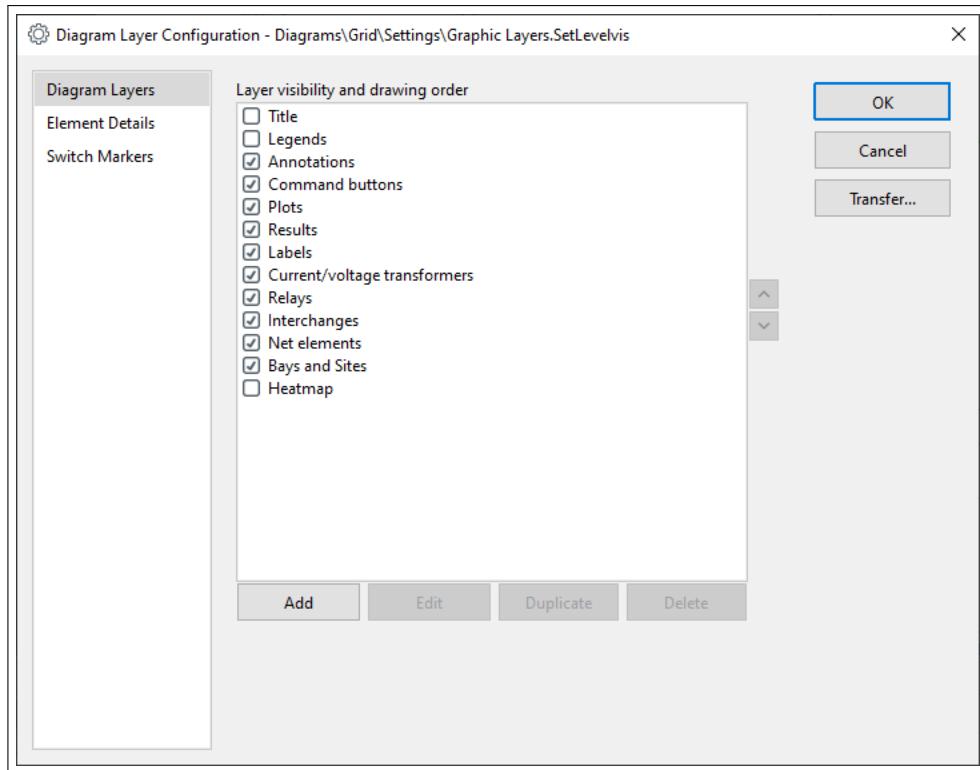


Figure 10.3.3: Diagram Layer Configuration (*SetLevelvis*)

As shown in Figure 10.3.3, the Diagram Layer Configuration dialog has three pages:

- **Diagram Layers**, where default and user-defined layers are created, modified, selected and ordered
- **Element Details**, where symbols related to specific elements are selected to be visible or not
- **Switch Markers**, where layers specifically to switch annotation are selected visible or not

These are described in more detail in the following subsections.

The **Transfer...** button is used to transfer settings etc. from the current diagram to other diagrams. There are three options:

- **Transfer visibility of layers:** This transfers the current visibility selections for diagram layers. These are the settings shown on the Diagram Layers page.
- **Transfer settings of elements and switch markers:** This transfers the settings from the Element Details and Switch Markers pages.
- **Copy user-defined layers to other diagrams:** This copies user-defined layers, both standard and filter layers. Objects already represented in the target diagram are not transferred to the copied layers. This third option is particularly useful for filter layers, where elements will be dynamically assigned to the layer, as described in Section 10.3.7.3

Once the user has made the required selections and clicked on OK, a browser showing all diagrams will be presented, and the target diagrams for the copying of the layers can be selected.

### 10.3.7.1 Diagram Layers

On this page, the default and user-defined layers are listed. The visibility of any one layer is determined by the check-box. New layers can be created using the **Add layer** button and deleted using the

**Delete layer** button. The detailed configuration of any layer can be changed using the **Edit layer** button, or by double-clicking on the layer name. The order of the layers can be changed using the up and down arrows to the right of the list (see Figure 10.3.3 above).

**Note:** It is possible to mix annotation graphical objects with other graphical objects in any user-defined layer. Nevertheless, there is still an “Annotation Layer” where such objects will by default be placed when created. See Section 10.6 for more details.

Depending upon the type of graphic, the following layers are available as a default:

Layer	Content	Configuration Options	Diagram Type: SL Single Line, GEO Geographic, B Block
<b>Title</b>	Graphic title	Edit title	SL/GEO/B
<b>Legends</b>	Results boxes legend and colour legend	Edit legends	SL/GEO
<b>Annotations</b>	Additional graphical information	See Section 10.6	SL/GEO/B
<b>Command buttons</b>	Command buttons used mostly for scripts	Text boxes format	SL/GEO
<b>Plots</b>	Plots placed in the diagrams	Plots background	SL/GEO/B
<b>Results</b>	Boxes with calculation results. See <b>Note</b> below	Text boxes format	SL/GEO/B
<b>Labels</b>	Boxes with names and additional data description, if configured	Text boxes format	SL/GEO/B
<b>Net elements</b>	Symbols for the elements of the grid	Text boxes format	SL/GEO/B
<b>Interchanges</b>	Boxes and arrows, together with associated results boxes	Separate visibility and configuration of boxes and arrows	SL/GEO
<b>Heatmap</b>	Accented colouring of the diagrams	none	SL/GEO
<b>Relays</b>	Graphical representation of relays	Text boxes format	SL
<b>Current/voltage transformers</b>	Contains the drawn current and voltage transformers	Text boxes format	SL
<b>Bays and Sites</b>	Bay representation and Site frames	Text boxes format and style of bays and sites	SL
<b>Load/generation distribution</b>	Shows circles for load and generation around substations	Selection of S, P or Q, colour-settings	GEO
<b>Geographic map</b>	Geographical map used as the background to allow easier drawing of the diagram	Map Provider (see Section 10.8), map type and graphic settings	GEO
<b>Device data</b>	Additional Text explanation given in the device symbol	Text boxes format	B

Layer	Content	Configuration Options	Diagram Type: SL Single Line, GEO Geographic, B Block
Block Definition	Definition each block is based on	Text boxes format	B
Signals	Name of the signal transmitted	Text boxes format	B
Background image	Graphic used as the background (“wallpaper”) to allow easier drawing of the diagram or to show additional information (map information)	Name of file with graphics (WMF, BMP, JPEG, PNG, GIF, TIF)	SL/B

Table 10.3.1: Predefined diagram layers in the layers dialog

### 10.3.7.2 Working with Diagram Layers

Each graphic symbol in a single line, geographic or block diagram is assigned by default to the corresponding layer at first. All busbar symbols, for example, are drawn on the *Net elements* layer by default, their name boxes on the layer *Labels*. Graphic symbols may be shifted onto other layers by right-clicking them in the single line graphic and selecting the option *Shift to Layer* from the context menu, then selecting the layer from the list presented. Should an object disappear when it has been re-assigned to a layer, the visibility of the target layer should be checked. Moving symbols from one layer to another is normally needed when only a few symbols from a certain group should be made visible (for instance the result boxes of one or two specific junction nodes), or when user defined layers are used. This allows to hide some elements or text boxes to improve the clarity of the diagram, or to show additional information for e.g. printing purposes.

**Note:** Certain names and result boxes are, by default, not visible. An example is the names and result boxes for internal nodes. This is done to reduce clutter on the graphic. To display such labels, simply right-click on the object and select *Text Boxes → Show Labels*.

The default layer for annotations is called Annotations Layer, which is empty until the first annotation object is inserted. If the user creates new layers, annotation objects can easily be shifted from one to another by right-clicking on them, selecting *Shift to layer* and select the destination layer from the list. More information about annotations can be found in Section 10.6.

For some layers, for example Text Boxes or Results, when the layer is edited to change the configuration, a *target* may be set; this target will be the focus of the performed configuration command. Various actions or settings may be performed, such as e.g. changing the font using the **Change Font** button. The configuration page may also be used to mark (select/highlight) the target objects in the graphic using the **Mark** button.

The options available to configure a layer depend on the type of Layer. Table 10.3.1 shows for each layer in which way its content can be changed in format.

As an example, suppose that a part of the single line graphics is to be changed, for instance, to allow longer busbar names. To change the settings, the correct graphical layer is first selected. In this example, it will be the *Labels* layer. In this layer, only the busbar names are to be changed, and the target must therefore be set to *All Nodes*. When the layer and the target has been selected, the width for object names may be set in the *Settings* area. The number of columns may be set using the **Visibility/Width** button. Alternatively, the **Adapt Width** will adapt all of the object name placeholders to the length of the name for each object. Changing a setting for all nodes or all branches at once will overwrite the present settings.

The visibility of the layer can be selected within the edit dialog of that layer (as well as in the main Layers dialog), and - as additional options - users can choose to set maximum and/or minimum zoom-dependent visibility levels, so that objects are not shown outside these zoom levels. For geographic diagrams the limits are expressed in terms of a map-scale such as 1:5000, for schematic diagrams as a percentage zoom level.

Many layers have an option to make the objects selectable or not. This can be useful if the layer contains annotation objects, for example.

For certain layers for example the Results layer, the Labels layer and user-defined layers, there is the possibility to define dedicated colours. The colours can be set using the colour palettes within the object dialog of each layer. See Section 4.7.8.1 for more information about the use of colour palettes. After selection of the colour, the colouring mode for layers can be activated using the *Diagram Colouring* icon . This allows the user to easily distinguish between the different layers present in the drawing and to easily identify which elements belong to each respective layer.

### 10.3.7.3 Filter Layers

Filter layers are a particular sort of user-defined diagram layer for network elements. Generally speaking, graphic representations of network elements are “permanently” assigned to a diagram layer, although they can be shifted from one layer to another. With filter layers, on the other hand, elements are dynamically assigned to the layers according to inbuilt or user-defined criteria. Examples of use-cases include creating filter layers according to voltage level, or according to ownership. This allows the filtering of objects of interest within a diagram.

#### How filter layers work

- A filter layer is created by the user and contains a set of one or more filter options.
- An element will be contained in the layer if it meets the criteria of at least one of the filter options in the set.
- An element can be in more than one filter layer (unlike “standard” layers).
- The visibility of filter layers can be individually set, as for other layers.
- There is also an option to activate or deactivate **all** filter layers.
- If filter layers are not active, they are ignored.
- If filter layers are active, any element that is contained in one or more of the filter layers is no longer represented in its standard layer (normally the net elements layer). This means that the element will only be visible if a filter layer that contains it is selected to be visible.

#### Creating a filter layer

- In the Layers dialog (*SetLevelvis*), the **Add** button is used to create a user-defined layer.
- The layer is given a name, and the layer type *Filter* is selected.
- A new panel appears, where the filter set can be defined. The **Generate...** button is used for this process.
- In the *Filter creation* tool, the filter type drop-down menu offers a number of inbuilt options. If one of these is selected, the user then adds the necessary information to define the filter, which is automatically contained in a filter set.
- Alternatively, the user can select just *Custom* and click on OK, then carry on to define the filter(s) as required.
- Using OK completes the definition.

#### 10.3.7.4 Element Details

For some elements shown on diagrams, additional details can optionally be shown, such as the vector groups for transformers or symbols to indicate the number of phases of a line. On the *Element Details* page, such details are turned on or off, with additional configuration for the Power flow direction arrows option.

- **Lines:**

- Connection points: dots at the connections between edges and buses/terminals and signal connections to blocks.
- Connection arrows: double-arrow at connections where the end point is not represented in the current diagram.
- Phases: number of phases of a line/cable, shown as parallel lines.
- Sections and line loads: symbols at lines consisting of sections and/or where line loads are connected.
- Line compensations: symbols at the ends of the lines, indicating the presence of line compensation.
- Fuses in bays: symbol for fuses located in bays.

- **Transformers, machines, shunts:**

- Vector groups: vector group for rotating machines and transformers.
- Tap positions: positions of taps for shunts and transformers.

- **Power flow direction arrows:** arrows to represent the power flow, which can be configured for:

- Active power
- Reactive power
- Active power (zero sequence)
- Reactive power (zero sequence)
- Active power (negative sequence)
- Reactive power (negative sequence)

In addition, the flow arrows can be animated, by selecting the option *Show animation*

#### 10.3.7.5 Switch Markers

On the Switch Markers page, markers specific to operation and/or status of switches (ElmCoup or StaSwitch) can be selected and configured:

##### Remotely controlled substations

This type of marker indicates if sites (ElmSite), substations (ElmSubstat) or secondary substations (ElmTrfstat) host remote controlled switches. This switch attribute can be defined in the field Fault Separation/Power Restoration, on the page Reliability of the switch edit dialog.

If checked, an user-defined colored circle is displayed behind sites, substations and secondary substations, where at least one switch within the aforementioned grouping object is set to be remote-controlled.

This switch mark is only visible if the grouping object has a Composite Node (Beach Ball) graphical representation. Additional information about node graphical representation can be found in Section [12.2.7](#).

##### Tie open points

The marker suggests that a feeder (ElmFeeder) terminates at the open switch.

If this option is selected, an user-defined colored circle is shown around open switches, if different feeders are found on either side, also if no feeder is detected at all on one side.

In addition, in the case that the switch at which the feeders terminate is located within a site, substation or secondary substation, the circle mark will be also displayed on diagrams where a composite node representation of the grouping object exists. It should be noted that the mark will be only presented at the composite node representation of a grouping object, if none of the connected branch objects already shows a tie open point circle at its line end.

### Normally open switches

This is self-explanatory. A circle mark is depicted around switches, if the normal state of a switch is open. The attribute that specifies whether the normal state of a circuit breaker is open can be defined in the Tie Open Point Optimisation page of the switch edit dialog.

For those cases where normally open switches are located within a site, substation or secondary substation, the circle mark will be also displayed on diagrams where a composite node representation of the aforementioned grouping object exists. It should be noted that the mark will be only presented at the composite node representation of a grouping object, if none of the connected branch objects already shows a normally open switch at its end.

### Open standby switches

It shows if despite opening a switch within a substation or secondary substation, elements on both sides of the switch remain supplied. In other words, it suggests that the aforementioned elements are still connected to a source (standby). The annotation mark works only for devices which belong to a substation or secondary substation (Bus couplers are excluded).

### Reclose switches

If this option is selected, an user-defined coloured circle is shown around switches that have the option *Consider as switch with automatic reclosing device* selected on the *Reliability* page of the switch.

## 10.3.8 Show Legend Layer

The legend blocks can be turned on and off from the Layers dialog (see [10.3.7](#)), or from the single line diagram toolbar ().

The results legend block describes the contents of result boxes (for information about result boxes see [10.5](#)).

Because more than one type of result box is normally used in the Single line graphic, for instance, one for node results and another one for branch results, the legend box normally shows more than one column of legends. After changing the result box definitions, it may be necessary to manually resize the legend box in order to show all result box legends.

The Legend Box definition dialog is opened by right-clicking the legend block and selecting *Edit* from the context menu. The font and format shown may be configured. When opening a new graphic the legend will appear by default.

The other block is the colour legend block, which updates automatically based on the colouring options selected.

### 10.3.9 Show Result Layer

Results layer can be easily toggled on and off without going into the Layers command by using the icon *Show Result Layer* (15) on the graphics toolbar.

### 10.3.10 Colouring Options

#### 10.3.10.1 Diagram Colouring

The single line and geographic diagrams have an automatic colour representation mode. The *Diagram Colouring* icon (16) on the diagrams toolbar will open the diagram colouring representation dialog (alternatively, select *View → Diagram Colouring* on the main menu). This dialog is used to select different colouring modes and is dependent if a calculation has been performed or not. If a specific calculation is valid, then the selected colouring for that calculation is displayed.

As described below, colours are selected via the *Colour Settings* button found in the *Diagram Colouring* dialog. Clicking on this button brings up a new dialog, which has two pages. On the first page called “General”, the user has the option to select a colour palette. This is not essential, but if a palette is selected, the user will find that colours from this palette will be offered as a default selection when configuring colours for elements in diagrams or in a Network Model Manager or Data Manager. For general information about configuring colours and the use of colour palettes see Section 4.7.8.

The *Diagram Colouring* has a 3-priority level colouring scheme implemented, allowing colouring elements according to the following criteria: 1<sup>st</sup> Energising status, 2<sup>nd</sup> Alarm and 3<sup>rd</sup> “Normal” (Other) colouring.

**Energising Status:** if this check box is enabled “De-energised” or “Out of Calculation” elements are coloured according to the settings in the “Project Colour Settings”. The settings of the “De-energised” or “Out of Calculation” mode can be edited by clicking on the *Colour Settings* button.

**Alarm:** if this check box is enabled a drop down list containing alarm modes will be available. It is important to note here that only alarm modes available for the current calculation page will be listed. If an alarm mode is selected, elements “exceeding” the corresponding limit are coloured. Limits and colours can be defined by clicking on the *Colour Settings* button.

**“Normal” (Other) Colouring:** here, two lists are displayed. The first list contains all available colouring modes. The second list contains all sub modes of the selected colouring mode. The settings of the different colouring modes can be edited by clicking on the *Colour Settings* button.

Every element can be coloured by one of the three previous criteria. Also, every criterion is optional and will be skipped if disabled. Regarding the priority, if the user enables all three criteria, the hierarchy taken into account will be the following:

- “Energising Status” overrules the “Alarm” and “Normal Colouring” mode. The “Alarm” mode overrules the “Normal Colouring” mode.

The graphic can be coloured according to the following list. Availability of some options will depend on the function that is selected (e.g. ‘Voltage Violations’ does not appear when the ‘Basic Data’ page is selected, but does when the ‘Load Flow’ page is selected) and on the licence (e.g. Connection Request is only available if the advanced function Connection Request Assessment is part of the licence).

#### **Energising Status:**

- De-energised
- Out of Calculation
- De-energised, Planned Outage

**Alarm:**

- Feeder Radiality Check (Only if “Feeder is supposed to be operated radially” is selected).
- Outages
- Overloading of Thermal/Peak Short Circuit Current
- Voltage Violations/Overloadings

**“Normal” (Other) Colouring:**

- Results
  - Fault Clearing Times
  - Voltage Angle (colouring according to absolute or relative voltage angles and angle differences along branch elements; relative voltage angles do not reflect transformer vector groups, while absolute voltage angles include the angle shift caused by transformer vector groups)
  - Voltages / Loading
  - Loading of Thermal / Peak Short-Circuit Current
  - Arc Energy
  - Incident Energy
  - PPE-Category
  - Connection Request: Approval Status
  - Contribution to EIC
  - Contribution to ENS
  - Contribution to SAIDI
  - Contribution to SAIFI
  - Loads: Average Interruption Duration
  - Loads: Load Point Energy Not Supplied
  - Loads: Yearly interruption frequency
  - Loads: Yearly interruption time
  - Optimal Manual Restoration
  - Hosted Power
  - Probabilistic Analysis
  - Unit Commitment - Redispatch
  - State Estimation
- Topology
  - Boundaries (Definition)
  - Boundaries (Interior Region)
  - Connected Components
  - Connected Components, Voltage Level
  - Connected Grid Components
  - Energising Status
  - Feeders
  - Missing graphical connections
  - Outage Check
  - Station Connectivity
  - Station Connectivity (Beach Balls only)
  - Supplied by Secondary Substation

- Supplied by Substation
- System Type AC/DC and Phases
- Voltage Levels
- Primary Equipment
  - Cross Section
  - Year of Construction
  - Forced Outage Duration
  - Forced Outage Rate
- Secondary Equipment
  - Measurement Locations
  - Power Restoration
  - Relays, Fuses, Current and Voltage Transformers
  - Switches, Type of Usage
- Groupings (Grids, Zones, Areas...)
  - Areas
  - Grids
  - Layers
  - Meteo Stations
  - Operators
  - Owners
  - Paths
  - Routes
  - Zones
- Variations
  - Modifications in Recording Expansion Stage
  - Modifications in Variations
  - Original Locations
- User-defined
  - Individual

The list *User-defined* may be used to define own colouring schemes. Pressing **Manage Filters...** opens an object browser with the list of all available user-defined filters, found in the subfolder Settings/-Colouring/Colouring Scheme. New filter sets (*IntFiltset*) can be created, containing several General Filter objects (*SetFilt*) with an assigned colour and the conditions, under which an element is coloured. This allows to implement very specific filters to identify graphically elements in the diagram with certain properties or results.

### 10.3.10.2 Heatmaps

In *PowerFactory*, heatmaps can be used to illustrate the state of a grid by colouring the area around network elements. The colour definition is carried out as described in Section [10.3.10.1](#).

To use the colour definition for heatmaps, click on the *Heatmap* button . On the *General* page of the dialog, the basic settings for the creation of the heatmap can be selected. The colour settings dialog (explained in Section [10.3.10.1](#)) is accessible from this page. The *Mode* shows which type of colouring is used.

The resolution of the Heatmap can be:

- Low
  - Medium
  - High
  - User-defined (in pixels)
- 

**Note:** The amount of time required to generate each heatmap increases with the specified resolution. Since the optimal settings for heatmaps vary for each grid, the process of finding this optimum might take a few iterations. Therefore it is advised to start with a small or medium resolution.

---

A background colour for the heatmap may additionally be set.

The *General* page defines the general settings for the Heatmap and the *Advanced* page defines specifics regarding the colouring. Five different parameters can be set; the first two being:

- **Number of closest influence points:** defines the number of reference points taken into account when colouring a certain point of the Heatmap.
- **Contour sharpness:** defines the smoothness of the transition between differently coloured areas.

The other three parameters define the *Fading Area*, i.e. the orthogonal transition from the element colouring to the background colour:

- **Begin:** defines how far away from the centre of the element on the lateral axis the colouring begins to fade to the background colour.
- **Extent:** defines how far away from the centre of the element on the lateral axis the colouring ends to fade to the background colour.
- **Fading exponent:** defines how fast the colour between *Begin* and *Extent* will fade to the background colour.

Figure 10.3.4 shows an example of a Heatmap, which is coloured according to loading, over- and under-voltage.

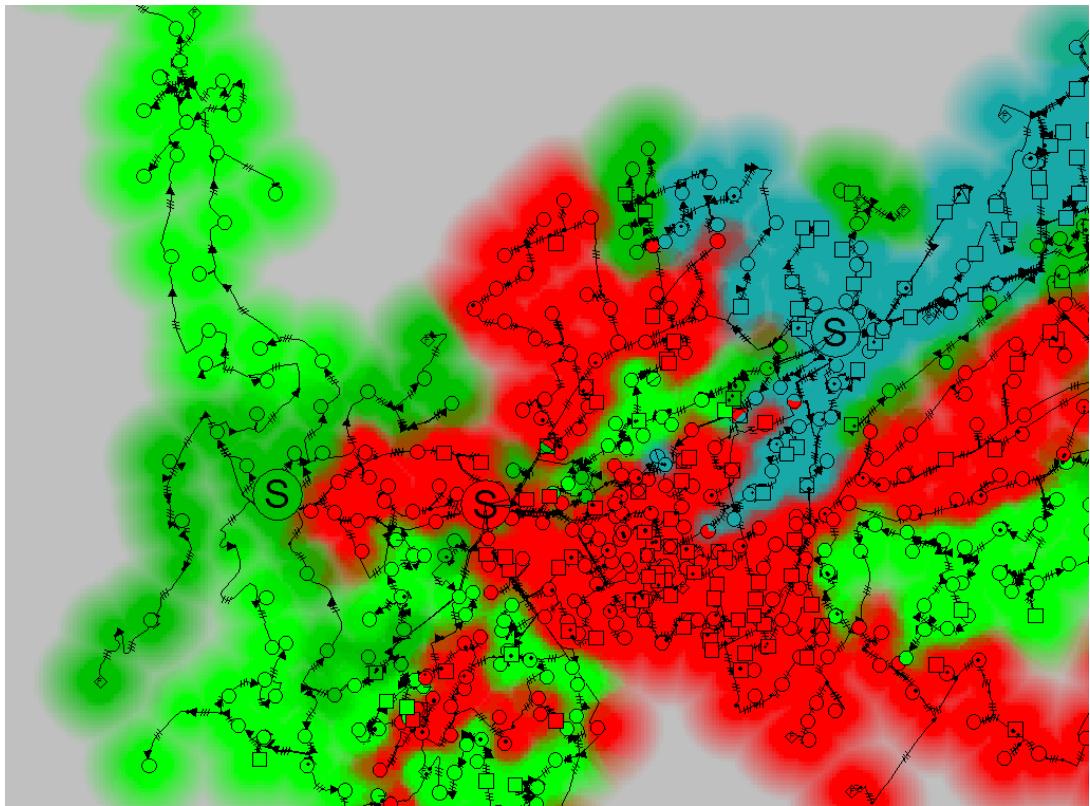


Figure 10.3.4: Example of a heatmap coloured according to loading, over- and under-voltage

### 10.3.11 Node Default Options

Figure 10.3.5 shows the commands available for setting node default options. These are discussed in further detail in this section.

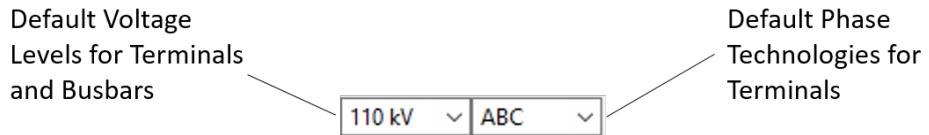


Figure 10.3.5: Node default options

#### **Default Voltage Levels for Terminals and Busbars:**

The default voltage level for terminals can be set in this field. New terminals placed on the single line diagram will have this voltage (e.g. 110 kV, 0.4 kV).

#### **Default Phase Technologies for Terminals:**

The default phase technology for terminals can be set in this field. New terminals placed on the single line diagram will be of this type (e.g. three-phase ABC, single-phase, DC, etc.).

### 10.3.12 Print and Export Options

#### 10.3.12.1 Print

This function is accessed via the  button, the menu *File* → *Print* or via the hotkey **Ctrl+P**. It opens a print preview, showing the diagram to be printed. The dialog offers all the options that the user would expect, including a Print Setup dialog, accessed via the  icon.

If only part of the diagram is to be printed, the selection of the area can be done within the print preview dialog. There is also a *Pages* option, which allows the user to print selected graphic pages.

On the *Page Setup* field, the format, orientation and frame of the page can be set. For the diagrams, there is an additional option to print as poster (i.e. print the diagram in several pages).

#### 10.3.12.2 Exporting Graphics

Graphics can be exported in a wide range of formats, using the *Export Diagram* command. This is accessed either from the main toolbar, using *File* → *Export* → *Diagram...*, or by using the *Export Diagram* icon () on the graphics toolbar.

If only part of the diagram is to be exported, the selection of the area can be done within the preview dialog. For PDF exports, there is also a *Pages* option, which allows the selection of multiple graphic pages to be exported. If there is no need to save a copy of the graphic, i.e. it is meant to be pasted into a report, it is possible to export the graphic to the Windows Clipboard. The option *Copy to clipboard* is available for the following formats: EMF, PNG, BMP, JPEG, TIFF and GIF.

### 10.3.13 Diagram Layout Tool

The *Diagram Layout Tool*  is a powerful feature to create graphical representations of network topologies. The tool offers a manual, semi- and fully automatic creation of nodes and branch elements, which are not yet graphically represented in the current network diagram. The options and the dialog are described in detail in Section 12.6.

### 10.3.14 Measure Distance

Distances in the geographic diagram can be measured using the *Measure Distance* tool by clicking on the icon  and then clicking on the diagram as many times as required. The total distance is displayed in a temporary box. To stop the measurement, the **Esc** key should be pressed.

### 10.3.15 Context Menu Options

Right-clicking on the graphical representation of an element displays the context menu. The options available in this menu depend not only on the selected element, but also on whether the graphic is “frozen” or not. This section describes some of the options.

 **Show in Data Manager:** opens the location (within the project) of the object in the Data Manager. See Chapter 11 (Data Manager and Network Model Manager) for more information.

 **Mark in Other Graphic:** is used to show the graphical representation of the element in another diagram. If there is a graphical representation of the element in more than one diagram, a list of the diagrams where the element is available is displayed.

**Connect:** right-click and select *Connect* to connect an element.

**Reconnect:** used to disconnect a selected element and then re-connect it. The branch to be connected will be 'glued' to the cursor. Left clicking a bar or terminal will connect the element. When right-clicking at the end of a connection element a different/reduced menu is shown which allows reconnecting just the selected side (*Reconnect Side*).

**Disconnect:** disconnects the selected element/s. When right-clicking at the end of a connection element a different/reduced menu is shown which allows disconnecting just the selected side (*Disconnect Side*)

 **Edit Layer:** opens the layer the elements belong to. See section [Layers](#).

**Shift to Layer:** the element can be moved to a different layer. The default layer and the custom (previously created) layers will appear in the list. See section [Diagram Layers](#).

 **Cut:** this function cuts the selected objects. Objects can then later be pasted as discussed below. Also accessed through the keyboard: **Ctrl+X** key.

 **Copy:** copies the selected objects to the clipboard. Also accessed through the keyboard: **Ctrl+C** key.

 **Paste:** pastes all objects from the clipboard into the current drawing. The objects are pasted at the current graphical mouse position. Objects that are copied and pasted create completely new graphic and data objects in the graphic that they are pasted into. Also accessed through the keyboard: **Ctrl+V** key.

---

**Note:** To copy and paste just the graphic,  *Paste Graphically Only* should be chosen from the right-click menu. Similar results are obtained when using the "Draw Existing Net Elements" tool (see Section [12.6: Drawing Existing Elements using the Diagram Layout Tool](#)).

 **Delete:** this function deletes both the database and graphical objects for the selected element(s). A warning message will appear first - this may be switched off in the *User Settings* dialog; see Section [7.3 \(Graphic Windows Settings\)](#)). Also accessed through the keyboard: **Del** key.

**Delete Graphic Object Only:** is used to delete only the graphical representation of the object in the current diagram. The object remains in the database.

---

**Note:** The undo command undoes the last graphic action and restores deleted elements, or deletes created elements. The undo command is accessed through the *undo* icon () or by pressing **Ctrl+Z**.

---

### Graphic Object options

**Redraw:** the option *Graphic Object → Redraw* is used to redraw a selected element. When right-clicking at the end of a connection element a different/reduced menu is shown, which allows reconnection of just the selected side (*Graphic Object → Redraw Side*)

**Rotate:** the option *Graphic Object → Rotate* rotates symbols clockwise, counter-clockwise, or 180 degrees. It is generally preferable to disconnect an element before rotating it.

**Set Individual Line Style:** if the option to allow individual line style is selected in the [Diagram Settings](#), the line style can be changed using this menu icon.

**Set Individual Line Width:** as for the individual line style.

**Add or delete Line Points:** the option *Graphic Object* → *Edit Line Points* will show the blue circles ('line points') that define the shape of the graphic object. Each of these circles can be moved by left clicking and dragging them to a new position. New circles can be inserted by left clicking the connection in between circles. Line points are deleted by right-clicking them and selecting the *Graphic Object* → *Delete Line Point* option from the context menu. This menu also presents the option to stop (end) the line point editing, which can also be done by left clicking somewhere outside the selected lines.

**Push to Back:** when multiple objects are lying on top of each other in network graphics, there is a possibility to temporarily push the elements back to be able to select other elements. For this purpose, the **Ctrl** key should be pressed and then right-click on the element, which is to be pushed to the back. The *Push to Back* option appears at the bottom of the context menu.

Marked objects can be moved by left clicking them and holding down the mouse button. The objects can be moved when the cursor changes to an arrowed cross (). Hold down the mouse button and drag the marked objects to their new position. Connections from the moved part of the drawing to other objects will be adjusted.

## 10.4 Graphical Symbols of Elements

The symbols used for the graphical representation of the elements can be changed. *PowerFactory* offers a number of symbols per element; these symbols are located in the Database in folder *System* → *Library* → *Graphic* → *Symbols*. To change the symbol used, right-click on the element and select *Graphic Object* → *Select Symbol...* and a window with the list of available symbols for the selected element will be displayed.

The original symbol used for the graphical representation of the elements can be used as a template to design a new symbol according to the user's needs or even create a completely new symbol from scratch. Both options are described in the following sections.

### 10.4.1 Create a new symbol based on a library symbol

To use an existing symbol from the system library to create a user-designed symbol, the following steps should be followed:

- Right click on a graphical representation of the element and select *Graphic Object* → *Open Symbol in Graphical Editor...*
- An info window will be displayed, indicating that a copy of the library symbol will be created in the project settings. Click on **Yes** to create the new symbol.
- Assign a name to the new symbol in the displayed input dialog.
- A graphic of the symbol will be opened in a new window group (or the additional existing one).
- An example is shown in Figure 10.4.1.

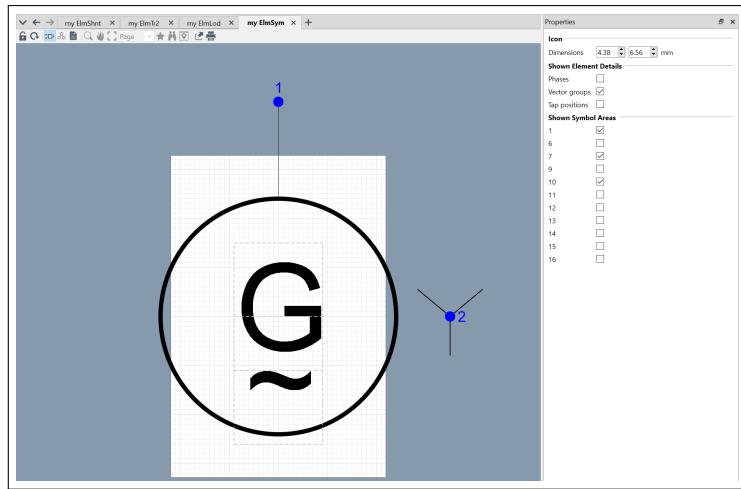


Figure 10.4.1: Graphic Symbol of a Synchronous Machine

The connection points of the element are represented by small blue circles and numbers. These can only be modified changing the XY coordinates defined in the edit dialog of the symbol, which can be opened using the *Show Model Dialog* button from the toolbar.

The *Properties* table contains the following:

- **Icon Dimensions:** the size of the icon, i.e. the size in mm of the white area where the symbol is drawn.
- **Shown Element Details:** the details of the element shown in the graphical symbol. By default it takes what is defined on the *Element Details* page of the layers configuration (see Section 10.3.7.4). Selecting the different checkboxes will show the graphical representation of the following details:
  - Phases
  - Vector groups
  - Tap positions
- **Shown Symbol Areas:** the graphical symbol is divided in areas. The checkboxes selected on this part correspond to what is actually displayed in the diagram. Selecting the different checkboxes will show the different graphical representations of the element (e.g. delta or star neutral connection)

The different parts of the symbol can be modified by clicking on them. For example, when clicking on the circle of the synchronous machine, the *Properties* window will change as shown in Figure 10.4.2, and the properties of this selected part, such as the line style, width, colour and angle, can be modified.

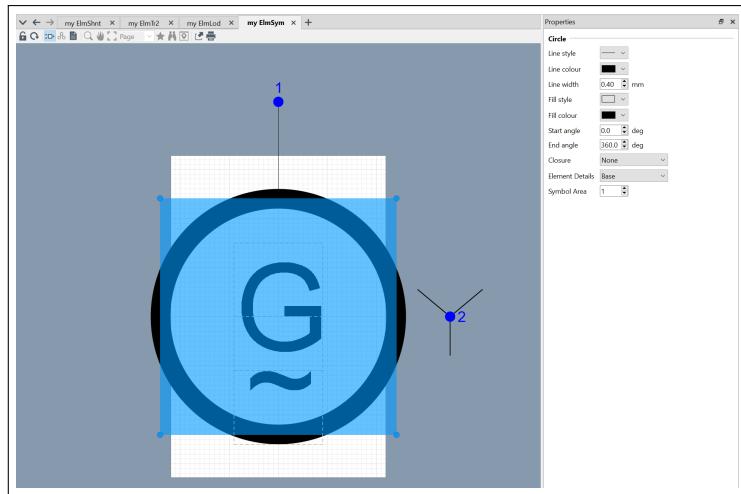


Figure 10.4.2: Changing the style of the symbol

The new symbol will be automatically saved and applied to the selected element. It can however be used for other elements of the same class if the symbol is selected as described before (*Right-click → Graphic Object → Select Symbol...*). Alternatively the Data Manager can be used to filter all the graphical objects using the original symbol (parameter `sSymNam`) and assign the new symbol.

### 10.4.2 Create a new symbol from the Data Manager

Although it is recommended to use the existing symbols as template to build new ones, it is possible to build a new graphic symbol from scratch in the Data Manager as follows:

- Go to the folder *Settings → Additional Symbols* within the project.
- Click on the *New Object* (+) icon and select *Symbol*
- The symbol dialog will open, the following fields should be defined on the *General* page:
  - *Symbol Description*: a description of the symbol (e.g. element represented in the symbol)
  - *Object Type*: class of the element that will use the symbol (e.g. *ElmSym*)
  - *Type of graphical representation*: select between *node* (substations, terminals), *branch* (branch elements, e.g. generators, loads, transformers, etc.) and *dummy* (a graphic that does not necessarily represent an element)
  - *Maximum number of connections*: the number of connections of the element
  - *Connection points*: the XY coordinates of the connection points of the symbol, which can be set to 0,0 or some other small value as a starting point
  - The **Open symbol editor** button can be used to open the diagram of the symbol
  - The symbol dialog can be closed using **OK**

Once the symbol editor is opened, the drawing tools *Annotations* can be used to insert lines, free-form polygons, square, circles and text. (If the drawing tools are not visible on the left-hand side of the *PowerFactory* window, use the icon to unfreeze the graphic.) The connection point should be visible; zoom in if necessary.

Alternatively, the symbol can be defined with a Modelica compatible code on the *Geometry* page of the edit dialog of the symbol.

The Modelica symbols have per default a bigger size than the symbols used in *PowerFactory*. To change the size of the symbol, the option *Rescale Symbol*, accessible by right-clicking on the symbol background, should be used. *PowerFactory*'s default symbols have usually a width of 5 mm.

## 10.5 Result Boxes, Text Boxes and Labels

*PowerFactory* uses result boxes, text boxes, and labels in the single sine diagram to display calculation results and other useful information. Figure 10.5.1 illustrates an example of this.

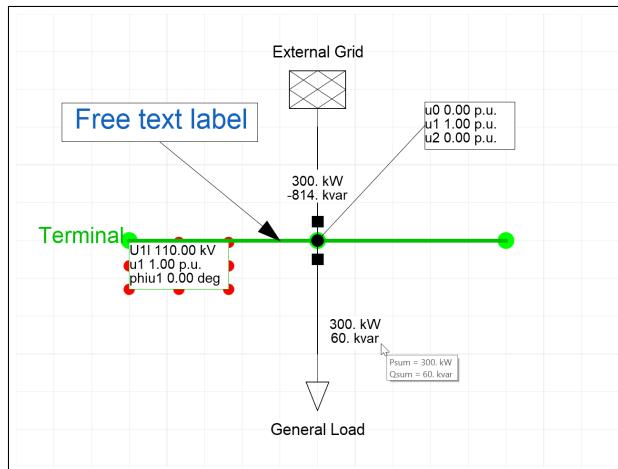


Figure 10.5.1: Results boxes, text boxes, and labels available in *PowerFactory*

### 10.5.1 Result Boxes

#### General

Result boxes are generally set up so that there are a series of different formats for each calculation function, with variables appropriate to that function. In addition, the format differs for the objects class and/or for individual objects. For example, following a load flow calculation, branch and edge elements will have different formats compared to nodes, and an external grid will have an individual, different, format as compared to the branch and edge elements.

Although the result boxes in the single line graphic are a very versatile and powerful way for displaying calculation results, it is often not possible to display a large (part of a) power system without making the result boxes too small to be read. *PowerFactory* solves this problem by offering balloon help on the result boxes. Positioning the mouse over a result box will pop up a yellow text balloon with the text displayed in a fixed size font. This is depicted in Figure 10.5.1. The result box balloon always states the name of the variable, and may thus also be used as a legend.

#### Reference points

A result box is connected to the graphical object for which it displays the results by a “reference point”. Figure 10.5.1 shows the default reference points for the result box of a terminal. A reference point is a connection between a point on the result box (which has 9 optional points), and one of the “docking” points of the graphical object. The terminal has three docking points: on the left, in the middle and on the right. The reference point can be changed by:

- Right-clicking the result box with the graphics cursor (freeze mode off), and selecting *Change Reference Points*.
- The reference points are shown: docking points in green, reference points in red. Select one of the reference points by left-clicking it.
- Left-click the selected red reference point, and drag it to a green docking point and drop it.
- An error message will result if you drop a reference point somewhere else than on a docking point.

Result boxes can be freely moved around the diagram. They will remain attached to the docking point, and will move along with the docking point. A result box can be positioned back to its docking point by right-clicking it and selecting *Reset Settings* from the menu.

### Editing Result Boxes

*PowerFactory* uses separate result boxes for different groups of power system objects, such as node objects (i.e. busbars, terminals) or edge objects (i.e. lines, loads). For each type of result box, a different result box definition is used.

A number of these predefined formats are available for display; they may be selected by right-clicking a result box to get the *Format for Edge Elements* (in this example) option, which then presents a number of formats that may be selected. The active format is ticked () and applies for all the visualised edge elements.

It is also possible to select predefined formats for a specific element class. If the edge element is for example an asynchronous machine, in the context menu it will be also possible to get the option *Format for Asynchronous Machine*, which shows the predefined formats for the element class Asynchronous Machine (*ElmAsm*). The selected format will in this case apply only to the visualised asynchronous machines.

If the user wants to create a specific format that is different from the pre-defined ones, the *Edit Format for Edge Elements (or Node Elements)* option should be used. Note that the new format will be applied to the entire group of objects (edge or node objects).

If a created format is expected to be used for just one specific element, then the *Create Textbox* option should be used. An additional result box/ textbox will be created, using the current format for the object. This may then be edited. Information about text boxes is given in [10.5.2](#).

When the *Edit Format* option has been selected, the user can modify the variables and how are they showed as described Chapter [19](#): Reporting and Visualising Results, Section [19.2.1](#): Editing Result Boxes.

### Formatting Result Boxes

Result boxes can be formatted by means of the context menu (right-clicking the desired result box). The available options include:

- Shift to layer (see [10.3.7](#)).
- Rotate
- Hide
- Change the font type and size of the text
- Change the width
- Change colour
- Set the text alignment
- Adapt width
- Change reference points
- Reset Settings, only available after changes have been made).

### Resetting Calculation Results

When pressed, the *Reset Calculation* icon () will clear the results shown on the Single Line Diagram. By default, *PowerFactory* will also clear the calculation results when there is a change to network data

or network configuration (such as opening a switch). However, if *Retention of results after network change* is set to *Show last results* in the User Settings (see Section 7.2: General Settings), results will appear in grey on the Single Line Diagram and on the Flexible Data page until the calculation is reset, or a new calculation performed. Reset Calculation can also be accessed from the main Calculation menu or using **F12**.

### 10.5.2 Text Boxes

As mentioned before, text boxes are used to display user defined variables from a specific referenced object within the single line graphic. To create a text box, right-click on the desired object (one end of the object when it is a branch element) and select *Create Textbox*. By default a text box with the same format of the corresponding result box will be generated.

The created text box can be edited, to display the desired variables, following the same procedure described in 10.5.1. In this case after right-clicking the text box, the option *Edit Format* should be selected. By default the text boxes are graphically connected to the referred object by means of a line. This “connection line” can be made invisible if the option *Show line from general text boxes to referenced objects* from the *Text Boxes* page of the Diagram Settings dialog (☰) is disabled.

### 10.5.3 Labels

In the general case, a label showing the name of an element within the single line graphic is automatically created with the graphical objects (see Figure 10.5.1). The label can be visualised as a text box showing only the variable corresponding to the name of the object. As for text boxes, the format of labels can be set using the context menu.

### 10.5.4 Free Text Labels

Free Text Labels (see Figure 10.5.1) can be anchored to an element on the single line diagram, and used to display custom text. They are created by right-click and selecting *Text Boxes* → *Create Text Label*.

## 10.6 Annotations

The Annotations feature offers the user the opportunity to include additional graphical information in one or more configurable layers in the single line, geographic or block diagrams. Examples include:

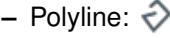
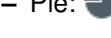
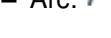
- Built-in graphical annotation elements
- Text
- Icons (bitmap files)
- Plots

---

**Note:** The Annotation Layer is the default layer for new annotation objects, but users can choose to move such objects into their own user-defined layers, which can also contain other graphical objects.

---

The annotation elements are as follows:

- Graphical annotation
  - Line: 
  - Polyline: 
  - Arrow: 
  - Polyline with arrow: 
  - Polygon: 
  - Rectangle: 
  - Circle: 
  - Pie: 
  - Arc: 
- Text: 
- Icons (bitmap files): 
- Plots: 

Except the icons and plots, all the annotation elements can be drawn directly in the diagram. Before placing an icon in the diagram, an available icon-object has to be selected or if not yet existing, created. To insert a plot into the diagram, an already existing plot can be selected from the list in the object browser, which opens after pressing the  button.

It is possible have multiple layers that contain annotations. To do this, the user should click on the  button and then use the **Add layer** button to make a user-defined layer, selecting *Net elements, annotations, text boxes* from the drop-down menu for Layer Type.

### Export graphical layer

To export a graphical layer the user should press the **Edit layer** button, then go to the Annotations page of the dialog. Using the **Export** button, the user can export the layer as an \*.svg file.

### Import graphical layer

To import a graphical layer, the user should first create the new layer as described above, then use the **Import** button on the Annotations page.

#### 10.6.1 Annotation of Protection Device

Adding a protection device into the single line diagram is described in Section [33.2.2](#).

## 10.7 Command Buttons

Command Buttons (*VisButton*) are used to allow DPL or Python scripts to be easily executed by clicking on a button in a single line diagram. A Command Button for a script can be created by turning off Freeze mode in the diagram, enabling Drag and Drop in a Data Manager, selecting the script in the Data Manager, and dragging it to the chosen spot on the diagram.

The options available in the Command Button can be seen by doing right-click, Edit.

- **Command:** here the script is selected. The script can be local or remote, for example from the Configuration, Scripts folder.

- **Edit command before execution:** this gives the user the possibility to change parameters before executing.
- **Use temporary folder for execution:** This is relevant for remote scripts.
  - If not selected, a local copy of the remote script will be created in the study case.
  - If selected, this local copy will instead be generated in the user's folder *Settings/Temp/Calculation Cases*, and will be deleted when *PowerFactory* is closed.
- **Freeze:**
  - If not selected, left-clicking on the button will execute the script if the graphic is in freeze mode but will make it graphically editable if not.
  - If selected, left-clicking on the button will always execute the script, regardless of the freeze mode status of the graphic.
- **Linked Data:** An object can be selected here, and it becomes assigned to the attribute pObj of the Command Button. If the Command Button itself is then supplied to the script as an external object, this pObj can then be used within the script. One use of this mechanism is to facilitate toggling behaviour by scripts.
- **Caption:** This is the text that will appear in the button on the graphic.
- Apart from the font and colour settings on the Advanced page, other settings are no longer used or should only be used if advised by *DlgSILENT*.

## 10.8 Geographic Diagrams

In *PowerFactory* it is possible to specify terminal GPS coordinates, and automatically generate geographic diagrams. GPS coordinates (latitude and longitude) are entered on the *Description* page of terminals and lines, on the *Geographical Coordinates* tab. One geographic diagram can be created per project by either:

- Opening the Data Manager, right-clicking on an active grid and selecting *Diagrams* → *Show Geographic Diagram*.
- On the main menu, under *Insert* → *Geographic Diagram*.

The geographic diagram provides a visual representation of the network and includes all terminals and lines for which GPS coordinates have been entered.

One port elements (e.g. loads, shunts, generators) can also be represented in the geographic diagram. The Diagram Layout Tool can be used to automatically draw all the edge elements in the diagram (see Section 12.6.1.2).

The settings for the geographic diagram are defined in the Diagram Settings, *Geographic Diagram* page (see Section 10.3.5.5).

An additional layer called *Load/Generation Distribution* is available for geographic diagrams, in order to illustrate the magnitude of network load and generation (apparent power), as illustrated in Figure 10.8.1. The *Layers* dialog is accessed using the  icon, and for information about working with layers, see Section 10.3.7 (Layers). If the *Load/Generation Distribution* layer is edited, the colours and other settings can be changed on the Load/Generation page. (For general information about configuring colours, see Section 4.7.8.)

Note that the displayed size of circles does not change as the user zooms in and out of the diagram.

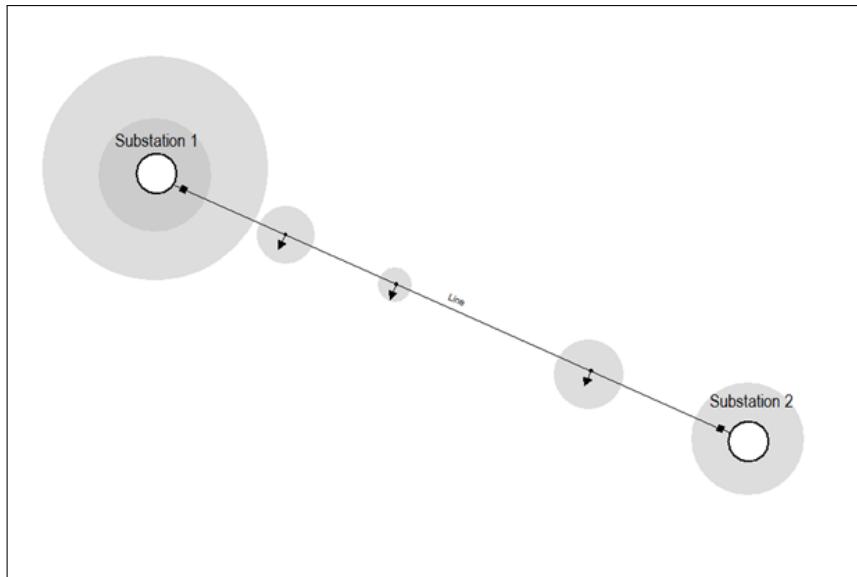


Figure 10.8.1: Load/Generation Distribution example

Maps can be used as background images and can be specified on the *Configuration* page of *Layers* (LAYERS). Maps from the following providers are pre-configured:

- OpenStreetMap (OSM), featuring free-of-charge mapnik-style maps
- Esri ArcGIS®, including road maps, satellite, and hybrid maps
- Google Maps®<sup>1</sup>, including road maps, satellite/aerial, hybrid, and topographic maps
- Bing Maps, including road and satellite maps
- IGN Géoportail, including road maps, satellite and special maps
- Local map files, stored in plain text-files

To use the map data of some providers, special licence keys are necessary, which can be stored in the Geographic Maps page of the configuration dialog accessed via *Tools* → *Configuration*.

### 10.8.1 Using an External Map Provider

If an external map provider from the internet is used, the *Map layer* can be chosen from (depending on which map layers the provider offers):

- Roadmap
- Satellite/Aerial
- Hybrid
- Topographic

The following parameters are available:

- Saturation adjustment
- Brightness adjustment

<sup>1</sup>requires Google Maps for Business account: <http://www.google.com/enterprise/mapsearch>

These parameters are valid in the range -100 % and +100 % and can be used to highlight either the map or the network elements.

Figures 10.8.2 and 10.8.3 illustrate small distribution grids where OpenStreetMap, and Esri ArcGIS® satellite maps, respectively, are used as the background image providers.

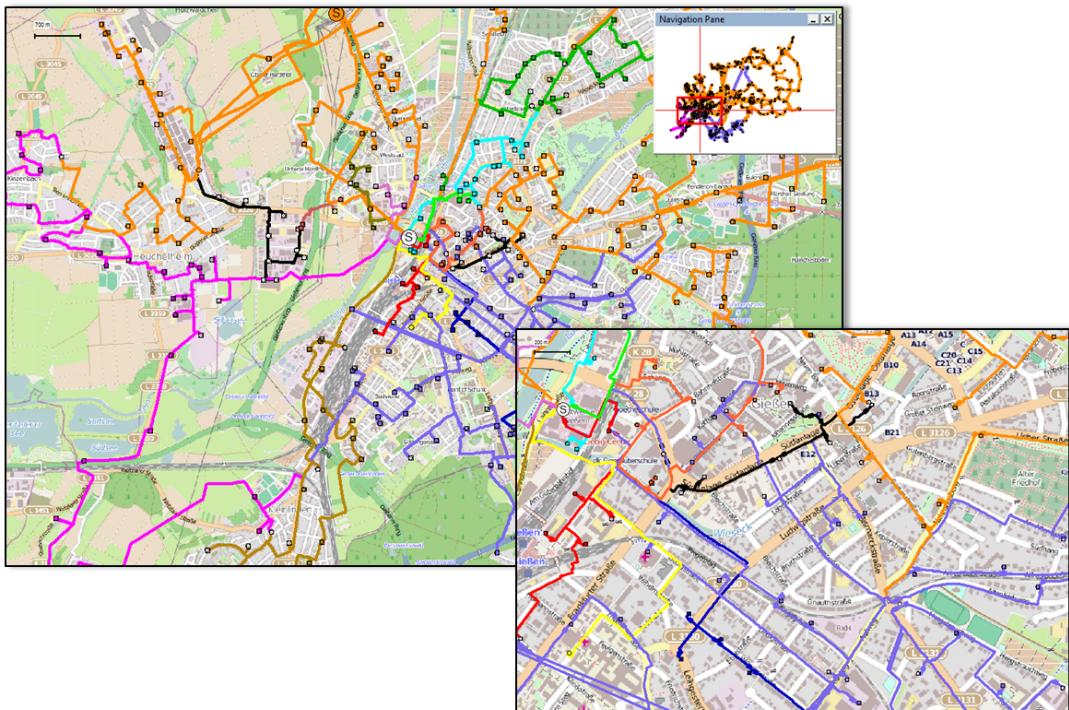


Figure 10.8.2: Network example with OpenStreetMap data

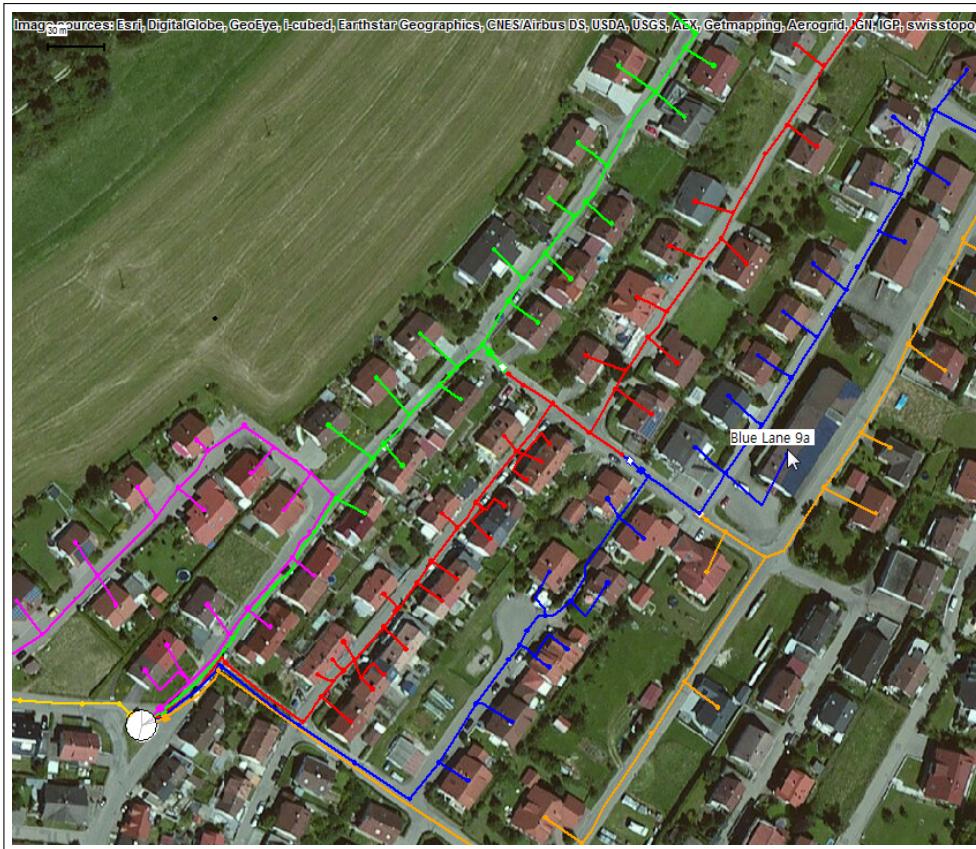


Figure 10.8.3: Network example with satellite background map (ESRI ArcGIS©)

Besides usage of pre-configured built-in map services, *PowerFactory* supports the use of user-configured map services based on the standardised WMS/WMTS protocol. The WMS are defined by the *Administrator* in the *Configuration* folder as shown in Figure 10.8.4.

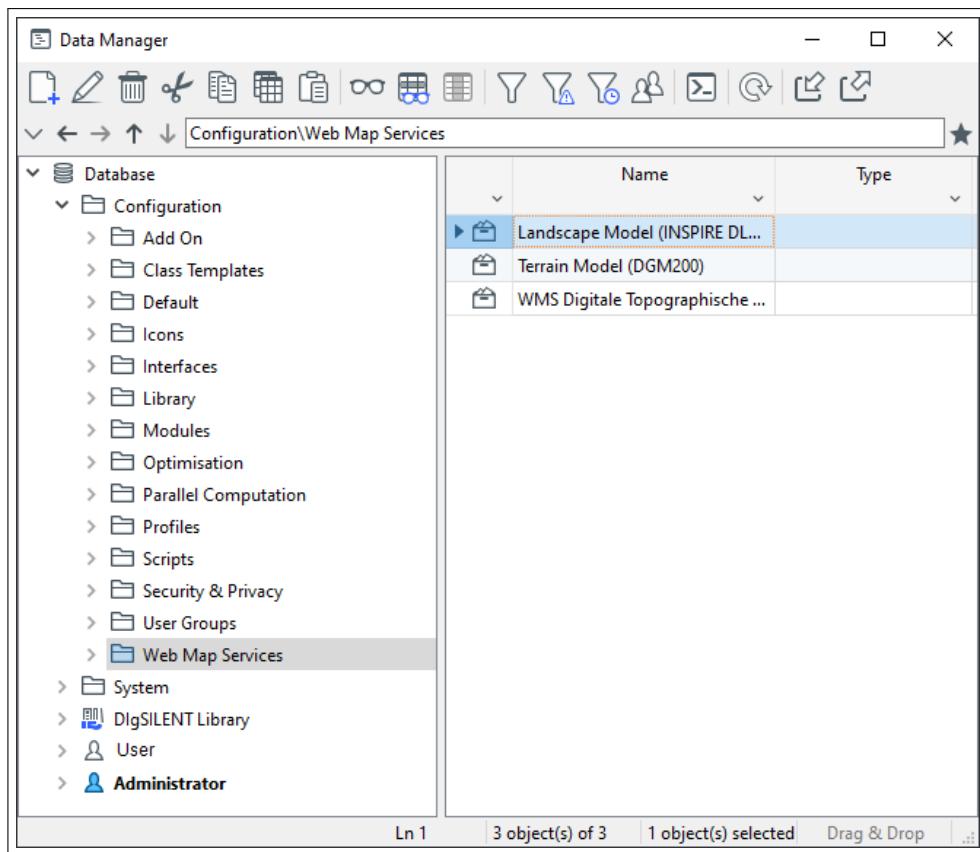


Figure 10.8.4: Web Map Services folder location

A new Web Map Service can be created by clicking on the button *New Object* ( in the Data Manager and selecting *Hyperlink (IntUrl)*. In the edit dialog of the hyperlink, the field *Resource type* must be set to *Map Service*.

Once *Address*, *Map server protocol* and the rest of the fields of the edit dialog are set, it is possible to verify the connection to the Map Service by clicking on [View](#).

Once a Web Map Service is defined by the Administrator, the user can select the desired map by configuring the *Background* layer of the geographic diagram, as shown in Figure 10.8.5.

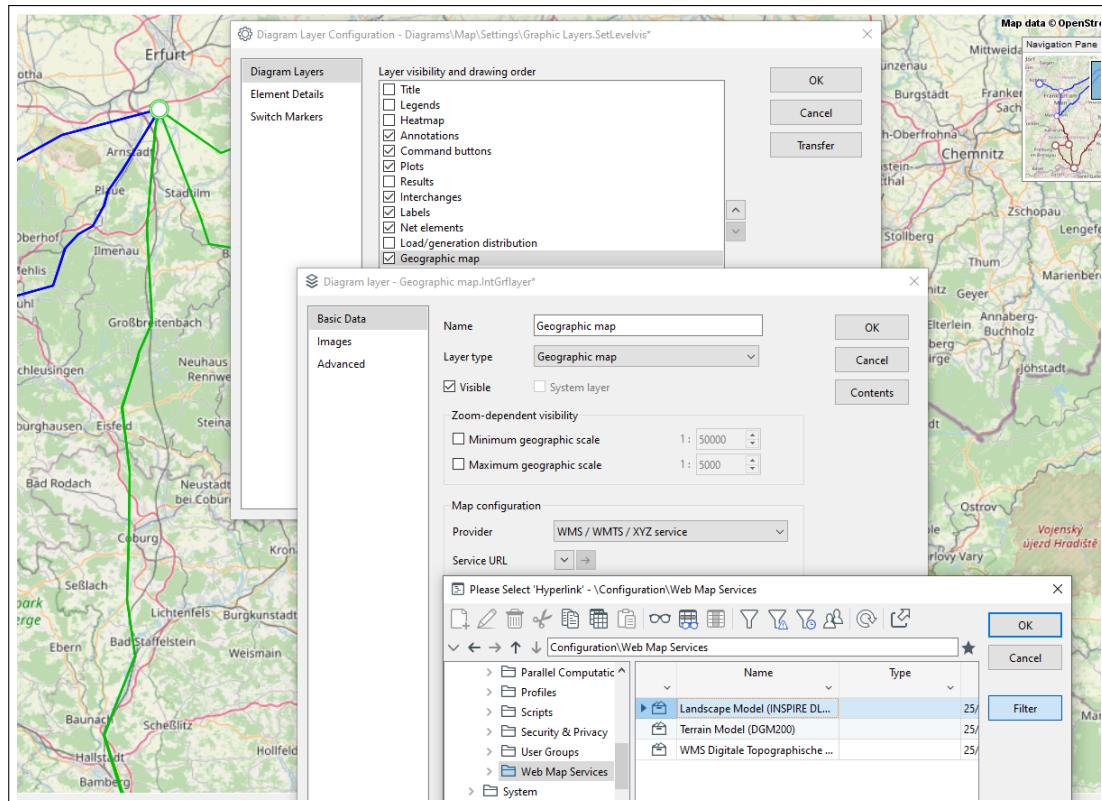


Figure 10.8.5: Background layer configuration

**Note:** A Web Map Services folder can also be created directly under the user or project and used as described above.

### 10.8.2 Using Local Maps

To display background images (e.g. maps) on the geographic diagram, the map provider must be selected as *Local map files* (on the Layers dialog, *Configuration* page). A *File* for reading background images must be selected. This facilitates 'tiling' of multiple images in the background of the GPS graphic if required.

The *File* is simply a text file with semicolon delimited entries, as follows:

Image\_filename; X1; Y1; X2; Y2

Where:

- *Image\_filename* is the name of the image file. If it is not in the same directory as the *File*, it should include the file path.
- *X* is the latitude and *Y* is the longitude.
- $(X_1, Y_1)$  are the bottom-left coordinates of the image.
- $(X_2, Y_2)$  are the top-right coordinates of the image.
- The # symbol can be used to comment-out entries.

**Note:** The figure size is limited to 7 MB.

## 10.9 Graphic Windows Hotkeys

When working with network graphics, many commonly used actions can be carried out using hotkeys or key-and-mouse combinations. These are listed below.

**Note:** Some of the hotkeys below are only available when Freeze Mode is *off*.

Combination	Where/When	Description
<b>Ctrl + -</b>	Single Line Graphic, Block Diagrams, Plots	Zoom out
<b>Ctrl + +</b>	Single Line Graphic, Block Diagrams, Plots	Zoom in
<b>Ctrl + Scrolling</b>	Single Line Graphic, Block Diagrams, Plots	Zoom in/out
<b>Ctrl + Double-click</b>	Busbar system	Open detailed graphic of substation
<b>Ctrl + Tab</b>	Single Line Graphic, Block Diagrams, Plots	Switch between the opened graphic pages
<b>Press Mouse Scroll Wheel + Moving</b>	Single Line Graphic, Block Diagrams, Plots	Panning, Moving the visible part of the graphic
<b>Alt + Area selection</b>		Only textboxes inside the rubber band are marked, no parent objects
<b>Alt + Left-click</b>	Textbox	Textbox and Parent-Object are marked
<b>Ctrl + A</b>		All elements are marked
<b>Ctrl + Shift + A</b>		All elements of the selected class are marked
<b>Ctrl + Alt + Moving</b>	Marked Object	Single Objects from a Busbar system can be moved
<b>Ctrl + Alt + Moving</b>	Block	The stub length of blocks in block diagrams remains when shifting
<b>Ctrl + Alt + Moving</b>	Marked Terminal	Line-Routes will move to the terminal, instead of terminal to the line
<b>Ctrl + Alt + Moving</b>	Marked Node	Symbol of the connected branch element will not be centred
<b>Ctrl + C</b>	Marked Element	Copy Element
<b>Ctrl + V</b>	Single Line Graphic, Block Diagram	Paste Element
<b>Ctrl + L</b>	Single Line Graphic, Block Diagrams	Will open the Define Layer dialog to create a new layer
<b>Ctrl +Left-click</b>	Element	Multiselect elements, all clicked elements are marked
<b>Ctrl + Left-click</b>	Inserting Loads/Generators	Rotate element 90°
<b>Ctrl + Left-click</b>	Inserting Busbars/Terminals	Rotate element 180°
<b>Ctrl + M</b>	Element dialog	Mark Element in the graphic

Combination	Where/When	Description
<b>Ctrl + P</b>		Open Print Dialog
<b>Ctrl + X</b>	Marked Element	Cut
<b>Esc</b>	Connecting Mode	Interrupt the mode
<b>Esc</b>	Inserting Symbol	Interrupt and change to graphic cursor
<b>S + Left-click</b>	Element	Mark only the symbol of the element
<b>S + Moving</b>	Marked Element	Move only the symbol of the element
<b>Shift + Moving</b>	Marked Element	Element can only be moved in the direction of axes
<b>Shift + Moving</b>	Marked Textbox	After rotation, textbox can be aligned in the direction of axes
<b>Tab</b>	Inserting Symbol	Change connection side of symbol
<b>Left-click</b>	Inserting Symbol	Place symbol, press mouse button and move cursor in the direction of rotation to rotate the symbol in this direction (depending on Ortho-Type)

Table 10.9.1: Graphic Window Hotkeys

# Chapter 11

## Data Manager and Network Model Manager

### 11.1 Introduction

The Data Manager and the Network Model Manager are two ways of accessing data within a *PowerFactory* project. Both are opened via icons from the main toolbar, and in each case a browser window is presented to the user.

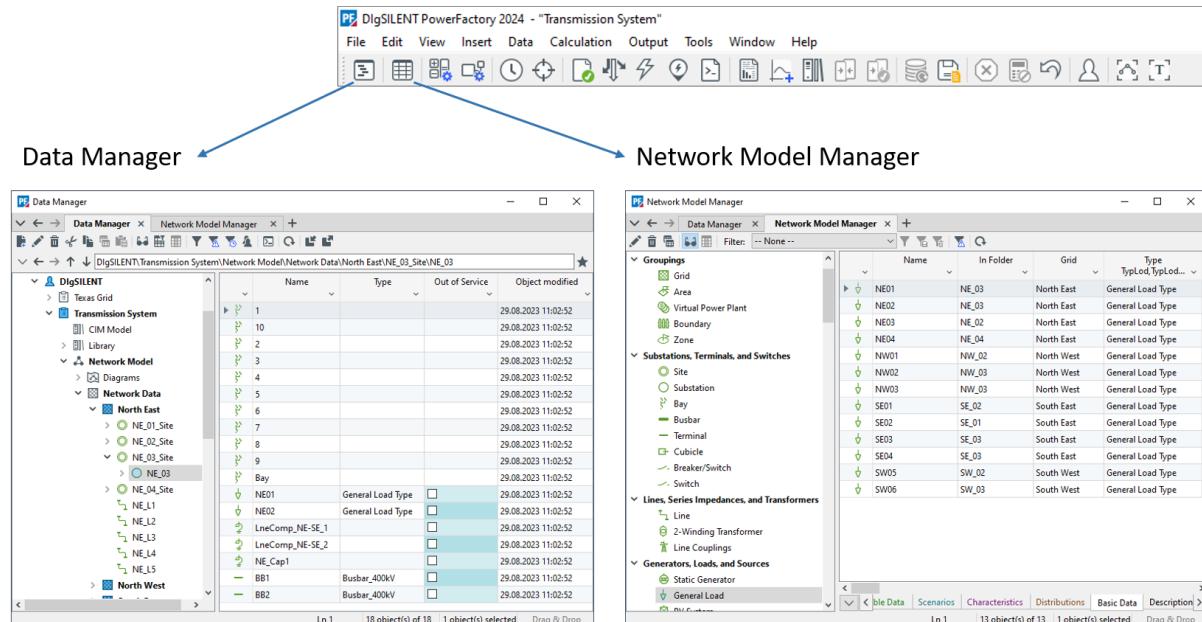


Figure 11.1.1: The Data Manager and the Network Model Manager

The two browsers are similar in appearance, each having a left and right pane and many similar icons in the tool bar. They also share many features and functionalities. However, the information and the way that is presented is different between the two, and users will find that both are essential tools, depending on the task at hand.

- The **Data Manager** enables the user to access all the objects in a database. Inside a project, this is not just network elements but also objects such as study cases, network variations, libraries and scripts. Everything is shown in its hierarchical structure in the left-hand pane, enabling users

to easily see where a particular object is located within this hierarchy. Typically, the Data Manager is used when a network is being created or modified, or users need to look outside the project at other parts of the database.

- The **Network Model Manager** is a browser for all calculation-relevant objects in the currently-active project. This means network elements and types, together with other objects related to these, such as characteristics. In the Network Model Manager, these objects are not shown in a hierarchical arrangement but are sorted by class. This enables the user, for example, to see all lines together, regardless of where they are located in the network or in the project hierarchy. Typically, the Network Model Manager is used to get an overview of network element parameters and for looking at calculation results.

---

**Note:** Elements that are “hidden” (for example created in a network variation expansion stage that is not currently active) will *not* appear in the Network Model Manager, but will be visible as greyed-out objects in the Data Manager.

---

The Data Manager can be opened by clicking on the  icon on the main toolbar.

The Network Model Manager can be opened by clicking on the  icon on the main toolbar.

By default, the *Network Model Manager* and *Data Manager* will open as tabs in the same floating window. This is part of a general logic that is applied to ensure that tabs in floating windows are by default grouped together according to their type. However, as with other tabs, it is easy to move *Network Model Manager* and *Data Manager* tabs into other groups. It is also possible to rename the tabs and to duplicate them.

---

**Note:** By default, the *Data Manager* and *Network Model Manager* can only be opened once, but the legacy behaviour of multiple instances can be allowed by selecting the relevant options on the *Data/Network Model Manager* page of the *User Settings* dialog, as described in Chapter 7 ([User Settings](#)).

---

General information about the use of tab groups in *PowerFactory* can be found in Chapter 4 ([PowerFactory Overview](#)), Section 4.7.3.

The rest of this chapter is divided into three sections:

- Section 11.2 describes the many features and functionality that are common to both the Data Manager and Network Model Manager.
- Section 11.3 describes features that are only available when using the Data Manager.
- Section 11.4 describes features that are only relevant for the Network Model Manager.

## 11.2 Data Manager and Network Model Manager Features

The Data Manager and Network Model Manager have many features in common. This section describes the features and functionality which are the same or similar in the two browsers.

### 11.2.1 Browser window layout

Both the Data Manager and the Network Manager consist of two panes, above which a toolbar gives access to commonly used actions. At the bottom of the window is a status bar.

### 11.2.1.1 Toolbar icons

The icons seen in the Data Manager (DM) and/or Network Model Manager (NMM) toolbars are listed in Table 11.2.1 below. (The list does not include the special options available on the Scenario tab of the Network Model Manager.)

More detail about some of these features can be found in later sections.

Icon	Action	Use	Comment
	New Object	Create a new object	DM only
	Edit Object	Edit the selected object	
	Delete Object	Delete the selected object	
	Cut Object	Cut the selected object prior to pasting it elsewhere	DM only
	Copy Object	Copy the selected object	DM only
	Copy (with column headers)	Copy selected data together with the column headings, so that it can be pasted into a spreadsheet	
	Detail Mode	Select <i>Detail Mode</i> , where only objects of a selected class are displayed, together with all their attributes	
	Detail Mode Class Select	Select the class to be shown in <i>Detail Mode</i>	
	Variable selection	Define which variables should be shown on a Flexible Data Page	
	Find	Search for objects using an existing filter or by creating a new filter	DM only
	Edit Filter	Edit the filter, only active when objects are already being filtered	NMM only
	Save Filter	Save the filter, only active when objects have been temporarily filtered	NMM only
	Delete Filter	Delete a saved filter	NMM only
	Hide out-of-service...	Hide out-of-service objects and objects that are not relevant for calculation	
	Hide inactive objects	Hide inactive objects (e.g. those that become active in a future expansion stage)	DM only
	Show All Users	Show other users in a multi-user database, together with any projects that are shared with the current user	DM only
	Formats and Units	Allow data formats and units to be configured in the Project Settings	Flexible Data Page
	Input Window	Open the Data Manager Input Window	DM only
	Refresh	Update the contents of the browser, for example when a new object has been added to the network	
	Import Data	Import data into <i>PowerFactory</i>	
	Export Data	Export data out of <i>PowerFactory</i>	DM only

Table 11.2.1: Data Manager and Network Model Manager icons

### 11.2.1.2 Status bar

The status bar shows the current status and settings of the Data Manager or Network Model Manager. Some of the fields may be double-clicked in order to change the settings.

The status bar contains the following fields:

- “Pause: on/off” (visible only when an Input Window has been opened in a Data Manager - see Section 11.3.6) shows the status of the message queue in the input window. With pause on, the command interpreter is waiting, which makes it possible to create a command queue. Double-clicking on this field will toggle the setting.
- “Ln N” shows the number of the currently selected row.
- “N object(s) of M” shows the number of elements shown in the browser window and the total number of elements in the current folder.
- “N object(s) Selected:” shows the number of currently selected objects.
- “Drag & Drop: on/off” shows the current drag & drop mode. Double clicking this field will toggle the setting.

### 11.2.2 Selecting objects and general object-specific actions

When a location in the project hierarchy is selected on the left-hand side of the Data Manager, the contents of that location (the “children” objects) will be shown in the right-hand pane. Similarly, when object class is selected on the left-hand side of the Network Model Manager, all elements of that class will be shown on the right-hand side.

In both cases, an object can be selected on the right-hand side by clicking on its icon in the first column. Then various actions can be carried out, such as deleting, copying or showing information about the element. The options available can be accessed by right-clicking on the object and selecting the action from the context menu. Many of the more common actions are also available using icons along the top of the browser window, as described above.

Note that objects that are relevant for calculations are tagged with a check-mark sign following a calculation. Editing one of these objects will reset the calculation results.

### 11.2.3 Detail Mode and browser tabs

The  icon is used to toggle *Detail Mode* on and off.

When *Detail Mode* is off, as is the default for the Data Manager, all objects within a selected location will be shown on the right-hand side, regardless of their class. The data fields displayed will be limited to common fields, such as Name, Type, Out of Service and the Object modified date.

When *Detail Mode* is turned on, as is the default for the Network Model Manager, only objects of the selected class will be listed. The selection of the class can be made using the  icon. As can be seen in Figure 11.2.1 below, once Detail Mode is activated, access to the many object attributes is possible, using the function-specific tabs. If required, the tabs to be shown can be restricted, via the *Functions* page of the User Settings  (see Section 7.6).

Tabs can be selected by clicking directly on the required tab, using the left  and right  arrows, or using the down-arrow  in order to select a tab from the list.

The Flexible Data, Scenarios (Network Model Manager only), Characteristics and Distribution tabs are highlighted by colouring. The colours reflect those selected in the User Settings  (refer to Section 7.8)

for these types of data.

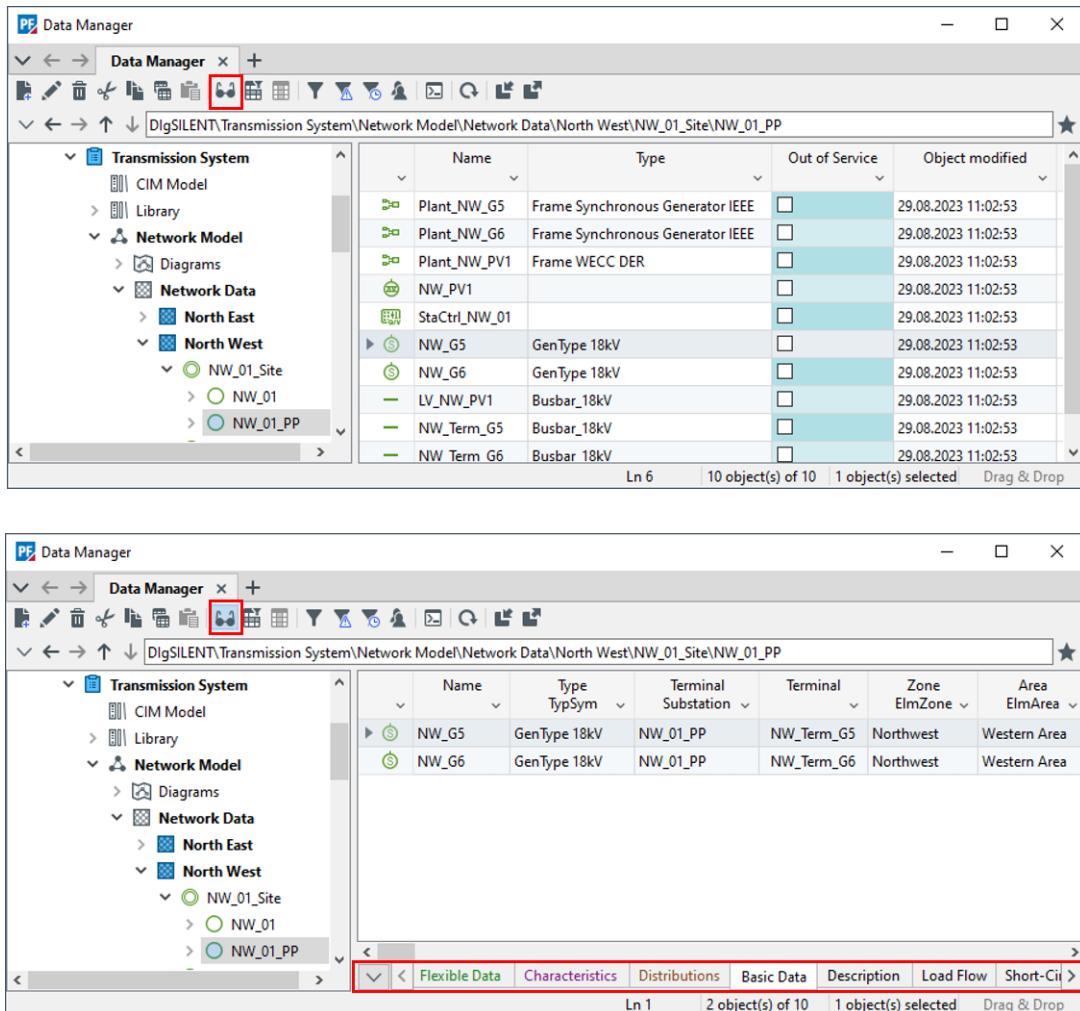


Figure 11.2.1: Data Manager with Detail Mode off (above) and on (below)

#### 11.2.4 Flexible Data Page

When the Data Manager or Network Model Manager is in Detail Mode, page tabs will be seen for each calculation function.

In addition, a Flexible Data page, normally used to display calculation results, allows the user to define a custom set of data to be displayed.

The default format for the calculation results displayed in the Flexible Data page depends on the calculation just executed: Following a load-flow calculation, the default variables for terminals are line-to-line voltage, per unit voltage and voltage angle. Following a short-circuit calculation the default variables are initial short-circuit current, initial short-circuit power, peak current etc. There is also a default selection of attributes that are shown when no calculation has been executed.

Figure 11.2.2 is an example of a Flexible Data page showing load flow calculation results.

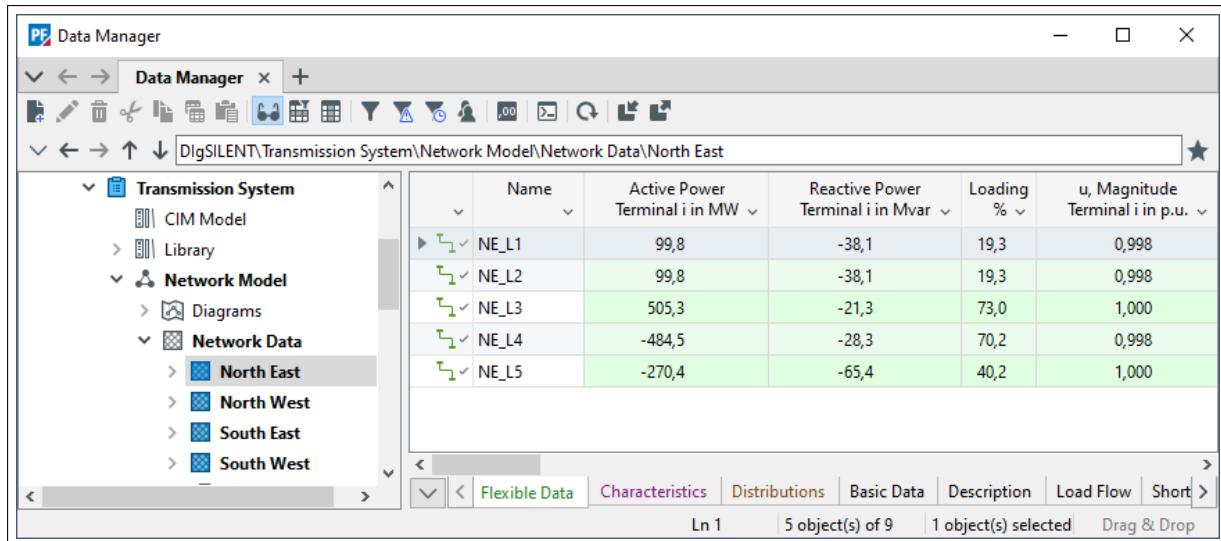


Figure 11.2.2: The Flexible Data Page

#### 11.2.4.1 Customising the Flexible Data page

The variables to be displayed on a Flexible Data page are customised using the variable selection dialog, which is opened by clicking on the icon. The dialog will open showing the variables corresponding to the most recent calculation.

The Variable Selection object is described in Section [19.3: Variable Selection](#).

There are other ways of changing the format of a Flexible Data page:

- **Column order:** The order of the columns (except: *Name*, *In Folder* and *Grid*) can be changed simply by clicking on a column heading and dragging it to a new location.
- **Number format:** For columns that contain numbers, the number format can be by right-clicking on the column header of the variable and selecting *Edit Number Format* .... A new window will appear where the user can define the number representation.
- **Project settings:** The icon can be used to access the *Formats and Units* page of the project settings. Here, settings such as the units and the number of decimal places shown for calculation results can be changed. (See Section [9.1.3](#) for more information.)

#### 11.2.5 Sorting

In the Data Manager or Network Model Manager, the order of the objects can be sorted by clicking on any column heading. This will cause the objects to be sorted alphabetically or numerically as appropriate. Clicking again will reverse the order. Sorting on multiple columns in sequence can also be useful. For example, a user may be interested in the voltage at terminals after running a load flow calculation. If the user sorts on *u, Magnitude p.u* first, and then sorts on *Nom.L-L Volt kV*, the outcome will be that the terminals are sorted by voltage level and within the voltage level sorted by calculated voltage.

#### 11.2.6 Finding objects by name

In the Data Manager or Network Model Manager, it is possible to find a particular object by typing the start of its name. This is done by clicking on the icon of one of the objects in the table and typing the

first letter or first few characters of the name of the required object. Note that if more than one character is typed, these must be done in quick succession.

The characters that are typed must be letters or numbers, and letters should be lower-case (the search is not case-sensitive)

### 11.2.7 Auto-filter functions

Columns within the Data Manager, browser windows or the Network Model Manager can be filtered. This is done using these steps:

- Click on the down arrow in the column header of the table. A window will open.
- A list of all the distinct entries in that column will be shown.
- Option: *Selection*
  - Use the check-boxes to select what should be shown in the table. The others will be hidden.
  - The list that is displayed can also be restricted by typing characters into the search field. Only items containing those characters will be shown.
- Option: *Custom...*
  - In addition to the possibility of selecting existing entries, custom filters can be used by choosing the radio button *Custom...* and confirming with **OK**.
  - A new window will open, in which up to two filters can be combined via an AND or an OR relation. The options are:
    - \* Equals
    - \* Is not equal to
    - \* Contains
    - \* Does not contain
    - \* Starts with
    - \* Does not start with
    - \* Ends with
    - \* Does not end with
    - \* Special
  - Once a criterion has been selected, a drop-down box will appear, from which one of the entries may be chosen or text may be entered.
  - The filter is confirmed by clicking on **OK**.

Filtered columns are indicated by a blue column heading and a filter symbol in the lower right corner of the column header. If the mouse is hovered over the heading of a filtered column, the applied filter settings will be shown.

When auto filters are used in the Data Manager, they are temporary; they will be lost if a different location is selected on the left-hand side of the browser.

However, in the Network Model Manager, the filter or filters that are applied form the *Current Working Filter*, which can be saved for future use. This is described further in Section [11.4.2](#).

### 11.2.8 Editing objects from a Data Manager or Network Model Manager

#### 11.2.8.1 Editing objects via individual object dialogs

Right-clicking on an object and selecting *Edit*, or simply double-clicking on its icon, will open the Edit dialog for that object. Edits can then be made, saved by clicking on **OK**, or discarded by clicking on **Cancel**.

### 11.2.8.2 Editing object attributes directly in the browser

Some object attributes can also be changed directly from the Data Manager or Network Model Manager, without the need to open the object dialog. Editing directly from the browser like this does open up the possibility of changing an attribute for multiple objects at the same time. However, the options for editing are dependent upon the type of field to be changed. Some general points to note are:

- When an object is selected in the browser, a small black triangle indicates this in the first column (where the icons are shown). When one of the attributes for this object is being edited, this marker changes to an Edit marker . The actual change is not committed to the database until the user clicks away from the object and this Edit marker disappears. While the Edit marker is still visible, erroneous edits can be cancelled using the Escape key.
- Copying and pasting data within editable attribute fields is generally possible.
- Right-clicking on a field will bring up edit options for that field.
- For some (Boolean) attributes, a double-click can be used for toggling the value. For other fields, for example where a number is to be entered, double-clicking on the field makes it editable, so that the contents can be changed.
- Some fields are actually references to other objects. In these cases, double-clicking on the field will open the dialog for the referenced object, so the right-click context menu can be used instead if the intention is to change the reference.

### 11.2.8.3 The *Modify Value(s)...* feature

One of the options offered in the context menu when editing an object attribute is *Modify Value(s)...*. This can be used on a single attribute cell, but is normally used on a selection of the same attribute for multiple objects.

As an example, the Derating Factor is to be changed for several lines at once. As shown in Figure 11.2.3, the Derating Factor field is multi-selected for the six lines, then right-click is used to access the context menu.

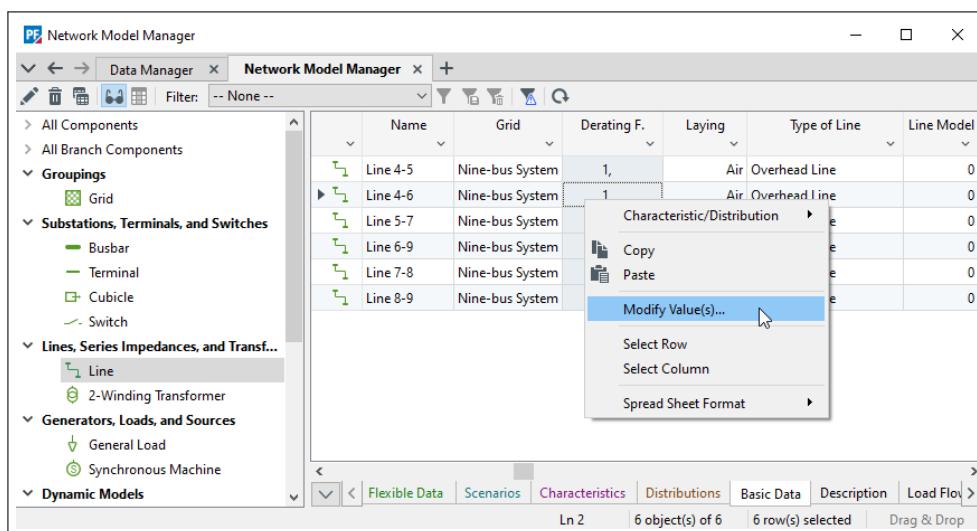


Figure 11.2.3: Modify values option

When *Modify Value(s)...* is selected, the dialog opens and three options are presented, as shown in Figure 11.2.4. The editable field shown at the bottom of the dialog depends on the option selected.

- **Relative:** change the values using a percentage specified in the *Scale Factor* field.

- **Relative to Sum:** scale the values, each by the same factor, so that their sum will be the value specified in the *New Sum Value* field.
- **Absolute:** Set all the selected parameter fields to the same value specified in the *Value* field.

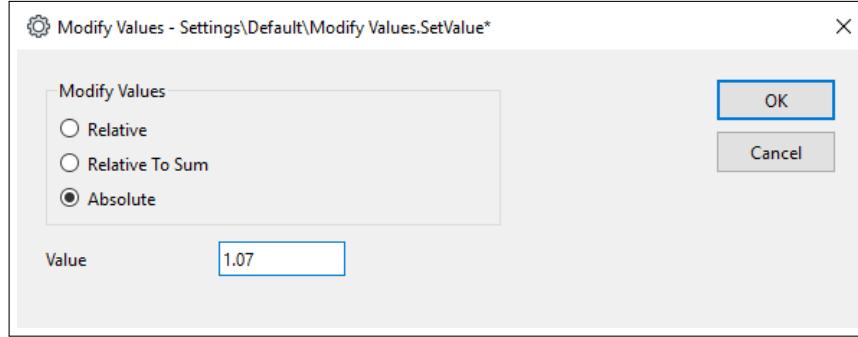


Figure 11.2.4: Modify values dialog

### 11.2.9 Other context menu options

This section lists some of the other useful options found in the context menu when right-clicking on an object in the Data Manager or Network Model Manager.

**Show All References:** produces the list of objects that have links, or references (plus the location of the linked object), to the selected object. The list is written to the output window. This option can be used, for example, to obtain a list of elements that all use the same type. The listed object names have hyperlinks to the objects themselves.

**Select All:** selects all objects in the browser.

**Mark in Graphic:** marks the highlighted object(s) in one or more single line diagram. This feature can be used to identify an object.

**Goto Busbar:** (Data Manager) opens the folder in the database browser that holds the busbar to which the currently selected element is connected. If the element is connected to more than one busbar, a list of possible busbars is shown first.

**Goto Connected Element:** (Data Manager) opens the folder in the database browser that holds the element that is connected to the currently selected element. If there is more than one connected element, which is normally the case for busbars, a list of connected elements is shown first.

**Calculation:** opens a second menu with several calculations which can be started, based on the currently selected objects. A short-circuit calculation, for example, will be executed with faults positioned at the selected objects, if possible. If more than one possible fault location exists for the currently selected object, which is normally the case for station folders, a short-circuit calculation for all possible fault locations is made.

### 11.2.10 Display of multi-dimensional attributes

Some object attributes are not just single values but consist of a vector (1-dimensional array) or matrix (2-dimensional array). Such attributes can be viewed in an additional window by double clicking on the attribute cell. Figure 11.2.5 shows an example of such a multidimensional attribute.

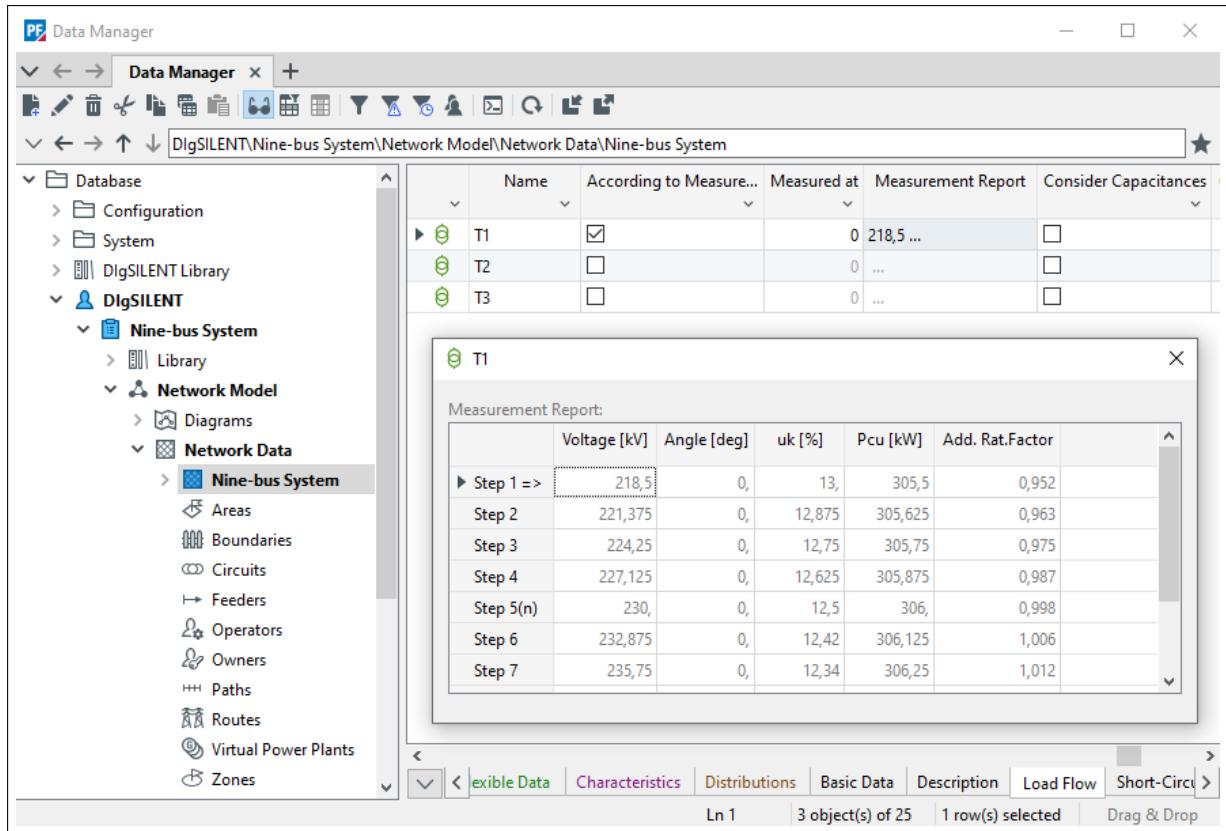


Figure 11.2.5: Displaying multidimensional attributes (arrays) in the Data Manager

For the particular case of matrices of complex numbers, as shown in Figure 11.2.6, it is also possible to select which of the variables should be displayed in the additional window.

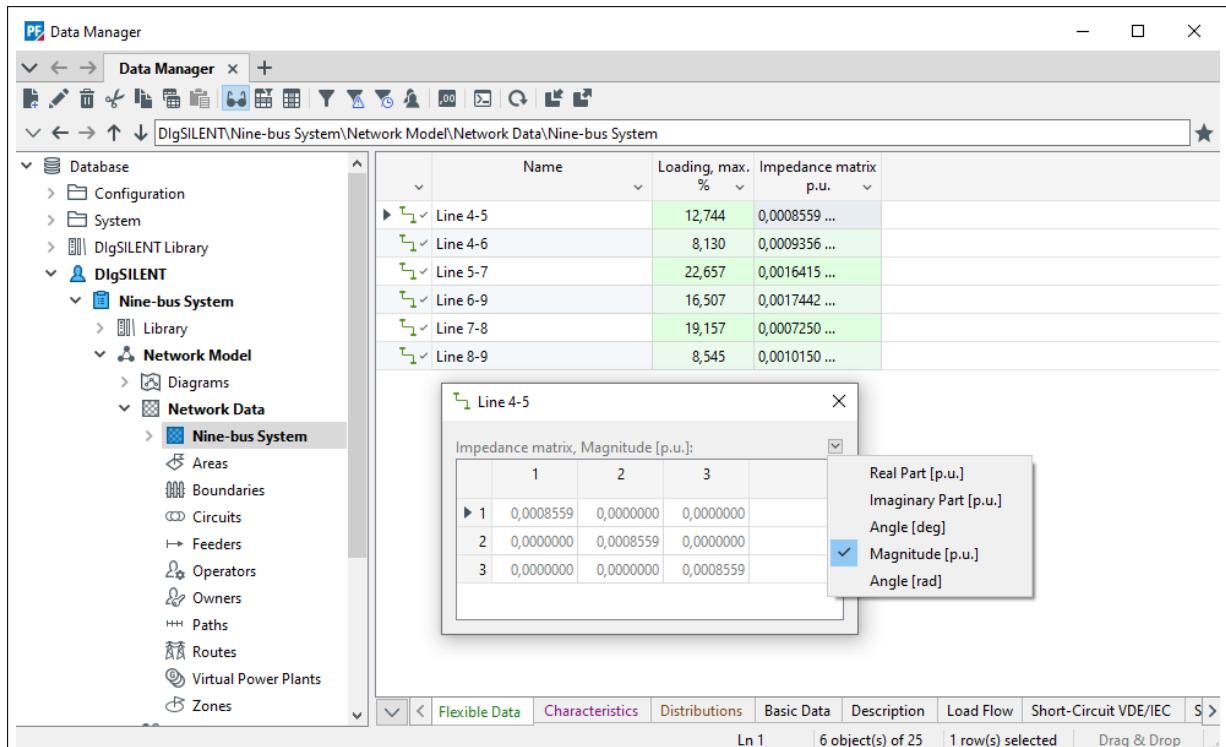


Figure 11.2.6: Displaying matrices of complex numbers in the Data Manager

### 11.2.11 Exporting and importing data in spreadsheet format

The Data Manager or Network Model Manager browser looks and acts like a spreadsheet program as far as creating and editing power system objects is concerned. To enable and simplify the use of power system element data which is stored in spreadsheet programs such as the Microsoft Excel or the Lotus 123 programs, the browsers offer “Spreadsheet Format” import and export facilities.

#### 11.2.11.1 Export to Spreadsheet Programs (e. g. MS EXCEL)

All data visible in the data browser may be exported as it is. The export format is such that most common spreadsheet programs can read in the data directly (space-separated ASCII). Exporting data is carried out as follows.

- Select a range of data in the data browser. Such a range may contain more than one column and more than one row, but the cells must be contiguous.
- Right-click the selected range.
- Now you have different options:
  - If you want to copy the content of the marked cells only, simply select *Copy* from the context menu.
  - If you want to copy the content of the marked cells together with a description header, select the *Spread Sheet Format* option. This opens a second menu which offers the choice between writing the Spreadsheet export to a file (*Write to File*), or to put it on the Windows Clipboard (*Copy (with column headers)*). See Figure 11.2.7.
  - Alternatively, the  icon in the toolbar can be used to do *Spread Sheet Format → Copy (with column headers)*.
- The exported data can now be pasted into a spreadsheet program.
- The cell colourings are kept when the data is pasted into an Excel spreadsheet.

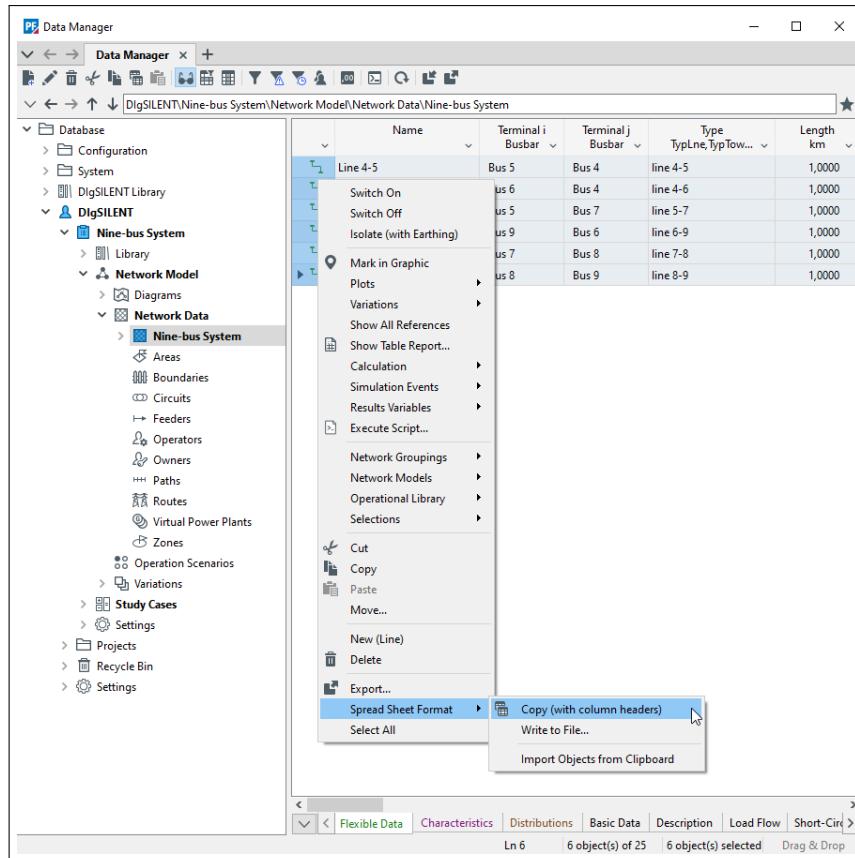


Figure 11.2.7: Exporting a range of data

### 11.2.11.2 Import from Spreadsheet Programs (e. g. MS EXCEL)

There are two methods available for importing data from a spreadsheet program.

The first method is used to change parameters of existing objects via a direct import of “anonymous” numerical data, with the term “anonymous” indicating that the imported data has no parameter description. The size of the imported value range and the required data are checked. Importing invalid values (e.g. a power factor of 1.56) will result in an error message.

The second method can be used to create new objects (or replace whole objects) by importing all the data from a spreadsheet.

#### Spreadsheet Import of Values

The import of values (anonymous variables), i.e. cells of a table, is explained by the following example:

In Figure 11.2.8, a range of active and reactive power values is copied in a spreadsheet program. In Figure 11.2.9, this range is pasted to the corresponding fields of 6 load objects by right-clicking the upper leftmost field that is to be overwritten.

In contrast to the import of whole objects, the anonymous import of data does not need a parameter description. This would complicate the import of complete objects, as the user would have to enter all parameters in the correct order.

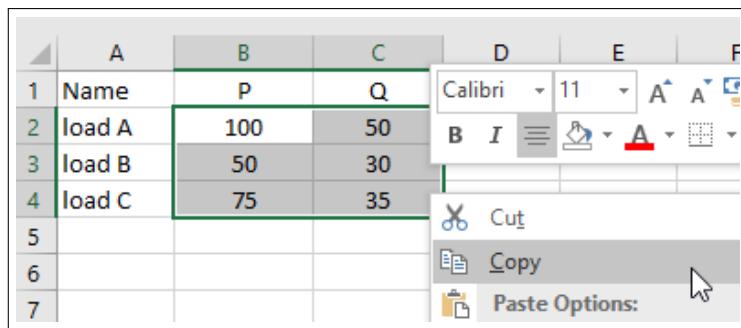


Figure 11.2.8: Copying a range of spreadsheet data

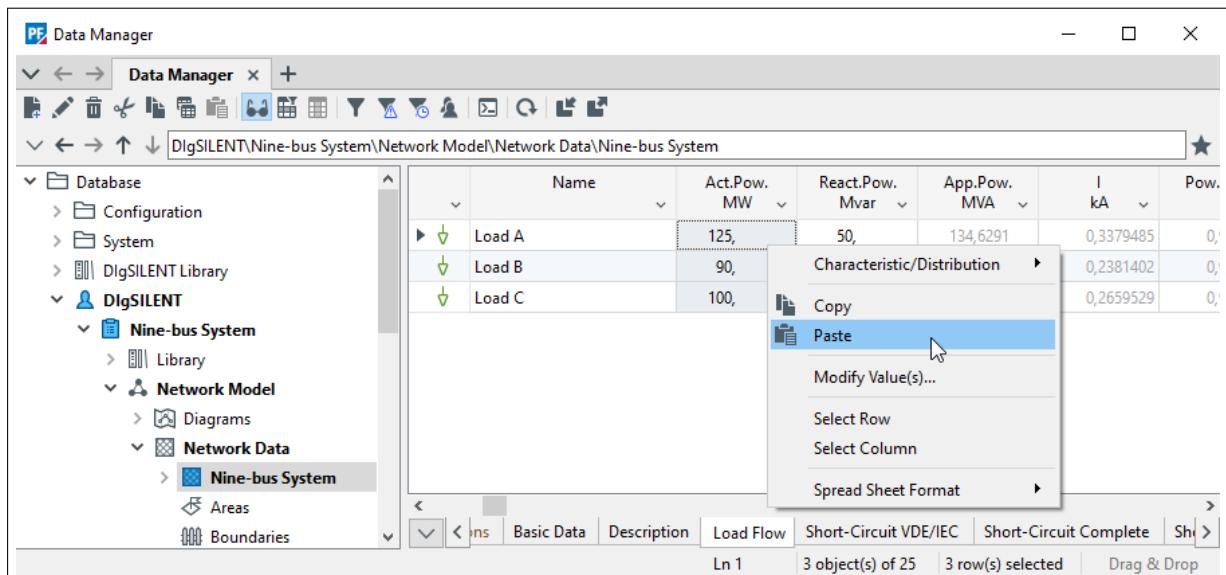


Figure 11.2.9: Pasting spreadsheet data from clipboard

### Spreadsheet Import of Objects and Parameters

With this kind of import, it is possible to import whole objects (in contrast to the import of pure values described above). The object import uses a header line with the parameter names (which is necessary in addition to the cells with the pure values). This header must have the following structure:

- The first header must be the class name of the listed objects.
- The following headers must state a correct parameter name.

This is shown in Figure 11.2.10.

A	B	C	D	E
1 Class Name	Param. Name 1	Param. Name 2		Param. Name N
2 Name1	value	value	...	value
3 Name2	value	value	...	value
4 ...				
5 NameM	value	value	...	value

Figure 11.2.10: Excel required format

Figure 11.2.11 shows an example of valid spreadsheet data of some line types and some 2-winding transformer types.

	A	B	C	D	E	F	G	H	I
3	<b>Line Types</b>								
4	TypLne	uline	sline	frnom	aohl_	rline	xline	rlin0	xline0
5	NKBA 3x120/70 1.00 kV	1	0.32	50	cab	0.157	0.083	0.261	0.97
6	NKBA 3X 25/ 16 1kV-TT	1	0.133	50	cab	0.724	0.092	0.292	1.672
7	NAYY 4x95 1.00 kV	1	0.211	50	cab	0.321	0.082	0.261	1.284
8	NAYY 4x70 1.00 kV	1	0.176	50	cab	0.444	0.082	0.261	1.776
9	NAYCWY 4x95/50 1.00 kV	1	0.211	50	cab	0.321	0.082	0.261	1.284
10	NAYCWY 4x70/35 1.00 kV	1	0.176	50	cab	0.444	0.082	0.261	1.776
11	NAYCWY 4x50/25 1.00 kV	1	0.142	50	cab	0.642	0.083	0.264	2.568
12									
13	<b>2-Winding Transformer Types</b>								
14	TypTr2	strn	utrn_h	utrn_l	uktr	pcutr	uk0tr		
15	0.4MVA 110/10kV	0.4	110	10	12	6	14		
16	10MVA 110/10kV	10	110	10	12	6	14		
17	TR_LV- 800/10	0.8	10	0.4	4	7	6		
18	TR_MV- 40/10	40	110	10	12	150	14		

Figure 11.2.11: Example of valid spreadsheet data

The data are imported into *PowerFactory* as follows:

- Select the header line and one or more objects lines.
- Copy the selection. See Figure 11.2.12 for example.
- Right-click the folder browser in the Data Manager to which the objects are to be imported. Select *Spread Sheet Format → Import Objects from Clipboard*, as shown in Figure 11.2.13.

13	2-Winding Transformer Types								
14	TypTr2	strn	utrn_h	utrn_l	uktr	pcutr			
15	0.4MVA 110/10kV	0.4	110	10	12	6			
16	10MVA 110/10kV	10	110	10	12	6			
17	TR_LV- 800/10	0.8	10	0.4	4	7			
18	TR_MV- 40/10	40	110	10	12	150			
19									
20									

Figure 11.2.12: Selecting object data in spreadsheet

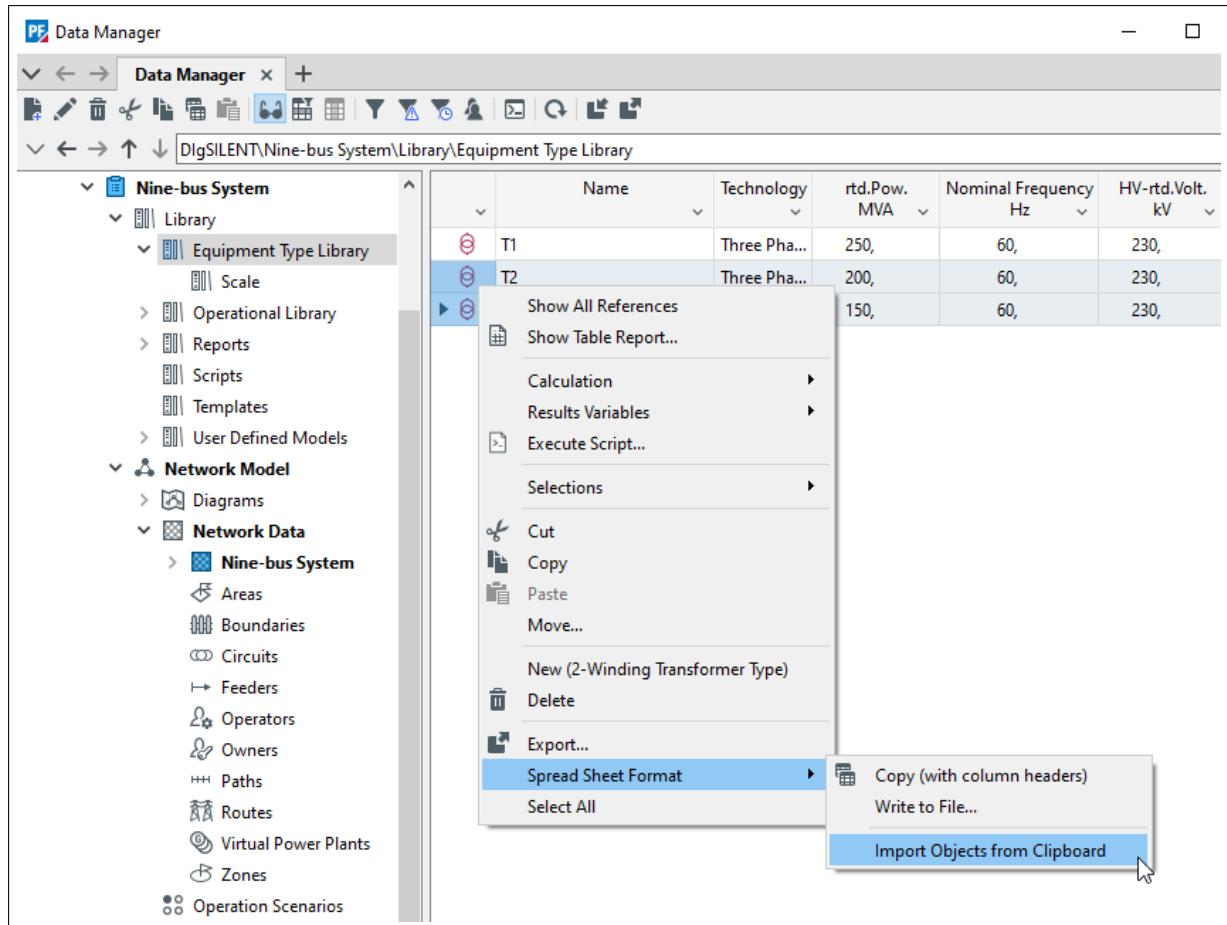


Figure 11.2.13: Importing objects from clipboard

The result of the object import depend on whether or not objects of the imported class and with the imported names already exist or not in the database folder. In the example of Figure 11.2.14, none of the imported objects existed in the database and so all were created. The example shows the database in detail mode.

Name	Technology	rtd.Pow. MVA	Nominal Frequency Hz	HV-rtd.Volt. kV	LV-Rtd.Volt. kV
T1	Three Phase T...	250,	60,	230,	16,5
T2	Three Phase T...	200,	60,	230,	18,
T3	Three Phase T...	150,	60,	230,	13,8

Figure 11.2.14: Result of spreadsheet object import

**Note:** New objects are created in the *PowerFactory* database folder only when no object of the imported class and with the imported name is found in that folder. If such an object is found, its data will be overwritten by the imported data.

### Remarks

- **Object Names:** Object names may not contain any of the characters:  
\*? = " , \ ~ |
- **Default Data:** When an imported object is created, the imported data is used to overwrite the corresponding default data. All parameters that are not imported will keep their default value.
- **Units:** The spreadsheet values are imported without units. No conversion from MW to kW, for example, will be possible. All spreadsheet values therefore have to be in the same units as used by *PowerFactory*.

## 11.3 Data Manager Features

This section describes functionality which is only available in the Data Manager.

### 11.3.1 Address bar

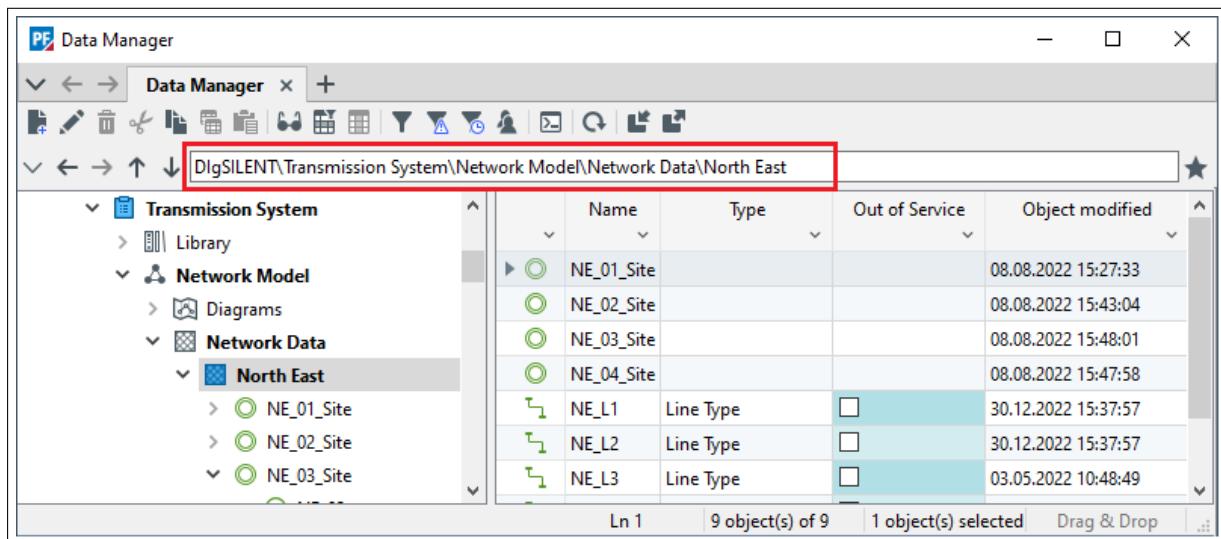


Figure 11.3.1: Data Manager address bar

The Data Manager address bar (see Figure 11.3.1) displays the file-path of the location currently selected on the left-hand side of the Data Manager, but also offers a number of features to help the user easily navigate the data structure.

#### Autocomplete

If the user starts typing a location in the address bar, matching locations will be listed, from which the user can select the one that is wanted.

#### Switch to...

On the left-hand side of the address bar there are left- and right-arrows, which can be used to switch to the previous or next selected folder, whilst the up-arrow is used to switch to the parent folder.

#### Favourites

A selected location can be added to (or removed from) the “favourites” list by clicking on the star-icon to the right of the address bar.

### Selecting common locations

On the far left-hand side of the address bar there is a down-arrow, which gives access to several features:

- **History:** This lists the locations viewed since the Data Manager was opened.
- **Favourites:** This presents the list of favourite locations
- **Current user**
- **Active project**
- **Active study case**

### 11.3.2 Navigating the database hierarchy

There are several ways to “walk” up and down the database tree:

- Use the mouse: all folders that have a small arrow  next to them may be expanded by double-clicking on the folder, or by single clicking on the arrow.
- Use the keyboard: the arrow keys are used to walk up and down the tree and to open or close folders (left and right arrows). The **Page Up** and **Page Down** keys jump up and down the tree in big steps and the - and + keys may also be used to open or close folders.
- Use the toolbar in combination with the browser window. Double-click objects in the browser to open the corresponding object. This could result in opening a folder, in the case of a common or case folder, or editing the object dialog for an object. Once again, the action resulting from your input depends on where the input has occurred (left or right side of the Data Manager).
- The buttons *Switch to parent folder* () and *Step into folder* () on the Data Manager toolbar can be used to move up and down the database tree.

### 11.3.3 Searching and filtering

Advanced filtering capability is provided with the *Find* function . A filter is normally defined to find a group of objects, rather than individual objects (although the latter is also possible). Advanced search criteria can be defined, e.g. lines with a length in the range 1 km to 2.2 km, or synchronous machines with a rating greater than 500 MW etc.

Clicking on the *Find* () in the Data Manager allows the user to apply a predefined filter or to define a new filter, called “General filter”. If a new filter is defined, the database folder that will be searched can be defined.

General Filters defined by the user are objects stored in the folder *Settings\Filters* within the project.

The options in the General Filter dialog window are:

**Name:** Name of filter.

**Object filter:** This field defines either the part of the search definition or all of it, and is optional.

Examples are as follows:

- **\*.ElmSym:** Include element objects of the class synchronous machines.
- **\*.TypSym:** Include type objects of the class synchronous machines.
- **Lahney.\*:** Include all objects with the name Lahney.
- **Lahney.Elm\*:** Include all element objects with the name Lahney.
- **D\*.ElmLod:** Include all load element objects whose names start with D.

- A drop down list providing various object classes can be accessed with .

**Look in:** This field is available if a filter is defined within the Data Manager. It allows the user to specify the folder in the database that will be searched. The selection is not saved as part of the filter; if the filter is later re-used, it will be set to the currently selected location (folder).

#### Check boxes:

- *Include Subfolders*: will search the subfolders of the root folder specified as well as the root folder itself. The search can be stopped at the matching folder.
- *Relevant Objects for Calculation*: will include only those objects considered by the active study case (if no study case is active, the search is meaningless and no search results will be returned).
- *Interconnecting Branches*: will search for branch elements that interconnect grids.

The **OK** button will close the search dialog, but save the filter object to the *Settings\Filters* folder. This makes it available for further use. The **Cancel** button will close the dialog without saving the changes. This button is useful if a search filter will only be used once. The **Apply** button starts the actual search. It will scan the relevant folders and will build a list of all objects that match the search criteria.

Once the search is complete, a list of results is returned in the form of a new data browser window. From this browser, the returned objects can be marked, changed, deleted, copied, moved, etc.

Advanced search options allow more sophisticated expressions as search criteria. These are specified in the *Advanced* page of the General Filter dialog (Figure 11.3.2). The filter criterion is defined in terms of a logical expression, making use of parameter names. Objects will be included in the data browser if, for their parameters, the logical expression is determined to be true. An example of a logical expression is *dline > 0.7*. The variable *dline* refers to the length of a line, and the effect of such a filter criterion is to limit the data in the browser to lines having a length exceeding 0.7 km. The logical expressions can be expanded to include other relations (e.g. *>=*), standard functions (e.g. **sin()**), and logical operators (e.g. **.and.**).

---

**Note:** Parameter names can be object properties or results. The parameter names for object properties are found, for example, by letting the mouse pointer hover over an input field in an object's dialog window. Parameter names for result variables are found from variable selections, which are described in Section 19.3 Variable Selection.

---

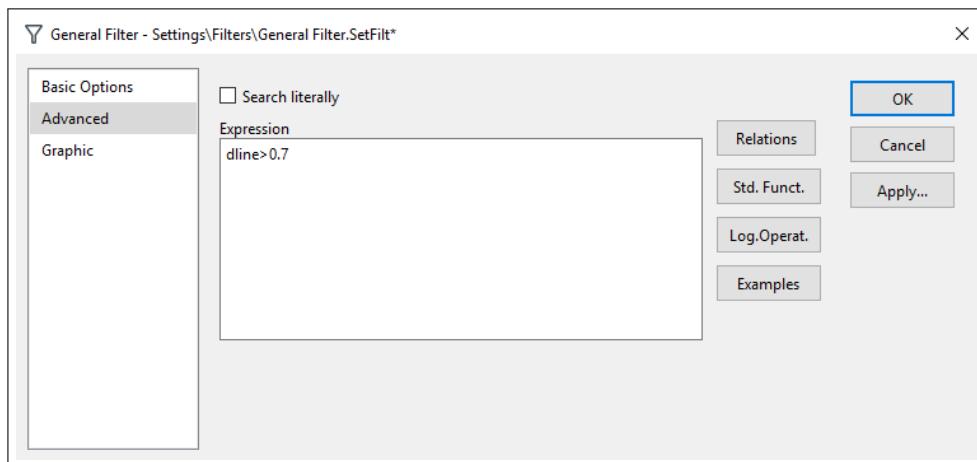


Figure 11.3.2: Filter dialog - Advanced page

**Search literally** is used to search for user defined strings “inside” parameter fields. The user can specify if the search is done in a specific parameter; if the field *in Parameter* is left blank, all parameter fields will be searched for this string.

As stated before, the objects matching the filter criteria are displayed in a data browser. They may also be highlighted in the graphic using the “Colour representation” function described in Chapter 10 ([Network Graphics](#)). The colour to be used in this case can be specified under the page *Graphic* of the General Filter dialog.

---

**Note:** New filters are saved to the *Settings\Filters* folder in the project and are available for use directly, using the right-click context menu. To use a saved filter on a grid, for example: right-click on the grid and *Find → Local Filters → Filter Name* (e.g. Lines longer than 700 m). Then click on Apply to execute the search.

---

## 11.3.4 Adding, deleting and moving objects

### 11.3.4.1 Adding new objects

Generally, new network components are added to the database via the graphical user interface (see Section 12.2: Defining Network Models with the Graphical Editor), such as when a line is drawn between two nodes, creating not only the graphical object but also the corresponding element data in the relevant grid folder. However, users may also create new objects “manually” in the database, from the Data Manager.

Certain new folders and objects may be created by right-clicking on folders in the Data Manager. A context menu is presented, offering a choice of objects to be created that will “fit” the selected folder. For example, right-clicking a grid folder will allow the creation (under the *New* menu) of a Branch, a Substation, a Site or a Folder object. The new object will be created in the folder that was selected prior to the new object button being pressed. This folder is said to have the “focus” for the requested action. This means that some objects may not be possible to create since the focused folder may not be suited to hold that object.

For instance: A synchronous machine should not go into a line folder. A line folder should contain only line routes, line sections and cubicles. The cubicles in their turn should contain only switches or protection elements.

To access the whole range of objects that may be created, the *New Object* icon can be used (⊕). This is found the Data Manager toolbar and presents the dialog shown in Figure 11.3.3.

The New Object dialog displays only the objects that can be added, which depends on the type or class of the originally selected folder.

To simplify the selection of the new objects, a filter is used to sort the object list. In the filter field, it is possible to type the either the name of the new element (e.g. three-winding transformer) or its class (e.g. *ElmTr3*).

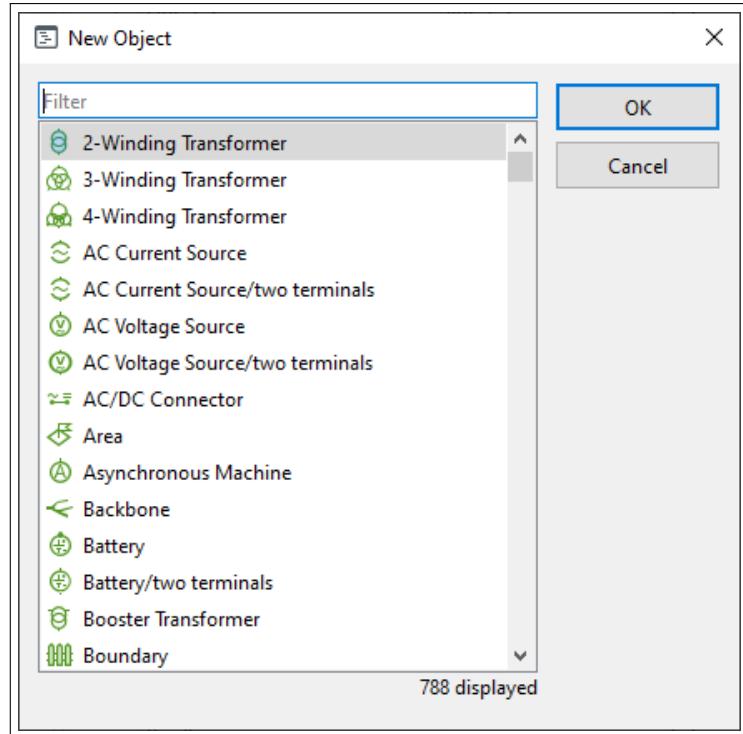


Figure 11.3.3: The New Object dialog

After the selection for a new object has been confirmed, the new object dialog will close, the new object will be inserted into the database and the edit dialog for the new object will open. If this dialog is closed by pressing the **Cancel** button, the whole action of inserting the new object will be cancelled: the newly created object will be deleted from the active folder. The dialog for the new object may now be edited and the **OK** button pressed to save the object to the database.

As any other object, folders can be created either by using the context menu or by using the icon. Common folders (*IntFolder* objects) may have an owner name entered, for documentation or organisational purposes. In this way it should be clear who has created the data. Descriptions may also be added. An existing folder may be edited by using the *Edit* icon on the toolbar or by using right-click.

The folder may be a “Common” or “Library” folder. These attributes can be changed in the edit-folder dialog. These settings have the following meaning:

- Common folders are used for storing non-type objects: electrical elements, command objects, settings, projects, etc.
- Type folders are used as “libraries” for type objects.

#### 11.3.4.2 Deleting objects

A folder or object which is selected may be deleted by pressing the **Delete** key on the keyboard, by clicking the icon on the toolbar of the Data Manager, or by doing right-click, *Delete*.

When deleting an object on the Data Manager or in the Single Line diagram, this object will be deleted immediately from the database. Only the Undo button or **Ctrl-Z** can restore the element and its references to the original location.

Because most power system objects that are stored in the database are interconnected through a network topology or through type-element relationships, deleting objects can cause anomalies in the

database consistency. For this reason, the user will be asked to confirm the deletion, unless this option has been suppressed via the *Data/Network Model Manager* page of the User Settings  (see Section 7.2).

#### 11.3.4.3 Cutting and pasting, coping and moving objects

##### Cut, Copy and Paste

Cutting, copying and pasting may be achieved in different ways:

1. By using the Data Manager tool bar buttons.
2. By using the normal MS Windows shortcuts:
  - **Ctrl-X** will cut a selection,
  - **Ctrl-C** will copy it,
  - **Ctrl-V** will paste the selection to the active folder.

Cutting a selection will colour the item-icons grey. The cut objects will remain in their current folder until they are pasted. A cut-and-paste is exactly the same as moving the object, using the context menu. All references to objects that are being moved will be updated. A cut-and-paste operation can be cancelled by pressing the **Ctrl-C** key after the **Ctrl-X** key has been pressed.

3. By using the context menu. This menu offers *Cut*, *Copy* and *Move* options. The *Move* option will open a dialog in which the target folder can be selected. When the selected objects have been cut or copied, the context menu will then show *Paste*, *Paste Shortcut* and *Paste Data* options.
  - **Paste** will paste the selection to the focused folder.
  - **Paste Shortcut** will not paste the copied objects, but will create shortcuts to these objects. A shortcut object acts like a normal object. Changes made to the shortcut object will change the original object. All other shortcuts to this original object will reflect these changes immediately.
  - **Paste Data** is only be available when just one object is copied, and when the selected target object is the same kind of object as the copied one. In that case, Paste Data will paste all data from the copied object into the target object. This will make the two objects identical, except for the name and the connections.
4. By dragging selected objects to another folder. The “Drag & Drop” option must be enabled first by double-clicking the “Drag & Drop” field in the Data Manager status bar (if it is greyed out). When the Drag & Drop option is on, it is possible to copy or move single objects by selecting them and dragging them to another target/destination folder, either in the database tree or in the database browser window.

---

**Note:** When dragging and dropping, a COPY of the object will be made (instead of moving it) if the **Ctrl** key is held down when releasing the mouse button at the destination folder.

---

#### 11.3.5 Data import and export

Data can be exported from *PowerFactory*, or imported into *PowerFactory*, using the  and  icons in the Data Manager. Examples of data to be exported or imported include whole *PowerFactory* projects, DPL scripts, or folders of data to be shared with other users.

##### Data export

A selected part of the database can be written to a \*.pdf or \*.pdfx file using the button **Export Data...** . A “File Save” dialog will appear, where a filepath is selected and a filename specified.

An alternative is to right-click on the folder or object to be exported and select the option *Export...*

The exported part of the database may be a complete project, a library, or a specific object in the browser window. Exporting a folder (i.e a project, grid, library, etc.) will export the complete content of that folder, including subfolders, models, settings, single line graphics, etc.

It is even possible to export a complete user account. However, only the Administrator is able to **import** a user-account. It is possible to export data from another user account, or even to export another user account completely. However, only the visible (shared) data will be exported.

### Data Import

Exported data can be re-imported or imported into another database by selecting the target location on the left-hand side of the Data Manager and clicking on the *Import Data...*  button. This will open a “File Open” dialog, where the \*.pdf or \*.pfds data-file can be selected.

The \*.pdf or \*.pfds file will be analysed and error messages will be displayed when the file is not a genuine *PowerFactory* data file, or if it is corrupted. If the file format has been found to be correct, a dialog will appear which shows the data and version of the file. The default target folder is also shown, but another target folder can be selected by clicking on the down-arrow ().

When exporting and importing projects and other data, it is important to understand what is included in the exported file, and what is not. If a *PowerFactory* project has references to other objects external to the project, it can cause problems if this is not correctly handled.

To understand the export/import process in more detail, please see Chapter 9 (Basic Project Definition), Section 9.1.5

## 11.3.6 Input Window

The input window is for the more experienced users of DlgSILENT *PowerFactory*. It is closed by default, but can be opened via the  icon. Almost all commands that are available in *PowerFactory* through the menu bars, pop-up menus, icons, buttons, etc., may also be entered directly into the input window, using the *PowerFactory* commands.

The contents of the input window can be saved to file, and commands can be read back into the window for execution.

*PowerFactory* also has special command objects which carry one single command line and which are normally used to execute commands. In this way, complex commands can be saved in the same folder as the power system for which they were configured.

### 11.3.6.1 Input Window Commands

In principle, everything that can be done in DlgSILENT *PowerFactory* can be done from the command line in the input window. This includes creating objects, setting parameters, performing load-flow or short-circuit calculations.

Some commands that are available are typically meant for command line use or for batch commands. These commands are rarely used in another context and are therefore listed here as “command line commands”, although they do not principally differ from any other command.

**Cd Command** Moves around in the database tree by opening another folder at a relative position from the currently open folder.

Example:

```
cd...\\gridB\\Load1
```

**Cls Command** Clears the output or input window.

```
cls/outclears output window  
cls/inpclears input window completely  
cls/inp/doneclears only previously executed commands  
.../y asks for confirmation
```

**Dir Command** Displays the contents of a folder.

Example:

```
dir Study Case
```

**Ed Command** Pops up the dialog of a default command, i.e. "Ldf", "shc", etc.

Example:

```
ed ldf
```

**Exit Command** Queries or sets a variable.

Example:

```
man/set obj=Load_1.elmlod variable=plini value=0.2
```

**Pause Command** Interrupts the execution of the command pipe until a next pause command is executed.

**Pr Command** Prints either the contents of the output window or the currently active graphics window.

**Rd Command** Opens and reads a file.

**Stop Command** Stops the running calculation.

**Wr Command** Writes to a file.

## 11.4 Network Model Manager Features

In this section, some additional features that are only found in the Network Model Manager are described.

### 11.4.1 Object classification

In the Network Model Manager, objects are grouped according to class. On the left hand side of the Network Model Manager window the classes of all calculation relevant objects are displayed with their names and symbols. To give structure to the list, they are sorted into groups, such as *Substations*, *Terminals and Switches* or *Lines, Series Impedances and Transformers*. By double-clicking on the group's name, its contents can be hidden or shown. If one of the classes is selected, all calculation relevant objects will be listed on the right side of the browser window, and *Detail Mode*  is automatically activated.

Figure 11.4.1 shows the Network Model Manager window, where some of the groups are collapsed and the class *Busbar* is selected.

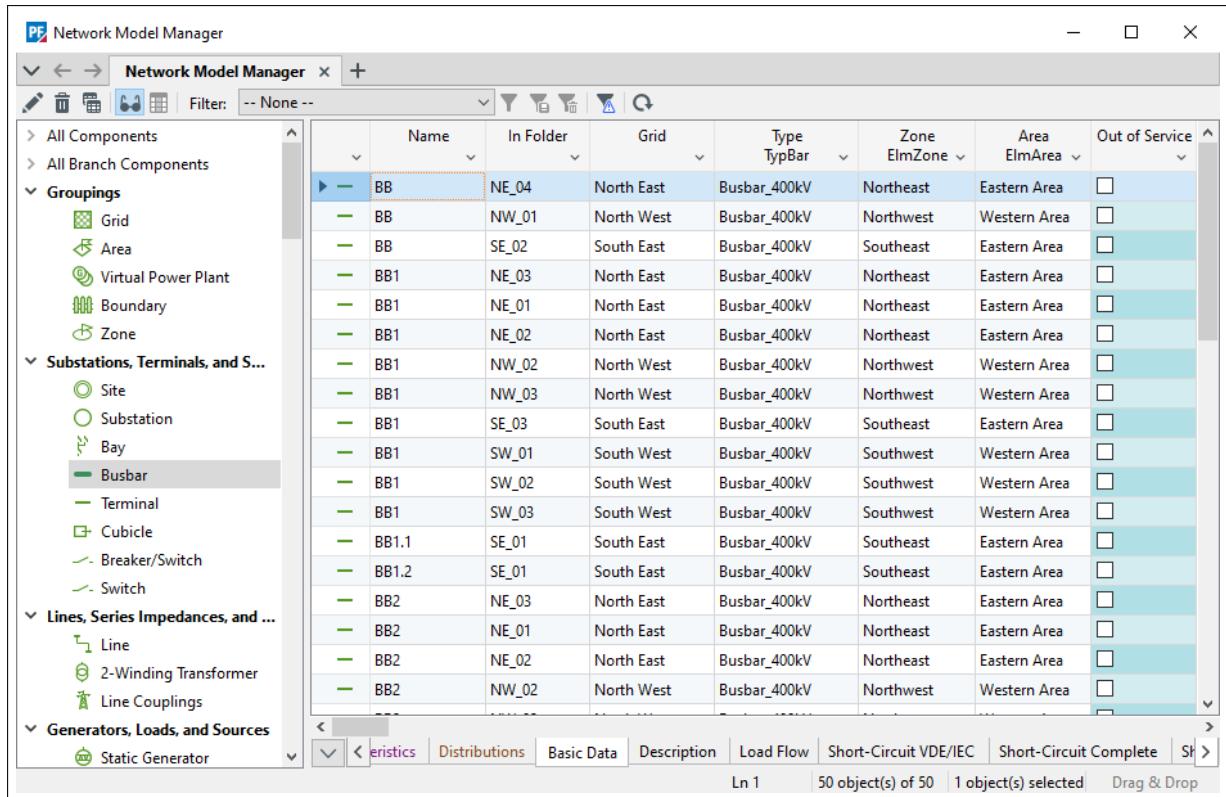


Figure 11.4.1: The Network Model Manager window

### 11.4.2 Saving of filters

As described above in Section 11.2.7, data shown in a Network Model Manager can be filtered. The filter or filters that are applied then form the *Current Working Filter*, and further options become available:

- If one or more columns are filtered, the *Edit Filter* button will become active. Clicking on this will open an edit dialog, in which a filter name can be entered. Within this dialog, it is also possible to modify the individual column filter(s) by double clicking the relevant cell(s) in the Filter column.
- The *Current Working Filter* is temporary. That means, if the Network Model Manager window is closed, the applied filters will all be discarded. However, the *Current Working Filter* can be saved under any name to reuse it, by clicking the *Save Filter* button. A window will appear, enabling the user to name the filter.
- Unwanted filters can also be deleted from the list of saved filters, by selecting a Filter from the drop-down list in the icon bar and clicking on the *Delete Filter* button.

Saved filters are stored in the Settings\Filters folder of the project, in subfolders according to the element class.

### 11.4.3 Scenario Manager

The *Scenarios* tab appears only in the Network Model Manager, not in the Data Manager, and it has a particular function. When the Scenario tab is selected, it activates the *Scenario Manager*, which is used to view and edit data held in Operation Scenarios. This functionality is described in detail in Chapter 16 (Operation Scenarios), Section 16.4.

# Chapter 12

## Building Networks

### 12.1 Introduction

This Chapter describes basic processes for setting up a network model in *PowerFactory*. Network models are usually constructed via a network graphic, or the Data Manager of the project. Therefore it is useful to have some understanding of these two concepts before starting. See Chapters 10 ([Network Graphics](#)) and 11 ([Data Manager and Network Model Manager](#)).

### 12.2 Defining Network Models using the Graphical Editor

This section explains how the tools of the Graphical Editor are used to define and work with network models. Some basic terms to understand are:

- Node: a node is another name for a terminal, which is an object of class *ElmTerm*. Other objects such as loads and lines are connected to the nodes.
- Edge element: any element connected to a terminal (e.g. load, shunt, line, switch, transformer). It can be a single-port element or have more than one port.
- Branch element: an edge element that is connected between two or more nodes (e.g. switch, line, transformer). It has more than one port.
- Cubicle: the cubicle is not an element represented on the diagram; it is internal to a terminal and can be thought of as the point where an object is connected to the terminal.

#### 12.2.1 Adding New Power System Elements

When new elements are created via a diagram graphic, they will by default be stored in the folder of the grid associated with that graphic (“Target folder for network elements”).

*PowerFactory* provides a Drawing Toolbox from which elements can be selected. This toolbox is only visible to the user when a project and study case are active and the open graphic is made editable by deselecting the *Freeze Mode* button (🔒). The Drawing Toolbox will then be seen on the right-hand side of the GUI. The process is that elements are first created and then their parameters subsequently edited through the element and type dialogs. Information about the element and type parameters are given in the [Technical References Document](#).

To create a new power system element, left-click once on the corresponding icon in the toolbox. Then

the cursor will have this symbol “attached” to it. Then a left-click on the graphic will create a new element of the selected class. The **Esc** key, or right mouse-click, can be used to stop this process.

Power system elements are placed and connected in the single line graphic by left clicking on empty places on the drawing surface (places a symbol), and by left clicking on nodes (makes a connection). It is therefore recommended to start by creating at least some of the nodes (terminals) in the network first.

The connection between edge elements and terminals is carried out by means of cubicles. When working with the graphical editor, the cubicles are automatically generated in the corresponding terminal.

---

**Note:** When connections to terminals are defined with switch elements of the class *ElmCoup* (circuit breakers), cubicles without any additional switches (*StaSwitch*) are generated.

---

### 12.2.2 Nodes

When starting to build a network, it is usual to first place the required terminals (*ElmTerm*) on the graphic. There are several representations of terminals available in the Drawing Toolbox. Note that terminals have a parameter `e:iUsage`, which is set to Busbar, Internal Node or Junction Node; by default this will be set to Busbar unless the “point” representation is selected.

- *Busbar*. This is the most common representation of a node.
- *Busbar (Short)*. Looks the same as a Busbar but is shorter and the results box and name is placed on the *Invisible Objects* layer by default. Typically used to save space or reduce clutter on the graphic.
- *Junction / Internal Node*. Typically used to represent a junction point, say between an overhead line and cable. The results box and name is placed on the *Invisible Objects* layer by default.
- Busbar (rectangular)*. Typically used for reticulation and / or distribution networks.
- Busbar (circular)*. Typically used for reticulation and / or distribution networks.
- Busbar (polygonal)*. Typically used for reticulation and / or distribution networks.

Busbars (terminals) should be placed in position and then, once the cursor is reset, dragged, rotated and sized as required. Re-positioning is performed by first left clicking on the terminal to mark it, then clicking once more so that the cursor changes to , and then holding the mouse button down and dragging the terminal to a new position. Re-sizing is performed by first left clicking on the terminal to mark it. Sizing handles appear at the ends.

If a terminal is inserted into a line, the line will be automatically split at the selected length.

### 12.2.3 Edge Elements

Edge elements are elements which connect to nodes.

Single port elements (loads, machines, etc.) can be positioned in two ways. The simplest method is to select the symbol from the toolbox and then left click the busbar where the element is to be placed. This will draw the element at a default distance under the busbar. In case of multi busbar systems, only one of the busbars need be left-clicked. The switch-over connections to the other busbars will be made automatically.

The “free-hand” method first places the element symbol wherever desired, that is, first click wherever you wish to place the symbol. The cursor now has a “rubber band” connected to the element (i.e. a

dashed line), left-clicking on another node will connect it to that node. To create corners in the joining line left click on the graphic. The line will snap to grid, be drawn orthogonally, as determined by the “Diagram Settings” that have been set.

If a single port element is connected to a terminal using the first method (single left click on busbar), but a cubicle already exists at that position on the busbar, the load or machine symbol will be automatically positioned on the other side of the terminal, if possible.

---

**Note:** By default all power system elements are positioned “bottom down”. If the element has already been placed and one wishes to flip it to the other side of the terminal, it can be done by selecting the element and the *right-click* → *Graphic Object* → *Flip at Node*.

---

Once drawn, an element can be rotated using the context menu by *right-click* → *Graphic Object* → *Rotate*.

Double port elements (lines, transformers, etc.) are positioned in a similar manner to single port symbols. By left-clicking the first busbar, the first connection is made. The second connection line is now held by the cursor. Again, left-clicking the drawing area will create corners. Double-clicking the drawing area will position the symbol (if not a line or cable - e.g. a transformer). The second connection is made when a node is left clicked.

Triple port elements (e.g. three-winding transformers) are positioned in the same manner as two port symbols. Clicking the first, and directly thereafter the second node, will place the symbol centred between the two nodes, which may be inconvenient. Better positioning will result from left clicking the first busbar, double-clicking the drawing space to position the element, and then making the second and third connection.

The ‘free-hand’ method for two and triple port elements works the same as for one port elements.

---

**Note:** Pressing the **Tab** key after connecting one side will leave the second leg unconnected, or jump to the third leg in the case of three port elements (press Tab again to leave the third leg unconnected). Pressing **Esc** or right-click will stop the drawing and remove all connections. If the element being drawn seems as if it will be positioned incorrectly or untidily there is no need to escape the drawing process; make the required connections and then right-click the on the element and select *Graphic Object* → *Redraw...*, to redraw the element whilst retaining the data connectivity.

---

It is recommended that the connections for a transformer are always made in order of voltage, starting with the highest voltage connection.

It is possible to insert a terminal into an existing line in the single line diagram by placing the terminal on the line itself. This splits the line into two, defaulting at 50 %. If the terminal is then moved, the adjacent line sections will automatically be redrawn. If the terminal needs to be moved (graphically) along the line, this can be done by holding the **Ctrl+Alt** keys whilst moving the terminal. Note that both these adjustments are just graphical and do not change the actual lengths of the two lines.

Annotations are created by clicking one of the annotation drawing tools. Tools are available for drawing lines, squares, circles, pies, polygons, etc. To draw these symbols left click at on an empty space on the single line diagram and release the mouse at another location (e.g. circles, lines, rectangles). Other symbols require that you first set the vertices by clicking at different positions and finishing the input mode by double-clicking at the last position.

For further information on defining lines, see Section [12.3 \(Lines and Cables\)](#).

### 12.2.4 Cubicles

A cubicle (*StaCubic*) in *PowerFactory* is an object which stores information about the connection between an edge element and a node element. Whenever an edge element is connected to a node element it must be connected via a cubicle. However, the cubicle is created automatically when an edge element and a node are connected and the user does not generally need to take special measures to facilitate the creation of the cubicle.

In the data manager cubicles are stored within node elements. A node element can contain cubicles which do not have an edge element associated with them. This can happen if for example an edge element is connected to a node and then disconnected. A cubicle is automatically created during the connection but is not automatically deleted upon disconnection and therefore remains in the node. If alternatively the edge element was deleted instead of disconnected, in this case, the cubicle would also be deleted. If an attempt is made to connect an edge element to a node containing such unassigned cubicles then *PowerFactory* will give the user a choice of unassigned cubicles to which they can connect.

In addition to storing information about the associated connection, a cubicle is also used as a storage location for certain objects. For example, relays, switches, circuit breakers and measurement devices can all be stored inside cubicles. Only one switching device (*StaSwitch*) can be stored inside a cubicle and this device can be used to toggle the connection between the edge element and the node element. By default a switching device is always created in a cubicle, but the user can also choose to remove the switching device if required.

### 12.2.5 Marking and Editing Power System Elements

A left-click on an element selects it and it then becomes the “focus” of the next action or command. For branch elements, the parts near their connection to nodes are treated differently and show specific context menu options regarding the marked side of the element (e.g. to insert a new device at the line end or to disconnect the line). To get all the menu options anyway, hold down the **Ctrl**-key while clicking the right mouse button.

The element can be un-marked or de-selected by clicking on another element, by clicking onto some free space in the graphic or just by pressing the **Esc** key.

There are different ways to select several objects at once:

- Pressing the *Mark All Elements* button (  ), or using **Ctrl+A** to mark all graphical elements.
- If the *Rectangular Selection* button (  ) is pressed (default condition), a set of elements can be selected by clicking on a free spot in the drawing area, holding down the left mouse button, moving the cursor to another place, and release it. All elements in the so defined rectangle will now be marked.
- One or more objects can be marked holding down the **Ctrl** key whilst marking the objects.
- Holding down the **Alt**-key while clicking on the same object again marks all the adjacent objects. Doing this several times marks more and more connected objects.
- If the area to be selected cannot be covered with a rectangular form, the *Free-form Selection* button (  ) can be used to select a custom area of the diagram.

The data of any element (its edit dialog) may be viewed and edited by either double-clicking the graphic symbol, or by right-clicking it and selecting *Edit*. If multiple objects are selected, *right-click* → *Edit* will bring up a data browser.

The option *Show in Data Manager* will show the element in a Data Manager environment. The object itself will be selected (highlighted) in the Data Manager and can be double-clicked to open the edit dialog. A new Data Manager will be opened if no Data Manager is presently active. The edit dialogs for

each element may be opened from this data browser one by one, or the selected objects can be edited in the data browser directly.

---

**Note:** The position of an object in the database tree can be found by:

- Opening the edit dialog. The full path is shown in the header of the dialog.
  - Using the keyboard shortcut **Ctrl+E**, which opens the Data Manager with the element marked in the folder hierarchy.
  - Right-clicking the object and selecting *Show in Data Manager*. This will open a new database browser when required, and will focus on the selected object.
- 

### 12.2.6 Interconnecting Power Subsystems

Interconnections between two different graphics can be done in one of two ways:

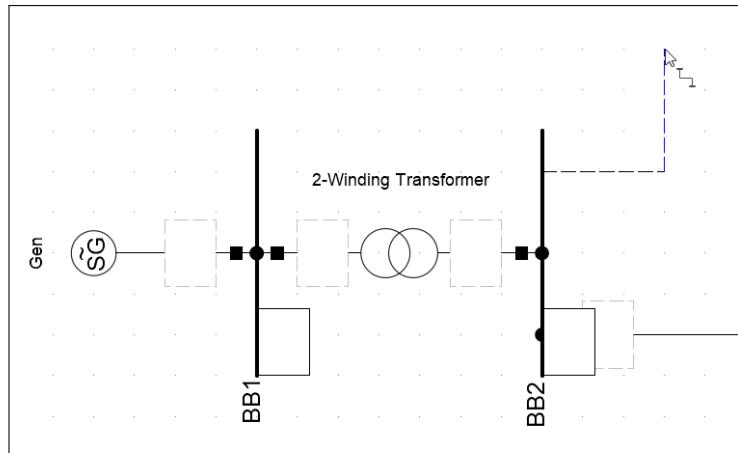
1. Representing a node in another diagram by copying (*right-click → Copy*) the node in the first graphic and pasting just the graphic object (*right-click → Paste Graphically Only*) into the second diagram. Both graphical objects are then associated with the same element; no new element is created.
2. Ensure that there is a node to connect to in the graphics that are to be interconnected. Then connect an edge element between the two graphics.

#### Example

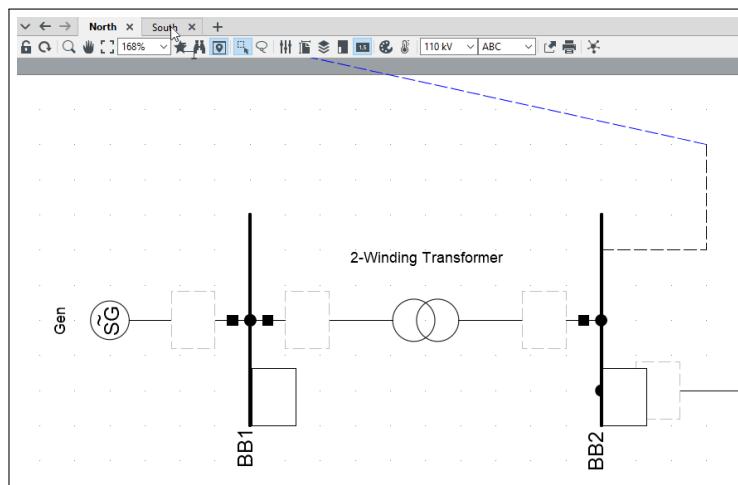
In this example a line will be used to interconnect two regions using the second method. See Figure 12.2.1.

1. Select a line drawing tool from the toolbox and create the first connection as normal by left clicking a node (see Figure 12.2.1a).
2. Double-click to place the symbol. Your cursor is now attached to the line by a “rubber band”. Move the cursor to the bottom of the drawing page and click on the tab of the graphic that the interconnection is to be made to (see Figure 12.2.1b).
3. Once in the second graphic left click to place the line symbol (see Figure 12.2.1c) and then left click on the second node.

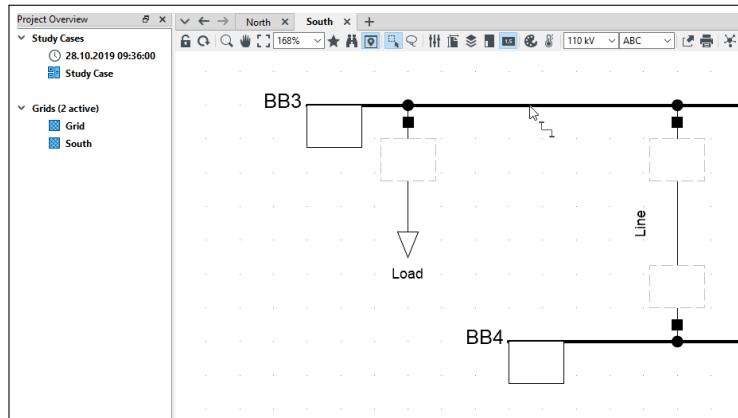
The interconnected leg is shown by a  $\gg$  symbol.



(a) First step



(b) Second step



(c) Third step

Figure 12.2.1: Interconnecting Power Subsystems

**Note:** The first method of interconnection, that of representing a node in two, or more, different graphics, may lead to confusion at a later point as the 'inflow' and 'outflow' to the node will not appear correct when just one graphic is viewed - especially if a user is not familiar with the system. The node may be right-clicked to show all connections in what is known as the "detailed diagram" (menu option *Diagrams → Show Detailed Diagram of Substation*). Thus, the second method may be preferred. To check for nodes that have connections on other graphics the *Topology → Missing graphical connections* diagram colouring may be employed.

### 12.2.7 Substations

Substations (*ElmSubstat*) and Secondary Substations (*ElmTrfstat*) can be created from predefined templates or from templates previously defined by the user by means of the icons located in the Drawing Tools toolbox, as explained below.

Depending on the user's preferences and taking into account the degree of detail required in the graphical representation of the network, the substations can be displayed as "composite nodes", which is suitable for overview diagrams, or have a so-called "design view", which provides a simplified representation with additional information on connectivity. A detailed representation of the substation topology is also offered. The different graphical representations are depicted in Figure 12.2.2.

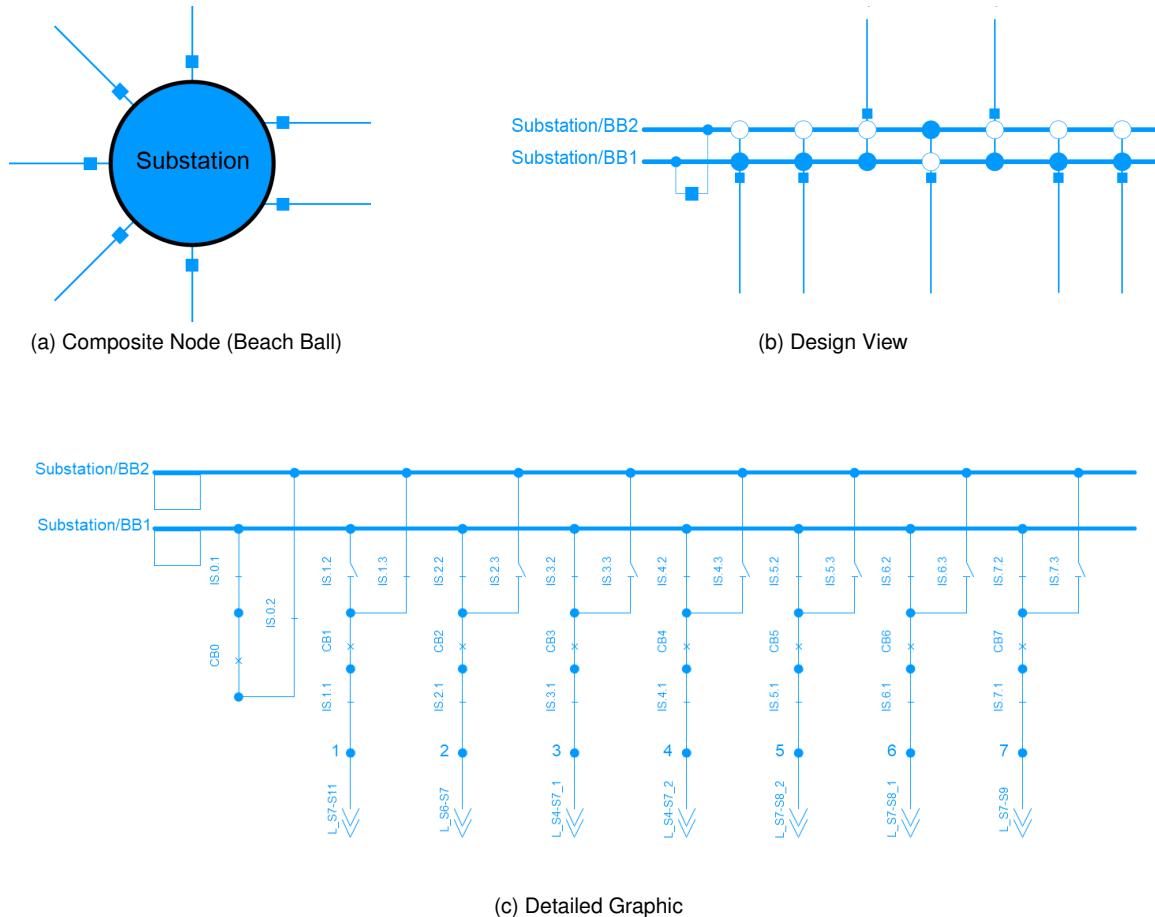


Figure 12.2.2: Substation graphical representations

**Note:** Before starting to create new substations, the user may wish to consider whether it would be useful to have Bays (*ElmBay*) created within the substation. If so, these can be automatically created if the necessary project setting is selected. See Section 9.1.3.4. More information about Bays is given in Section 12.2.7.3.

### 12.2.7.1 Creating a substation from predefined templates

#### Substation representation: Composite Node

To create a substation from a predefined template with a graphical representation as a “Composite Node” (also “Beach Ball” or “Single Node”), the following steps should be taken:

- From the Drawing Tools toolbox, select one of the following *Busbar Systems*:
  - Click on  for a substation represented by a circle
  - Click on  for a substation represented by a rectangle
  - Click on  for secondary substation
- Select the required substation template from the browser that appears.
- Click on the drawing area to place the symbol. The substation is automatically created in the Grid set as target folder for network elements.

---

**Note:** If the substation is inserted into a line or a branch, a line will be automatically split at the selected length and connected to free bays in the inserted substation. For branches, the branch itself will be split, as well as the relevant line within the branch.

---

- Close the window with the templates. Alternatively press **Esc** or right-click on the mouse to get the cursor back (browser will close automatically).
- To name the substation, right-click its symbol, select *Edit Substation...* from the context menu and rename it appropriately.
- Resize the substation symbol as required, by clicking on it and dragging one of the corners or sides.

---

**Note:** The nominal voltage of the substation can be set before drawing it, via the Node default option on the graphics toolbar (see Section 10.3)

---

To highlight the connectivity within a substation represented by means of a composite node, the user can use *Diagram Colouring*. For this, press the  button to open the diagram colouring scheme dialog. Choose the “function” for which the colouring mode is relevant (for example, the *Basic Data* page) and then select *Other → Topology → Station Connectivity*.

#### Substation representation: Design View

To create a substation from a predefined template with a “Design View” representation, the following steps should be taken:

- From the Drawing Tools toolbox, select one of the following *Busbar Systems*:
  - Click on  for a Single Busbar
  - Click on  for a Single Busbar with Tie Breaker
  - Click on  for a Double Busbar
  - Click on  for a Double Busbar with Tie Breaker
  - Click on  for a 1 1/2-Busbar
  - Click on  for a Single Busbar with Tie Breaker and Bypass
  - Click on  for a Double Busbar with Bypass

- Click on  for a Double Busbar with Tie Breaker and Bypass
- Click on the drawing area to place the symbol. The substation is automatically created in the Grid set as target folder for network elements.

**Note:** If the substation is inserted into a line or a branch, a line will be automatically split at the selected length and connected to free bays in the inserted substation. For branches, the branch itself will be split, as well as the relevant line within the branch.

- Press **Esc** or right-click on the mouse to get the cursor back.
- To name the substation, right-click its symbol, select *Edit Substation...* from the context menu and rename it appropriately.

**Note:** The nominal voltage of the substation can be set before drawing it via the Node default option on the graphics toolbar (see Section 10.3). Alternatively, the user can change the nominal voltage of one of the terminals forming the busbar system and update the nominal voltage of all the terminals inside the station.

### Switching substation representation

It is possible to change the graphical representation of the substation, i.e. to switch from a composite node representation to a design view and vice versa. To do this, select the composite node representation of the substation with a right-click and choose the option *Convert to design view*, or select one or more of the busbars in design view and *Convert to beach ball*.

#### Substation representation: Detailed Graphic

Predefined substation templates have a detailed graphical representation. To access it:

- If the substation is represented via a composite node, there are two ways to open its detailed graphic page. The first is to double-click on the corresponding composite node in the overview diagram. The second is right-click its beach ball symbol and select *Diagrams → Show Detailed Diagram* from the context menu.
- If the substation has a design view representation, its detailed diagram can be opened by selecting the option *Diagrams → Show Detailed Diagram of Substation* from the context menu.

On the detailed graphic of the substation, it is possible to display the names of neighbouring substations/sites next to the lines leaving the substation, as illustrated in Figure 12.2.3.

This is set in the Project Settings, with the option *Show jump-to labels at graphically half-connected lines* on the *Graphics* page, tab *Text Boxes*.

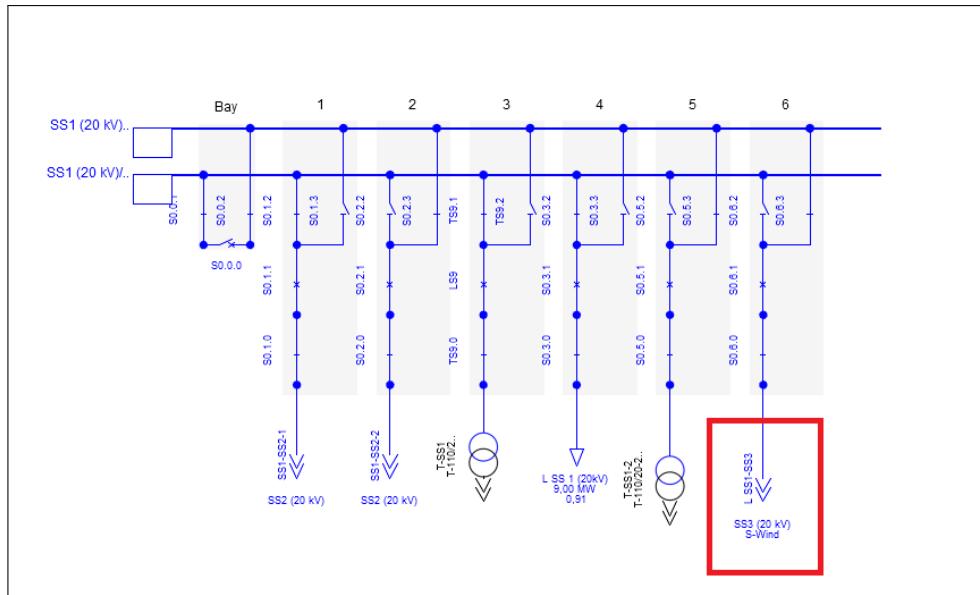


Figure 12.2.3: Displaying the names of neighboured substations in detailed graphic of substation

**Note:** Detailed graphics of substations can be generally accessed from the Data Manager. For this, the user has to go to the *Diagrams* folder (in *Network Model*), find the graphic object of the substation and open it by selecting *Diagrams* → *Show Diagram* from the context menu.

### 12.2.7.2 Creating a substation from user defined templates

The following steps describe the general procedure for inserting a user defined template:

- From the Drawing Tools tool-window select the *General Busbar System* symbol (
- Select the required substation template from the browser that appears. If the *Templates* folder in the project library contains just one template, no window is opened and the user can place the substation in the drawing area as described in the following step.
- Click on the drawing area to place the symbol. The substation is automatically created in the Grid set as target folder for network elements.

**Note:** If the substation is inserted into a line or a branch, a line will be automatically split at the selected length and connected to free bays in the inserted substation. For branches, the branch itself will be split, as well as the relevant line within the branch.

- Close the window with the templates. Alternatively press **Esc** or right-click on the mouse to get the cursor back (browser will close automatically).
- To name the substation, right-click its symbol, select *Edit Substation* from the context menu and rename it appropriately.
- Resize the substation symbol as required, by clicking on it and dragging one of the corners or sides.

**Note:** To make use of user-defined substation templates, such templates have to be created first. Information about creating user-defined templates can be found in sections [14.9.2](#) and [14.9.3](#).

### 12.2.7.3 Bays

The Bay object *ElmBay* is used to group together the network elements that normally constitute a standard bay connection of a circuit to a busbar within a substation. This grouping is useful for visualisation and organisation of the substation.

The rated current can be specified for each bay and a Thermal Rating object can be linked to it, allowing the bay capabilities to be distinguished from those of the connected element.

If the detailed diagram of the substation is viewed, the bays are highlighted by rectangular blocks (by default pale grey). These blocks are not just annotation - users can move additional elements into the area and they will also be moved into the Bay object in the project hierarchy. The Bay representation can be manually resized but will also expand automatically to encompass all objects that form part of the bay.

Additional bays may be added using the Bay icon  in the toolbox.

Note that the bay representation on the graphic is held within a layer called “Bays and Sites”, so may be made visible or invisible as required. The colour can also be configured.

### 12.2.7.4 Switching Rules

*Switching Rules (IntSwitching)* store switching actions for a selected group of switches that are defined inside a substation. The different switching actions (no change, open or close) are defined by the user considering different fault locations that can occur inside a substation. By default, the number of fault locations depends on the number of busbars and bay-ends contained inside the substation; although the user is allowed to add (and remove) specific fault locations and switches belonging to the substation. The switch actions will always be relative to the current switch positions of the breakers.

The selection of a *Switching Rule* for a substation is independent of the selection of a *Running Arrangement* and if required, the reference to the switching rule in a substation can be stated to be operational data; provided the user uses the *Scenario Configuration* object. For more information on the scenario configuration refer to Chapter 16 ([Operation Scenarios](#)).

The typical application of Switching Rules is in contingency analysis studies or reliability analysis studies, where the predefined switching rules could be immediately applied after a fault. For example, a busbar fault in a double-busbar system could be followed by switching the connections to the other healthy bus bar. The Switching Rules are composed of a matrix, which defines the required switch actions for several fault locations in the substation. Please refer to [27.4.6.1](#) for the application in contingency analysis.

*Switching Rules* are also considered during Reliability Analysis (see Chapter [46](#))

#### To create a switching rule

- *Edit a Substation*, either by right-clicking on the substation busbar from the single line graphic, and from the context menu choosing *Edit Substation*, or by clicking on an empty place in the *substation graphic*, and from the context menu choosing *Edit Substation*. This will open the substation dialog.
- Press the *Select* button (▼) in the *Switching Rule* section and select *New...*
- The new *Switching Rule* dialog pops up, where a name and the switching actions can be specified. The switching actions are arranged in a matrix where the rows represent the switches and the columns the fault locations. By default the fault locations (columns) correspond to the number of busbars and bay-ends contained inside the substation, while the switches correspond only to the circuit breakers. The user can nevertheless add/remove fault locations and/or switches from the *Configuration* page. The switch action of every defined breaker in the matrix can be changed by double clicking on the corresponding cell, as illustrated in Figure [12.2.4](#). Press afterwards **OK**.

- The new switching rule is automatically stored inside the substation element.

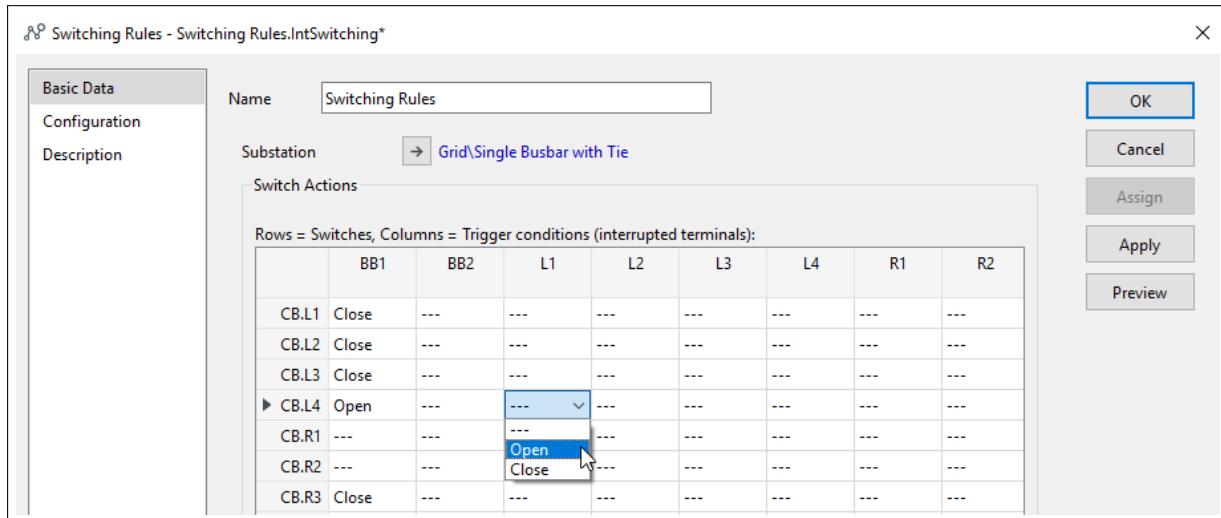


Figure 12.2.4: Switching Rule Dialog

### To select a Switching Rule

A *Switching Rule* can be selected in the *Basic Data* page of a substation dialog (*ElmSubstat*) by:

- Opening the substation dialog.
- Pressing the *Select* button (▼) in the *Switching Rule* section. A list of all Switching Rules for the current substation is displayed.
- Selecting the desired Switching Action.

### To apply a Switching Rule

A *Switching Rule* can be applied to the corresponding substation by pressing the **Apply** button from within the switching rule dialog. This will prompt the user to select the corresponding fault locations (busbars) in order to copy the statuses stored in the switching rule directly in the substation switches. Here, the user has the option to select either a single fault location, a group or all of them.

The following functional aspects must be regarded when working with switching rules:

- A switching rule can be selected for each substation. By default the selection of a switching rule in a substation is not recorded in the operation scenario. However, this information can be defined as part of an operational scenario by using the *Scenario Configuration* object (see Chapter 16 ([Operation Scenarios](#))).
- If a variation is active the selection of the Switching Rule is stored in the recording expansion stage; that is considering that the *Scenario Configuration* object hasn't been properly set.

### To assign a Switching Rule

The **Assign** button contained in the switching rule dialog allows to set it as the one currently selected for the corresponding substation. This action is also available in the context menu in the Data Manager (when right-clicking on a switching rule inside the Data Manager).

### To preview a Switching Rule

The **Preview** button contained in the switching rule dialog allows to display in a separate window the different switch actions for the different fault locations of the corresponding substation.

#### 12.2.7.5 Meteorological Station

A meteorological station *ElmMeteostat* can be selected for substations and sites to link meteorological data such as ambient temperature, wind speed and solar irradiance. This data will then be used by Thermal Rating object *IntThrating* assigned to the bays and/or terminals to determine the actual nominal current.

### 12.2.8 Sites

As noted in Section 4.6.8, a site is normally used to group network components, for example, substations of different voltage levels at the same location. Due to this particular characteristic, site elements do not have predefined templates inside the software.

The site element can be represented in overview and/or geographic diagrams; a detailed representation can also be defined.

#### 12.2.8.1 Creating a new site in overview and geographic diagrams

Site elements can be represented by a square or a circle using the buttons  and  from the Drawing Toolbar. For geographic diagrams, only the circular representation is available.

To draw a new site:

- Click on one of the site symbols ( 

Once the site is defined, a detailed diagram is automatically created. It is possible then to draw all the elements directly inside the Site diagram, using detailed substation diagram templates as explained in Section 12.2.7.1.

If the site already exists it is possible to use the *Diagram Layout Tool* to generate its detailed representation automatically. See Section 12.6 for more information about the Diagram Layout Tool.

The resizing and colouring according to connectivity of the site can be done as explained in Section 12.2.7.1.

#### 12.2.8.2 Site frames

In some overview diagrams, it may not be sufficient to use site objects for all sites; and users may want to see more detail at certain sites. This can be done by using a representation of the site as a frame within which the substations can also be seen.

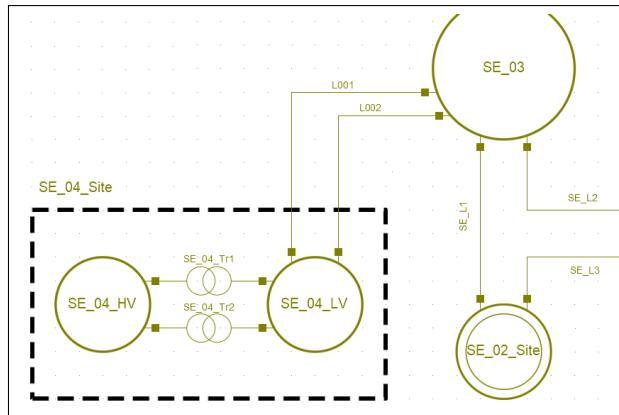


Figure 12.2.5: Substations within a site frame in an overview graphic

The site can be introduced into the graphic using the graphic object , which draws the site as a rectangular frame. The user can then create new substations within this frame and they will become part of the *ElmSite*. The Site frame will automatically resize to accommodate the new substation representations.

This new graphical option gives the user the flexibility to show sites as single symbols or in more detail, as required, and the site frame representation is held within a graphical layer called “Bays and Sites”, so may be made visible or invisible as required.

### 12.2.9 Composite Branches

New composite branches (*ElmBranch*) can be created in the Data Manager using the procedure described in Section 12.5.4 (Defining Composite Branches in the Data Manager). The definition and connection of the branch components can then be done in the single line diagram that is automatically generated upon creation of a new branch.

Branches are created in single line diagrams using previously defined templates. To create a new branch from a template:

- Click on the *Composite Branch* symbol () in the Drawing Toolbox. If there is more than one branch template (in the Templates library), a list will appear, so that the correct one can be selected.
- If the branch is to be connected to two terminals of the same single line graphic, simply click once on each terminal.
- If the branch is to be connected to a terminal from another single line diagram, you have to ‘Paste graphically’ one of the terminals on the diagram where you want to represent the branch, or connect across pages as discussed in Section 12.2.6 (Interconnecting Power Subsystems).
- If the branch is to be connected to terminals from a substation, click once on each composite node to which the branch is to be connected. You will be automatically taken inside each of those composite nodes to make the connections. In the substation graphic click once on an empty spot near the terminal where you want to connect the branch end, and then on the terminal itself.

A diagram of the newly created branch can be opened by double clicking on its symbol. In the new diagram it is possible to rearrange the branch configuration and to change the branch connections.

Details of how to define branch templates can be found in Section 14.9.4 (Composite Branch Templates).

### 12.2.10 Single and Two Phase Elements

It is possible to define the phase technology of elements such as terminals, lines, and loads. In instances where the number of phases of a connecting element (e.g. a circuit breaker or line) is equal to the number of phases of the terminal to which it connects, *PowerFactory* will automatically assign the connections. However, when connecting single-phase elements to a terminal with greater than one phase, or two-phase elements to terminals with greater than three phases, it is sometimes necessary to adjust the phase connectivity of the element to achieve the desired connections. The phase connectivity can be modified as follows:

- Open the dialog window of the element (by double-clicking on the element).
- Press the **Figure >>** button to display a figure of the elements with its connections on the bottom of the dialog window.
- Double-click on the dark-red names for the connections inside this figure.
- Specify the desired phase connection/s.

Alternatively, click the right arrow ( $\rightarrow$ ) next to the terminal entry and specify the desired phase connection/s.

---

**Note:** It is possible to colour the grid according to the phases (System Type AC/DC and Phases). For more information about the colouring refer to Section [10.3.10.1](#) (Diagram Colouring).

---

## 12.3 Lines and Cables

This section describes specific features and aspects of line and cable data models used in *PowerFactory*. Detailed technical descriptions of the models are provided in [Technical References Document](#).

In *PowerFactory*, lines and cables are treated alike, they are both instances of the generalised line element *ElmLne*. A line may be modelled simply as a point-to-point connection between two nodes and will refer to a line (*TypLne*), tower (*TypTow*), a tower geometry (*TypGeo*), a line coupling (*ElmTow*), or a cable system coupling (*TypCabsys*, *TypCabmult*) type. Alternatively, lines may be subdivided into sections referring to different types.

---

**Note:** Anywhere that 'line' is written in this section, 'lines and/or cables' may be read, unless otherwise specified.

---

The three basic line configurations are shown in Figure [12.3.1](#):

1. Top line: The simplest line is a single line object (*ElmLne*) connected between two terminal objects via two cubicle objects.
2. Middle Line: Line (*ElmLne*) objects can also be cascaded or subdivided. Again, with each *ElmLne* connected between two terminal objects via two cubicle objects.
3. Bottom line: Line (*ElmLne*) objects can also be subdivided into line section objects (*ElmLnsec*). The *ElmLne* object is again connected via cubicle objects between two terminal objects, but here the line section (*ElmLnsec*) objects constituting the line are not specifically connected between terminals themselves meaning the number of cubicles and terminals associated with such a configuration could be substantially reduced.

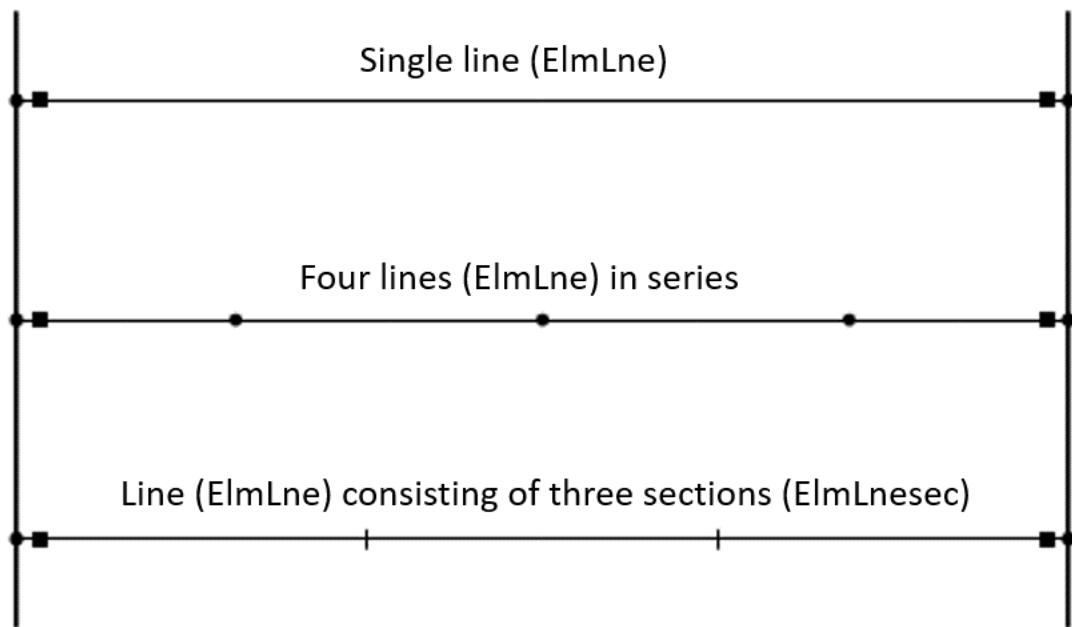


Figure 12.3.1: Basic line configurations

Cascading line objects together can be useful where it is necessary to explicitly represent the transposition of conductors. It is also possible to give each separate cascaded line a different conductor/cable type or tower geometry. Further, it is possible to branch off towards loads, generators and other substations (for example) at the intermediate terminals and include additional switching devices in the line. It may also be a useful representation to approximate the behaviour of a distributed parameter model using lumped parameter models.

An arrangement of ElmLnesec objects is generally used to represent circuits or parts of circuits consisting of multiple conductor/cable types. Such arrangements being typical of low voltage and medium voltage distribution networks.

In addition to the configurations described above, objects known as branch (*ElmBranch*) objects can be defined to organise and simplify the handling of complex composite arrangements of lines. Handling of these objects is described in sections [12.2.9](#) and [12.5.4](#).

### 12.3.1 Defining a Line (*ElmLne*)

The simplest line model is a point-to-point connection between two nodes. This is normally done in the single line graphic by selecting the icon and by left clicking the first terminal, possibly clicking on the drawing surface to draw a corner in the line and ending the line at the second terminal by left clicking it. This will create an *ElmLne* object in the database. When this object is edited, the following dialog will appear.

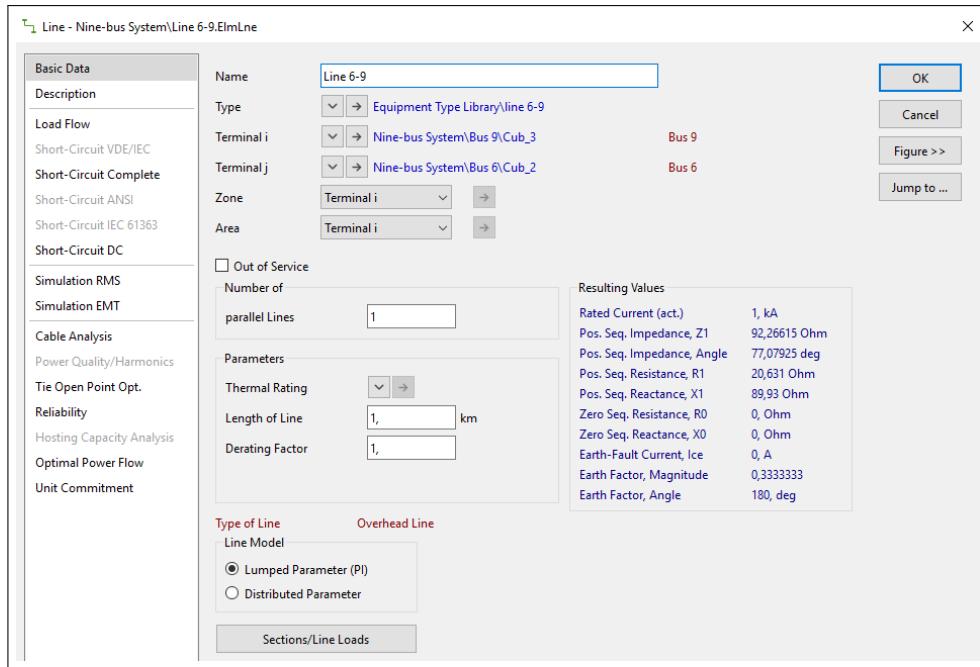


Figure 12.3.2: Editing a line

The dialog shows the two cubicles to which the transmission line is connected (*terminal i* and *terminal j*). The line edit dialog shows the name of the node (in brown) in addition to the name of the cubicle (in blue). The actual connection point to the node is the cubicle and this may be edited by pressing the edit button ( $\rightarrow$ ). The cubicle may be edited to change the name of the cubicle, add/remove the breaker, or change phase connectivity as discussed in Section 12.2.10 (Single and Two Phase Elements).

The type of the line is selected by pressing the ( $\downarrow$ ) next to the type field. Line types for a line are:

- The *TypLne* object type, where electrical parameters are directly written (the user can select if the type is defined for an overhead line or a cable).
- Tower types (*TypTow* and *TypGeo*), where geometrical coordinates and conductor parameters are specified, and the electrical parameters are calculated from this data. Selection of the tower type will depend on the user's requirement to link conductor type data to the line element as in *TypGeo* (for re-use of the one tower geometry with different conductors), or to link conductor type data to the tower type as in *TypTow* (for re-use of one tower geometry with the same conductors).
- Cable definition types (*TypCabsys*), used to complete the definition of a cable system. It defines the coupling between phases, i.e. the coupling between the single core cables in a multiphase/multi-circuit cable system.

Once the lines (or cables) have been created it is possible to define couplings between the circuits that they are representing by means of line coupling elements *ElmTow* (for overhead lines) and cable system coupling elements *ElmCabsys* (for cables).

Details of how to create Line Sections, Cable Systems, and Line Couplings are provided in the following sections, and further information about line/cable modelling is given in the [Technical References Document](#).

### 12.3.2 Defining Line Sections

To divide a line into sections:

- Press the **Sections/Line Loads/Compensation** button in the line dialog. This will open a data browser showing the existing line sections (if any).
- Click on the *New Object* icon ( ) and select the element *Line Sub-Section (ElmLnesec)*.
- The edit dialog of the new line section will appear, and the type and length of the new section can be entered.

### 12.3.3 Defining Line Compensation

It is possible to define the Line Compensations at one or both ends of a line element. In order to achieve that, following steps should be followed:

- Press the **Sections/Line Loads/Compensation** button in the line dialog. If there are existing line compensations then they will be listed in the opened data browser.
- Next, click on the *New Object* icon ( ) and select the element *Line Compensation (ElmLncomp)*.
- The edit dialog of the new line compensation will appear. Based on the selected *Input Mode*, the parameters can be updated.

### 12.3.4 Defining Line Couplings

The Line Couplings element (*ElmTow*) is used to represent electromagnetic coupling between transmission lines. In order to define a line coupling, a tower type (*TypTow* or *TypGeo*) determining the geometrical characteristics is required, along with the conductor type (*TypCon*) of the circuits.

Since line coupling occurs between lines on the same tower or between lines running approximately parallel to each other, the lines should be the same length; if they are not, a warning message will be displayed when calculations are executed and the shorter length will be considered for the coupling. To facilitate this line objects can divided into two objects with one of the divided parts assigned a length as well as a coupling to other line objects of the same length. The other divided part can be assigned the remainder of the length of the circuit and no coupling.

The line coupling can be directly defined in the Data Manager; however it is easier to do it from the single line diagram as follows:

1. Select the *Line Couplings* icon from the drawing tools ( ) and with the left mouse button held down, drag the cursor over the lines in the single line diagram. Alternatively, select the lines, right-click on them and select *Network Models* → *Line Couplings* → *New...* from the context menu. In both cases, a graphical representation of the tower will be automatically added to the diagram as shown in Figure 12.3.3.

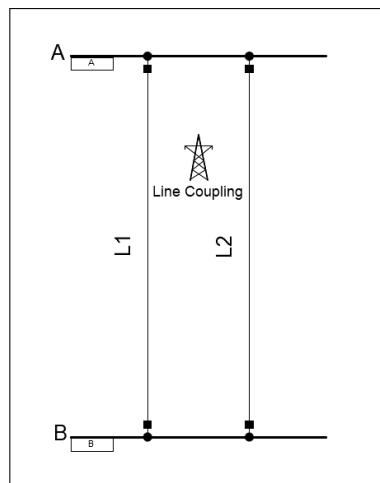


Figure 12.3.3: Graphical representation of the Line Coupling element

When using the Data Manager to define the line coupling, if the graphical representation is wanted, the Diagram Layout Tool should be used.

2. A dialog pointing to the *Equipment Type Library* will open. At this point you have to select the tower type, either a *TypTow* or a *TypGeo*. If none of them are yet available, the button *New Object* ( ) can be used to define a new tower type. In this example a *TypTow* will be used.
3. On the edit dialog of the tower type, shown in Figure 12.3.4, the number of circuits and earth wires should be defined. Then the conductor types should be selected, by double clicking on the *TypCon* field.

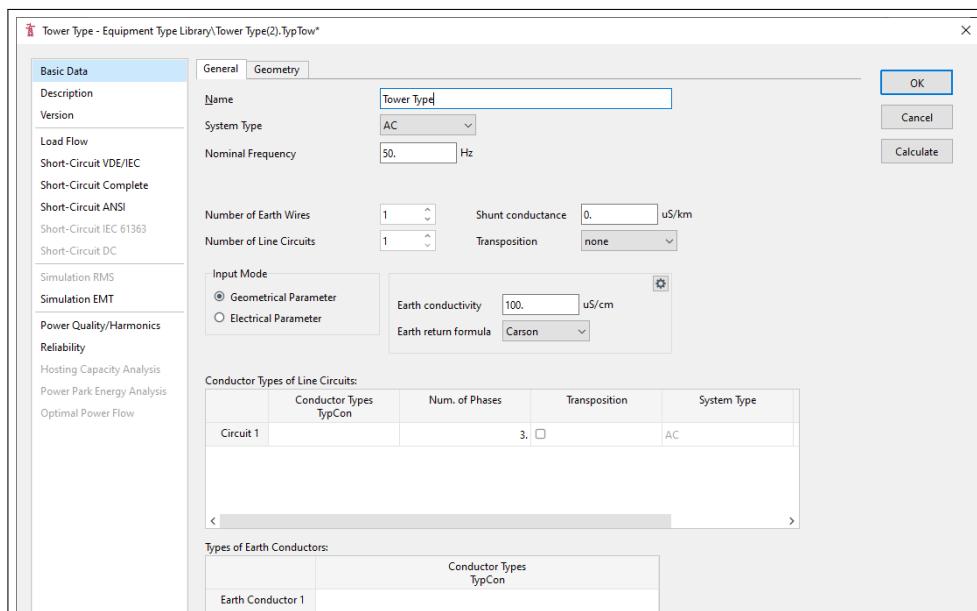
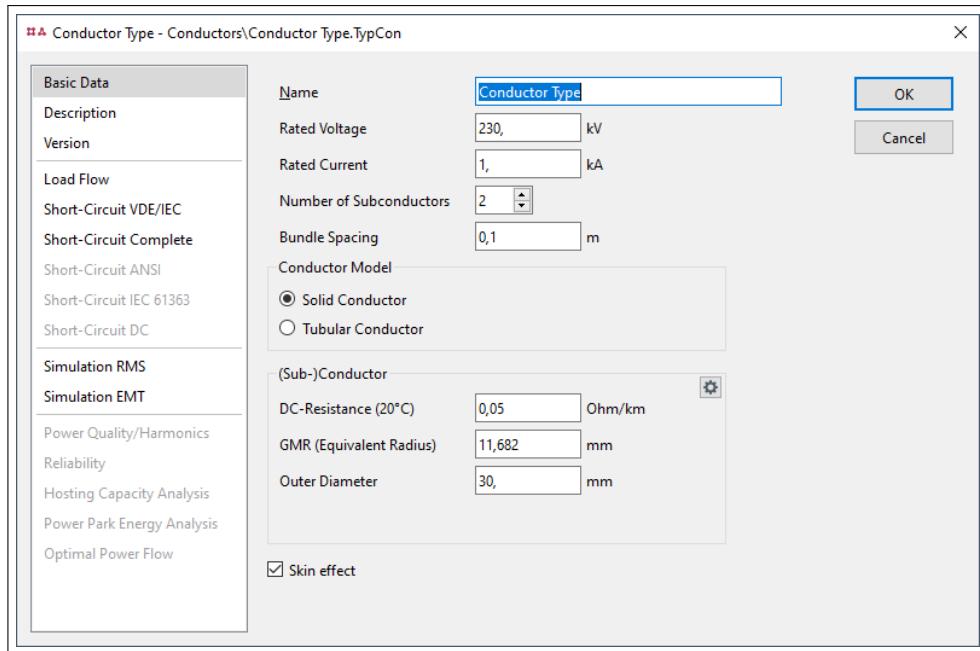
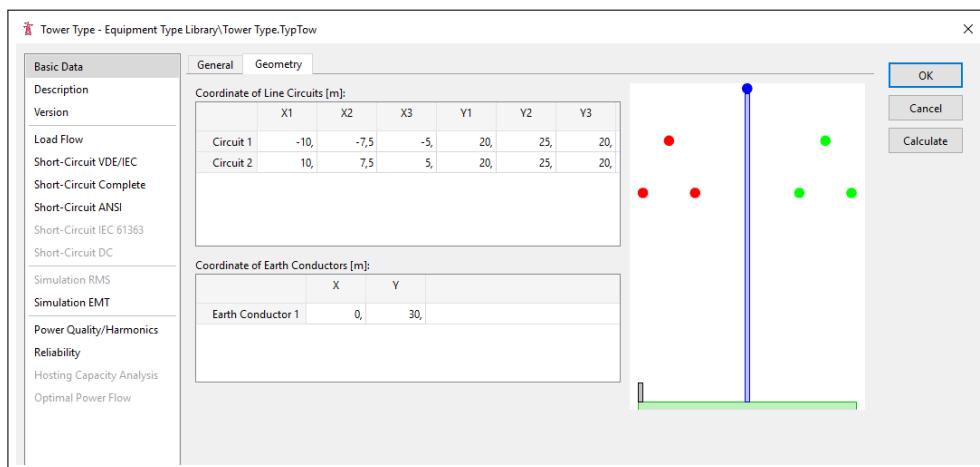


Figure 12.3.4: Tower Type (*TypTow*)

4. Once again, a dialog pointing to the *Equipment Type Library* will open to select the conductor type. If no type is yet available, the button *New Object* ( ) can be used to define a new conductor type (*TypCon*).
5. On the edit dialog of the conductor type, shown in Figure 12.3.5, the rated voltage and current, number of subconductors, model and measurements should be defined.

Figure 12.3.5: Conductor Type (*TypCon*)

6. Separate conductor types can be used for each circuit and earth wires. Note that earth wires are only to be entered if the *Input Mode* is set to *Geometrical Parameters*.
7. Once the conductors are defined, the rest of the parameters of the tower type should be entered. The transposition can be selected as:
  - None
  - Circuit-wise
  - Symmetrical
  - Perfect
 More information about these options can be found in the [Technical References Document](#) (Overhead Line Constants).
8. On the *Geometry* page of the tower type edit dialog, the disposition of the conductor can be defined by inserting the coordinates in metres, on the right side of the dialog, an image of the location of the phases is shown. The button **Calculate** can be used to get the matrix of impedances; more information about the calculation and values obtained is also available in the technical reference for the Overhead Line Constants.

Figure 12.3.6: Geometry of the Tower Type (*TypTow*)

9. Once the tower type is defined, click on the **OK** button. A new dialog will open, where the lines (*ElmLne*) are assigned to the circuits. Select the line for each circuit and click **OK**. Now the line coupling element (*ElmTow*) is complete. Note that once a line coupling has been assigned to a line element, the type of the line changes to line coupling.

The example above uses a *TypTow* tower type, but line couplings can also be defined using the tower geometry type *TypGeo*. The main difference is that within the tower type (*TypTow*) the geometry of the tower is associated with the corresponding conductor types of each circuit and therefore the tower type contains all data of the overhead line transmission system as required for the calculation of the electrical parameters. The tower geometry type (*TypGeo*), however, does not contain a reference to the conductor type, so that the definition is not complete and the conductor types are added later on in the line (*ElmLne*) or coupling (*ElmTow*) elements. This makes the tower geometry type (*TypGeo*) more flexible, and it is therefore the preferred option when combining the same tower geometry with different conductor types.

The following figure presents a comparison of line couplings using different tower types.

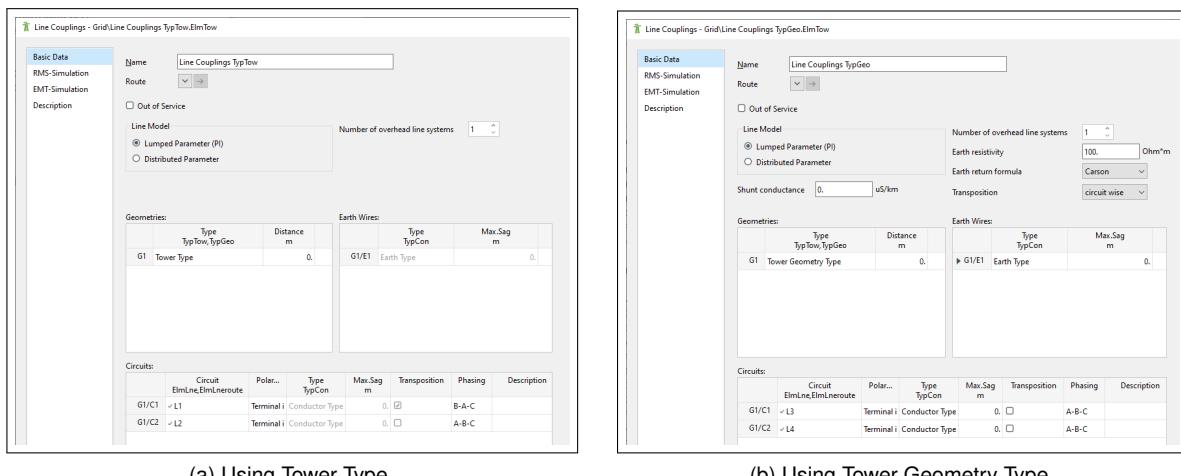


Figure 12.3.7: Line Couplings Element

### 12.3.5 Defining Cable Systems

A cable system can be used to calculate and represent the impedance of a cable or group of cables including the calculation of mutual impedances developed between all conductors comprising the system. Unlike the simple *TypLne* representation of a cable the input data does not include sequence impedance data but rather includes geometrical data e.g. the relative positions of individual cables and individual cores as well as data about the construction of the cables for example the core material, the core cross section, the insulation type, or the presence of a sheath and armour. It is from this data that *PowerFactory* calculates the impedances and admittance matrices of the arrangement, which are subsequently used to represent the system in the various calculations.

Figure 12.3.8 illustrates that there are essentially two different kinds of cable system that can be constructed in *PowerFactory*.

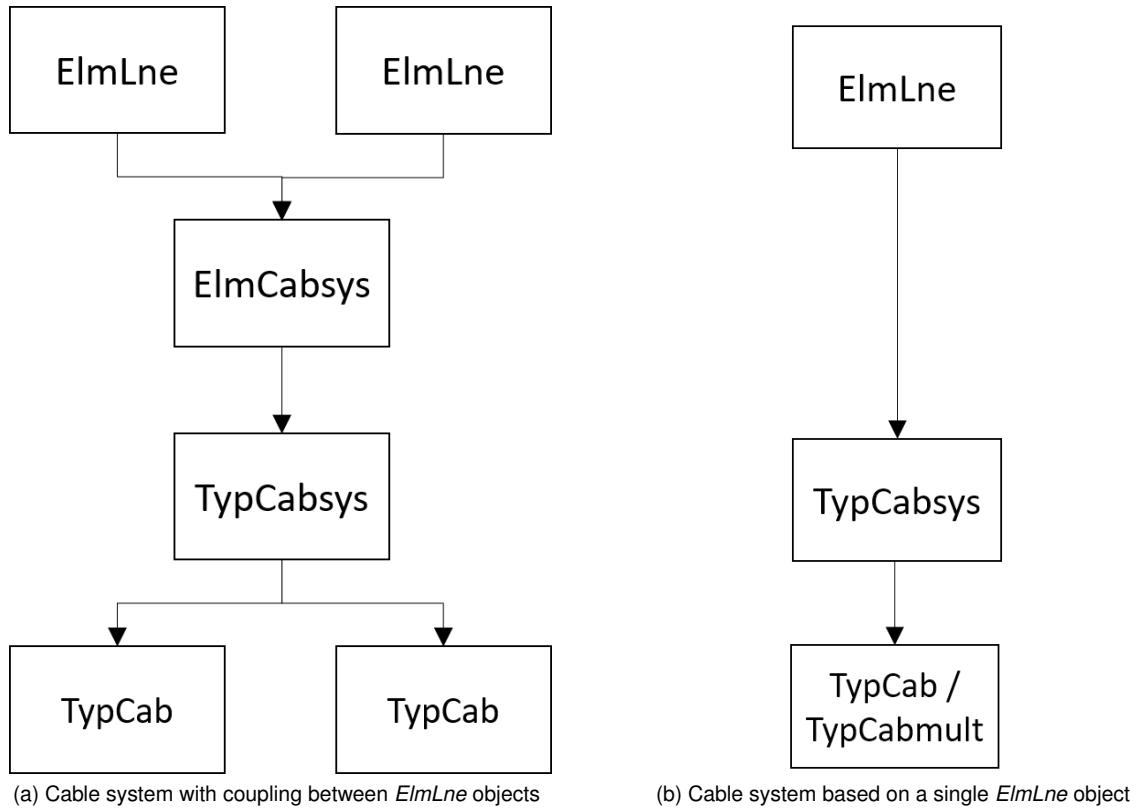


Figure 12.3.8: Cable System Overview

The cable system shown on Figure 12.3.8a illustrates two line objects (*ElmLne*) which are coupled together using a Cable System Element object (*ElmCabsys*). Although only two line objects are illustrated it is important to realise that practically there is no limit to the number of *ElmLne* objects which can be coupled. Further, the *ElmLne* objects being coupled can have 1,2 or 3 phases and they can represent many different types of conductor for example a phase conductor, a neutral, a sheath or an armour with the type of conductor they represent likely to correspond with the designation of the node on which they are terminated. They can also be connected at different voltage levels. Each *ElmLne* representing one or more phase conductor is referred to as a circuit in the *ElmCabsys* and *TypCabsys* objects. Each circuit must be assigned a Single Core Cable type (*TypCab*) or a Multicore/Pipe Cable type (*TypCabmult*). If sheaths or armours of the cables are to be considered then this is specified in the *TypCab* or *TypCabmult* objects.

**Note:** If a Multicore/Pipe Cable type (*TypCabmult*) is assigned to the Cable Definition (*TypCabsys*), only one circuit is allowed.

Since coupling generally occurs between cables sharing a route over the part of the route where the cables run approximately parallel to each other, the cables coupled in this cable system arrangement should be specified to have the same length; if they are not, a warning message will be displayed when calculations are executed and the shorter length will be considered for the coupling. To facilitate this Line objects can divided into two objects with one of the divided parts assigned a length as well as a coupling to other line objects of the same length. The other divided part can be assigned the remainder of the length of the circuit and no coupling.

The cable system can be directly defined in the Data Manager; however it is easier to do it from the single line diagram as follows:

1. Select the Cable System element from the drawing tools (🌐) and with the left mouse button held

down, drag the cursor over the cables in the single line diagram. Alternatively, multi-select the cables to be coupled, right-click on them and select *Network Models* → *Cable System* → *New...* from the context menu. In both cases, a graphical representation of the cable system will be automatically added to the diagram.

When using the Data Manager to define the cable system, if the graphical representation is wanted, the Diagram Layout Tool should be used.

2. A dialog pointing to the *Equipment Type Library* will open. At this point you are asked to select a *TypCabsys* object. If no appropriate existing types are available, the button *New Object* () can be used to define a new Cable System Type.
3. On the basic data page of the dialog of the cable definition type, shown in Figure 12.3.9, the number of circuits should be defined. Then the cable types should be selected, by double clicking on the *TypCab*, *TypCabmult* field.

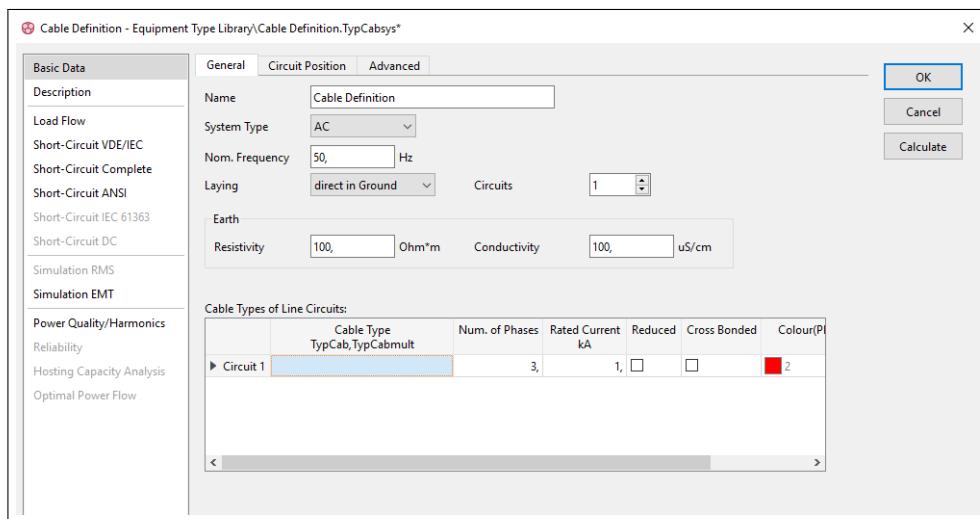
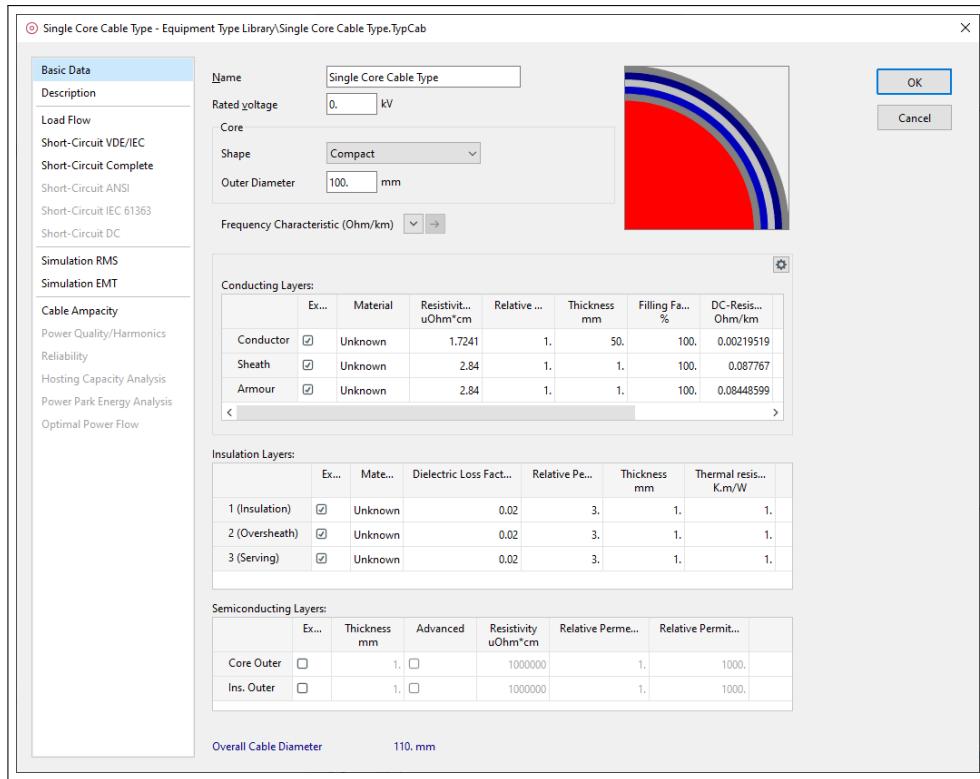
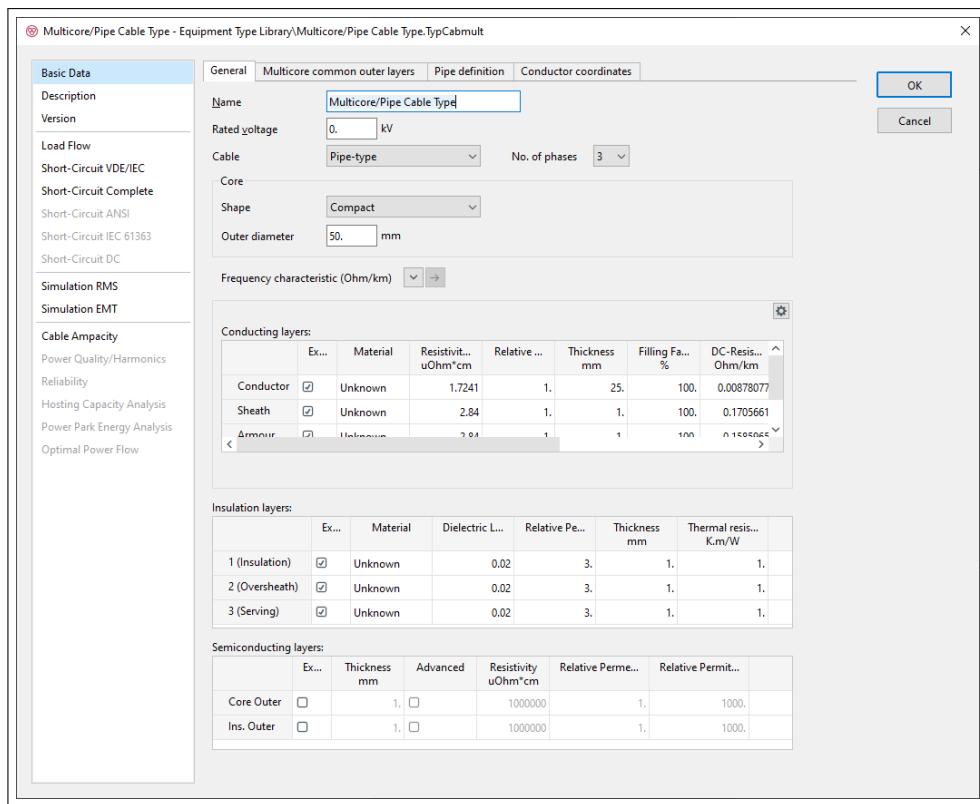


Figure 12.3.9: Cable Definition Type (*TypCabsys*)

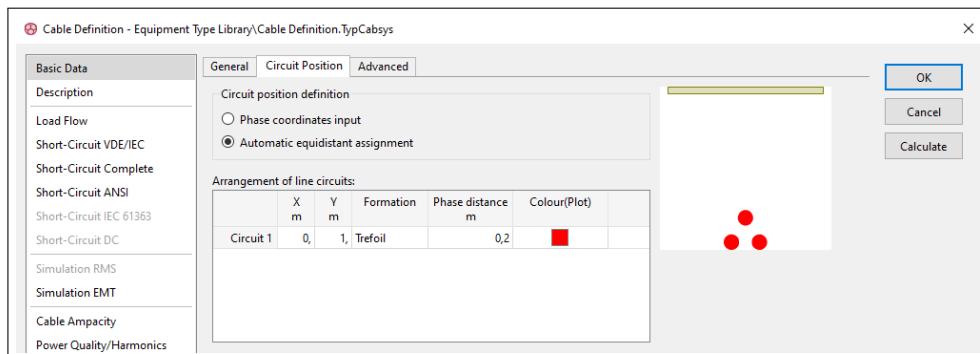
4. Once again, a dialog pointing to the *Equipment Type Library* will open for selection of a Single Core or Multicore/Pipe Cable Type. If no appropriate type is yet defined, the button *New Object* () can be used to define a new type.
5. On the Basic Data page of the dialog of the Single Core Cable type, shown in Figure 12.3.10, the rated voltage, and details of the core construction should be defined along with the characteristics of the various conducting, insulating and semi-conducting layers from which the cable is constructed. If a sheath or armour is to be considered then the relevant checkbox should be selected to confirm that it exists.

Figure 12.3.10: Single Core Cable Type (*TypCab*)

- On the Basic Data page of the dialog of the Multicore/Pipe Cable Type, shown in Figure 12.3.11, just as for the Single Core Cable type the rated voltage, and details of the core construction should be defined along with the characteristics of the various conducting, insulating and semi-conducting layers from which the cable is constructed. Again, if a sheath or armour is to be considered then the relevant checkbox should be selected to confirm that it exists. Note that additionally here it is possible to select whether the Pipe Type model or the Multicore model is to be used. Note also that depending on which model is selected, different data may be entered in the additional tabs defining either the multicore common outer layers or the pipe. Finally the geometry of the conductors within the arrangement should be defined using the Conductor coordinates tab.

Figure 12.3.11: Multicore/Pipe Cable Type (*TypCabmult*)

7. Separate Single Core Multicore/Pipe Cable types can be used for each circuit.
8. Once the types are defined, the rest of the parameters of the Cable Definition should be entered.
9. On the *Circuit Position* tab of the Cable Definition Basic Data dialog, one of two possible options for a circuit position definition can be selected. With *Phase coordinates input* the positions of the circuits can be defined by inserting the X- and Y-coordinates of each conductor. With *Automatic equidistant assignment* the user can set up a circuit for *Flat* or *Trefoil* formation with an equidistant *Phase distance*, see example in Figure 12.3.12. An image on the right side of the dialog shows the resulting location of the phases. The button **Calculate** can be used to get the matrix of impedances; more information about the calculation and values obtained is also available in the technical reference for Cable Systems.

Figure 12.3.12: Geometry of the Cable Definition (*TypCabsys*)

10. Once the Cable Definition is defined, click on the **OK** button. A new dialog will open, where the lines (*ElmLne*) are assigned to the circuits. Select the line for each circuit and click **OK**. Now the Cable System element (*ElmCabsys*) is complete. Note that once a line coupling has been assigned to a line element, the type of the line changes to line coupling.

The cable system on Figure 12.3.8b illustrates a more simple cable system configuration where coupling between *ElmLne* objects is not to be considered. In this case a Cable definition (*TypCabsys*) is assigned directly to an individual line object. This line object can also represent 1,2 or 3 phases and can also represent many different types of conductor for example a phase conductor, a neutral, a sheath or an armour with the type of conductor they represent again likely to correspond with the designation of the node on which they are terminated.

The *ElmLne* objects associated with this configuration can actually be used to represent multiple circuits and for considering the coupling between them. However, there is one limitation in that each circuit must be identical. Each circuit must be assigned the same Single Core Cable type (*TypCab*) or a Multicore/Pipe Cable type (*TypCabmult*). If sheaths or armours of the cables are to be considered then this is specified in the *TypCab* or *TypCabmult* objects. However, in this case these cannot be considered explicitly by representation with their own *ElmLne* object as they could be with the coupled cable system.

For this configuration it is easiest to define the cable system via the associated *ElmLne* dialog. In this case the type parameter of the line should be selected or defined with selection of the class *TypCabsys*. The *TypCabsys* itself is defined exactly as described for the coupled cable system.

## 12.4 Neutral Winding Connection in Network Diagrams

*PowerFactory* offers the user the option to explicitly represent the neutral connections and interconnections of the following commonly used elements:

- Power transformers (*ElmTr2*, *ElmTr3* and *ElmTr4*)
- Shunt elements (*ElmShunt*)
- External grids (*ElmXnet*)
- Synchronous (*ElmSym*) and asynchronous machines (*ElmAsm*)
- Static generators (*ElmGenstat*)
- PV systems (*ElmPvsy*)
- Neutral earthing elements (*ElmNec*)
- Harmonic Filter (*ElmFilter*)
- Step-Voltage Regulator (*ElmVoltreg*)

The interconnection of separate neutral wires is illustrated with the help of the Synchronous Generator.

A separate neutral connection can be activated by choosing the option N-Connection on the Zero Sequence/Neutral Conductor tab on the basic data page of the element as shown in Figure 12.4.1, the graphical symbol of the object will change. An illustration for the Synchronous Generator element is shown in Figure 12.4.2. Please note, once the N-Connection via a separate terminal option is selected, the Vector Groups layer can no longer be hidden in the single line diagram.

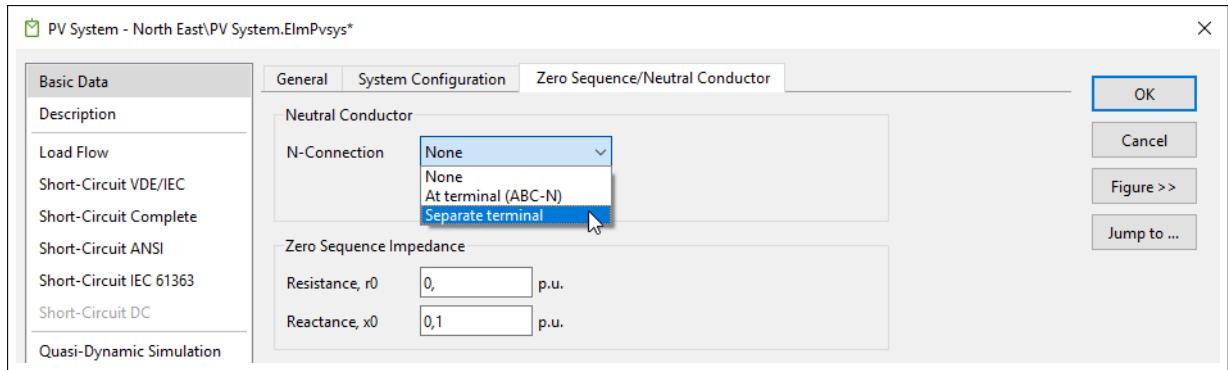


Figure 12.4.1: Zero Sequence/Neutral Connection Tab

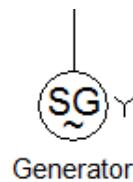


Figure 12.4.2: Generator with N-Connection via separate terminal

To connect the neutral of the Element to a neutral busbar, right-click on the element and select *Connect...*. An example of a single line diagram with the interconnection of neutral wires is shown in Figure 12.4.3. A Neutral terminal is configured by ensuring that the Phase Technology of the terminal is set to N as shown in Figure 12.4.4.

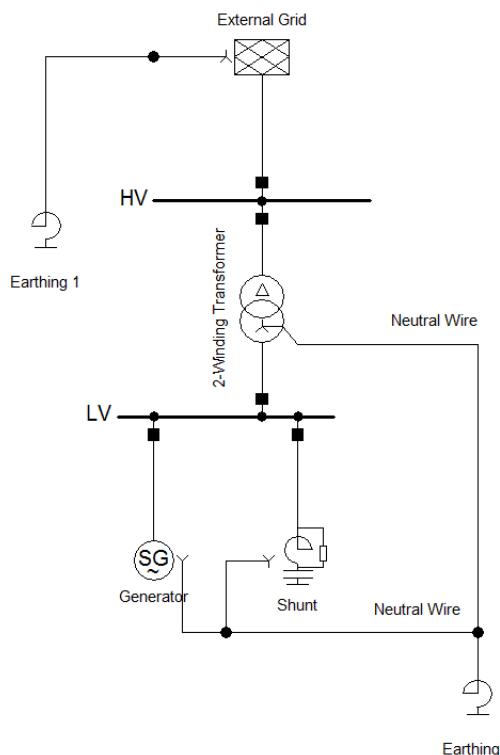


Figure 12.4.3: Grid with neutral winding connection

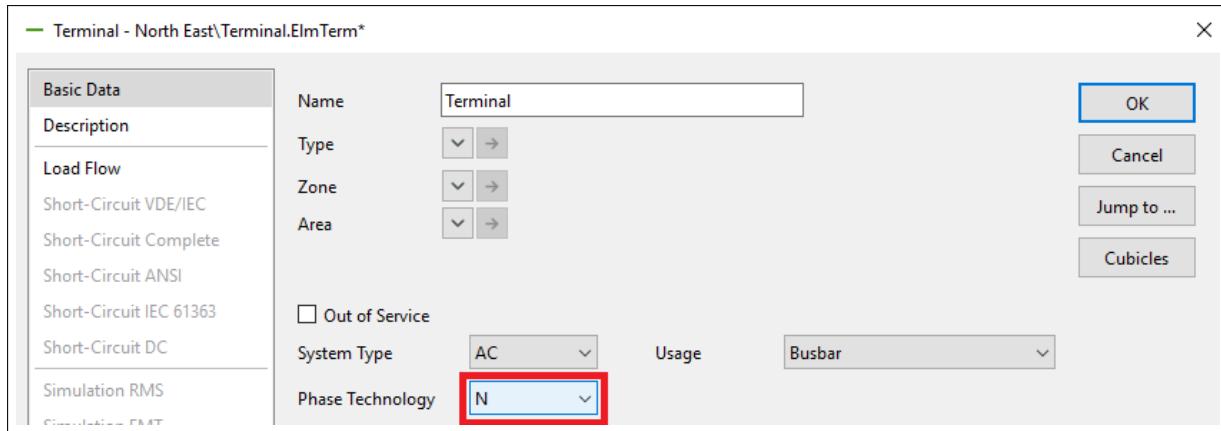


Figure 12.4.4: Set neutral Terminal

## 12.5 Defining Network Models using the Data Manager

In this section it is explained how the tools of Data Manager are used to define network models.

### 12.5.1 Defining New Network Components in the Data Manager

This section deals with defining and connecting network components using the Data Manager. General information about editing data objects in the Data Manager can be found in Chapter 11 ([Data Manager and Network Model Manager](#)), Section 11.3.4.

New network components can be directly created in the Data Manager. This is done by clicking on the target grid/expansion stage (left pane) to display its contents in the browser (right pane). Then the *New Object* icon  is used; the required object class is selected or the class name typed in directly.

### 12.5.2 Connecting Network Components in the Data Manager

To connect newly created branch elements to a node, a free cubicle must exist in the target terminal. In the 'Terminal' field (Terminal i and Terminal j for two port elements, etc.) of the edge element you have to click on the (▼) arrow to select (in the data browser that appears) the cubicle where the connection is going to take place.

To create a new cubicle in a terminal you have to open its edit dialog (double click) and press the **Cubicles** button (located at the right of the dialog). A new browser with the existing cubicles will appear, press the *New Object* icon  and select *Cubicle* (StaCubic) from the list. The edit dialog of the new cubicle will appear; by default no internal switches will be generated. If you want a connection between the edge element and the terminal through a circuit breaker, you have to press the **Add Breaker** button. After pressing **OK** the new cubicle will be available for connecting new elements.

---

**Note:** New users are recommended to create and connect elements directly from the single line graphics. The procedures described above are intended for advanced users.

---

### 12.5.3 Defining Substations in the Data Manager

The concept and the application context of substations is presented in Section 4.6 (Project Structure). A description of the procedure used to define new substations with the Data Manager is given as follows. For information about working with substations in the graphical editor refer to Section 12.2 (Defining Network Models using the Graphical Editor).

To define a new substation from the Data Manager:

- Display the content of the grid where you want to create the new substation.
- Right-click on the right pane of the Data Manager and select *New → Substation...* from the context menu.
- The new substation edit dialog will appear. There you can change the name, assign running arrangements and visualise/edit the content of the substation (directly after creation it is empty).
- After pressing **OK** the new substation and an associated diagram (with the same name of the substation) will be created.

The components of the new substation can be created and connected using the associated single line diagram or using the Data Manager; the first option is recommended. For the second option, the **Contents** button is used to bring up a browser, where the new components can be created using the *New Object* icon.

Components of a substation can of course be connected with components of the corresponding grid or even with components of other networks. The connection in the Data Manager is carried out following the same procedure discussed in the previous section.

For information about working with substations in the graphical editor refer to Section 12.2 (Defining Network Models using the Graphical Editor). For information about the definition of Running Arrangements refer to Section 14.6.11 (Running Arrangements).

### 12.5.4 Defining Composite Branches in the Data Manager

The concept and the application context of composite branches (*ElmBranch*) is discussed in Section 4.6 (Project Structure), and a description of how to define branches from within the diagram is provided in Section 12.2 (Defining Network Models using the Graphical Editor). This section explains how to define new branches from within the Data Manager.

Branches can be defined in the Data Manager as follows:

1. To create a Branch template, navigate to the *Library → Templates* folder in the Data Manager.
2. Right-click on the right pane of the Data Manager and select *New → Branch...* from the context menu.
3. In the branch edit dialog, define the name of the branch and press **OK**.
4. Now navigate back to the branch edit dialog (right-click and 'edit', or double click), and select **Contents** to add terminal and line elements etc. to the template as required. The internal elements can be connected as discussed in Section 12.5.2.
5. Use the fields 'Connection 1' and 'Connection 2' to define how the branch is to be connected to external elements.
6. To create an instance of the Branch from the created Branch template, either:
  - Select the Composite Branch  icon and connect the branch to existing terminals on the Single Line Diagram.

- Select the Composite Branch  icon and place the branch on the single line diagram, press **Tab** twice to place the branch without making any connections. Then connect the branch to external elements by right-clicking and selecting 'Connect', or double-clicking the branch and selecting external connections for the relevant internal elements (e.g. lines). Select **Update** on in the Branch dialog to update the external connections.

Alternatively, for a single Branch (i.e. not using Templates) the branch can be defined in the grid folder.

### 12.5.5 Defining Sites in the Data Manager

The concept and the application context of sites are presented in the Section [4.6](#) (Project Structure).

To define a new site from the Data Manager do the following:

- Display the content of the grid where you want to create the new site.
- Right-click on the right pane of the Data Manager and select *New → Site...* from the context menu.
- The new Site edit dialog will appear.
- After pressing **OK** the new site will be created.

---

**Note:** It is possible to move objects from a grid to a Substation, Branch, Site, etc. and vice versa.

---

## 12.6 Drawing Existing Elements using the Diagram Layout Tool

This section provides information about how to draw network components from existing objects. *PowerFactory* separates strictly the electrical (and therefore for calculations relevant) data of network elements from their graphical representation in the diagrams. Calculations of networks without any graphical representation is possible.

Designing new (extensions to) power system grids, is preferably done graphically. This means that the new power system objects may be created in a graphical environment. After the new components are added to the design, they can be edited, either from the graphical environment itself (by double-clicking the objects), or by opening a Data Manager and using its editing facilities.

It is however possible, to first create objects in the Data Manager (either manually, as explained in Section [12.5](#), or via data import using e.g. the DGS format), and subsequently draw these objects in one or more single line diagrams. If the imported data contains geographical coordinates, a geographic diagram can be created automatically by right-clicking on the Grid and choosing *Diagrams → Show Geographic Diagram*.

If no geographic coordinates are given or if a single line diagram should be created, *PowerFactory* provides the *Diagram Layout Tool*  to do that.

The following sections describe the options and possibilities of the *Diagram Layout Tool* , located in the graphic icon bar.

## 12.6.1 Action

### 12.6.1.1 Generate new diagram

When this option is selected from the *Action mode* part of the Diagram Layout Tool dialog, it is possible to create graphical representations of grids and network elements. It's a quick way to get a graphical overview of a network, offering visualisation of, for example, results or topology (colouring schemes for feeders, zones, etc.). The options in the *Generate new diagram for* part of the dialog are:

- **entire grid:** with this option a complete new diagram of the selected grid is automatically drawn. It is possible to select more than one grid; in this case one diagram showing all the selected grids will be created.  
Additional settings when using the option *Generate new diagram for* → *entire grid* are set in pages *Node Layout* (Section 12.6.2), *Edge Elements* (Section 12.6.3) and *Protection Devices* (Section 12.6.4)
- **k-neighbourhood:** if this option is used, a complete new diagram will be generated, starting with the selected elements and extending as far as specified. The “k-neighbourhood” logic is described below in Section 12.6.1.2.
- **detailed representation of:** this option can be used for substations, branches, sites and terminals. It creates a detailed diagram with all the elements contained inside the original element. No additional settings are needed.
- **feeder:** with this option a complete new schematic feeder diagram is created. It is possible to select more than one feeder; in this case a separate diagram will be created for each feeder.  
This option replaces the previous option *Show* → *Schematic visualisation* by Distance or Bus Index of the feeder. See Section 15.6 (Feeders) for further information on how to define feeders.  
Additional settings when using the option *Generate new diagram* → *Feeder* are set in pages *Node Layout* (Section 12.6.2) and *Edge Elements* (Section 12.6.3).
- **area interchange type:** this option generates an interchange diagram, used to show the power flows between defined parts of the network. Grids, Areas or Zones may be selected.  
The diagram will consist of summary boxes for those regions, and arrows showing power flows between the regions, held in a separate, configurable graphic layer. The thickness of each power arrow indicate the relative size of the power interchange.  
Additional settings when using the option *area interchange type* are set in page *Interchanges* (Section 12.6.7).

### 12.6.1.2 Auto-insert elements into current diagram

When this option is selected from the *Action mode* part of the Diagram Layout Tool dialog, it is possible to insert additional elements into an existing diagram. This option is only available if the diagram is not in “freeze” mode.

The options in the *Insert elements into current diagram* part of the dialog are:

- **K-neighbourhood expansion:** this action creates graphical elements starting from a selection of elements already graphically represented in the diagram. A selection of elements is therefore necessary. If some or all graphic elements are selected before opening the Diagram Layout Tool, these elements are automatically inserted into the Start elements selection. Alternatively the Start elements can be selected directly from the dialog using ▾ *Select...* or ▾ in the *Neighbourhood Expansion* settings.  
Starting from every selected element, the connected and not yet graphically represented neighbours are created and subsequently also their neighbours. The depth of this recursive algorithm is defined by the K-factor, which can be configured in the *Neighbourhood Expansion* settings.  
This approach offers a step-by-step creation of a diagram, where an intervention after each step is possible to adapt the final appearance of the network diagram.

Additional settings when using the *Auto-insert element into current diagram → K-neighbourhood expansion* option are set in pages *Node Layout* (Section 12.6.2), *Edge Elements* (Section 12.6.3) and *Protection Devices* (Section 12.6.4)

- **Edge elements:** this action automatically completes the current diagram with the branch elements which are not yet graphically represented. It is only available for diagrams which already contain some existing graphical node elements. Additional settings when using the option *Auto-insert element into current diagram → Edge elements* are set in pages *Edge Elements* (Section 12.6.3) and *Protection Devices* (Section 12.6.4)
- **Protection devices:** when this option is selected, protection devices are included into the current diagram according to the options set on the *Protection Devices* page described in Section 12.6.4.
- **Bays and sites:** this option enables the graphical representation of sites and bays to be added to existing diagrams, taking into account the options on the *Bays and Sites* page (Section 12.6.5).
- **Miscellaneous:** when this option is selected, the miscellaneous objects selected on the *Miscellaneous* page described in Section 12.6.6 will be inserted in the diagram.
- **Area interchange type:** the graphical representation of the power interchanges between network regions (Grids, Areas or Zones) is added to currently-active graphic. This consists of summary boxes for those regions, and arrows showing power flows between the regions, held in a separate, configurable graphic layer. The thickness of each power arrow indicate the relative size of the power interchange.  
Additional settings when using the option *area interchange type* are set in page *Interchanges* (Section 12.6.7).

### 12.6.1.3 Assisted manual drawing

This option is only available if the diagram is not in “freeze” mode. It can be used to decide which elements should be introduced in the diagram, where they should be located and which representation should be used.

Upon execution, a window will appear, listing all the elements that are not yet graphically represented in the diagram.

#### Drawing in an empty diagram

If the diagram is empty, a complete list of elements in the network is shown. This includes cubicles, bays, dynamic models, measurement devices, etc.

The process of adding elements from the list to the diagram is done via the Drawing Tools, so for example if a terminal is to be added, the required *ElmTerm* symbol is selected and the list will automatically be filtered for terminals. In this way, the Drawing Tools is used both to filter the elements to draw and also to selected the graphical representation of those elements that can have more than one, e.g. sites, substations and switches.

When clicking on a symbol, the mouse cursor will change to the selected symbol and one can start placing the elements in the diagram. It is a good practice to start with the busbars and/or busbar systems and then continue with the edge elements. As each element is placed, it disappears from the list.

Clicking on a symbol in the Drawing Tools for a different element class will switch the list to elements that can be represented by that symbol.

When selecting the symbol of an edge element, the connection(s) of the element is(are) marked in the diagram. One can then click on the connection(s) to drawn the elements of the list.

If a branch cannot be completely drawn (for example, when the terminal at only one end of a line is shown on the diagram), it is possible to draw one connection, then press **Tab** or double click on the

diagram. An arrow will appear to indicate that the branch element connects to a terminal that is not shown.

For substations, the different graphical representations, described in Section 12.2.7.1, are:

- Composite node: symbols  and 
- Simplified representation: symbol 
- Detailed representation: symbol 

User-defined objects (created using Data Extensions) can be inserted using the *Any Object* symbol .

### Drawing in a non-empty diagram

The procedure in this case is similar to the one previously described, with the difference that one or more elements of the diagram can be used as starting point for the drawing.

For example, if a terminal is in the diagram and this terminal is connected to other elements that are not in the diagram, one can click on the terminal and then execute the Diagram Layout Tool with the option *Assisted manual drawing*. This is equivalent to set the terminal as *Start element* of the neighbourhood expansion on the Diagram Layout Tool command.

The list of elements adjacent to the selected element(s) in the network is shown, i.e. the button *Adjacent Element Mode*  is automatically pressed. This activates the selecting of distance to the start element(s). Increasing the distance increase the number of adjacent elements shown.

If the button *Use all drawn nodes as starting objects*  is also selected, the list will be filtered based on all drawn nodes (not just a single starting node).

If *Show internal elements of drawn or adjacent composite objects*  is selected, elements internal to already drawn composite nodes will be shown in the list. However, since they are already drawn as part of the composite node, they should not be re-drawn.

## 12.6.2 Node Layout

The settings regarding the Node Layout take effect on the following actions:

- Generate new diagram for
  - entire grid
  - k-neighbourhood
  - feeder
- Auto-insert element into current diagram
  - K-neighbourhood expansion

The following options are available for node insertion:

- **Node spacing:** this option defines the distance between the newly created nodes in the diagram and can be set to *low*, *medium* or *high*.
- **Representation of sites and substations:** one of the following options can be selected to represent sites *ElmSite*, substations *ElmSubstat* and secondary substations *ElmTrfstat*:
  - *detailed*: the detailed diagram of the composite element, including all the internal elements (i.e. nodes, switches, transformers, lines, etc.) will be added to the diagram.
  - *design view*: a simplified view of the sites and substations (i.e. without internal elements) will be added to the diagram.

- *prefer sites*: the sites and substation will be drawn using a “beach ball” representation (circle or rectangle). If a substation is inside a site, only the site will be drawn.
- *prefer substations*: same as the previous option, but in this case all the substations will be drawn.
- **Consider physical line lengths:** with this option checked, the length of the graphical representation is based on the corresponding line length. The graphic object length is not strictly proportional to the actual line length, but nevertheless gives a good view in the diagram of the relative line lengths.
- **Adjust diagram size:** the size of the diagram defined in the *Drawing Format* is ignored and overwritten by the algorithm, which uses as much space as is needed. To get clearer outputs, this option should be selected. The new drawing size is saved and can be reused in other diagrams.
- **Custom target layer:** this option is only visible when the option *Auto-insert elements into current diagram* is selected on the *Action* page.  
A layer other than the default *Net elements* can be selected in this field, if there is a user-defined layer in the project. More information on working with layers is available in Section 10.3.7.2.

If the option *Generate new diagram for → feeder* is selected, the options of the Node Layout page include:

- **Layout Style:** this option defines the layout of the feeder; the options are *Rectangular* and *Tree*. *Rectangular* is usually recommended, since it provides the best overview of the topology of the feeder. For very large feeders, however, the rectangular layout may become too large. In this case the tree-like layout may be better, since it produces a narrower layout.
- **Layout Direction:** defines whether the feeder should be drawn from *top to bottom* or *left to right*.
- **Horizontal/Vertical node spacing:** this option defines the distance between the feeder nodes, can be set to *low*, *medium* or *high*.
- **Representation of sites and substations:** as explained in the options for node insertion.
- **Minimise extent of switches and fuses:** when this option is selected, the representation of switches and fuses is reduced to generate a smaller diagram.
- **Use graphic settings from currently shown diagram:** the settings of the feeder diagram will be taken from the diagram where the Diagram Layout Tool was executed.
- **Consider backbones:** if selected, backbones (if available) are emphasised by laying them out strictly vertically (straight line from top to bottom). Otherwise, the longest path within the feeder will be laid out vertically.

### 12.6.3 Edge Elements

The settings on the Edge Elements page take effect on the following actions:

- Generate new diagram for
  - entire grid
  - k-neighbourhood
  - feeder
- Auto-insert element into current diagram
  - K-neighbourhood expansion
  - Edge elements

The following options are available in the Edge Elements page:

- **Insert edge elements:** if this is not checked, the Diagram Layout Tool only creates graphical representations of nodes (or composite elements, if the *Draw each composite as single node* option is selected in the Node Layout page). If the option *Auto-insert element into current diagram* → *Edge elements* is selected, the edge elements are always inserted and so the setting of this option is ignored in that case.
- **Ortho Type:** if set to *Ortho*, all inserted branch elements will consist of only vertical or horizontal sections. The opposite option is *Ortho Off*, where the branch elements show a direct point-to-point connection between the according start and end nodes. With the option set to *Semi-Ortho*, the branches have a orthogonal part near the start and end node and in between a direct connection.
- **Consider composite branches (ElmBranch):** when selected, the branches will be drawn as one single element, otherwise all the internal elements of the branch will be included in the diagram.
- **Insert one-port devices connected to substations:** this option should be checked to make sure that all the one-port elements (e.g. loads, shunts) connected to the substation are inserted into the diagram regardless of the representation of the substation.
- **Insert classes only:** this option can be used as a filter to insert elements of particular classes. To achieve that, the class name of the element can be mentioned in the provided field. Also, there is a possibility for the user to insert the elements of different classes and for that purpose a comma operator “,” should be used to separate the class names.
- **Custom target layer:** this option is only visible when the option *Auto-insert elements into current diagram* is selected on the *Action* page.  
A layer other than the default *Net elements* can be selected in this field, if there is a user-defined layer in the project. More information on working with layers is available in Section [10.3.7.2](#).

#### 12.6.4 Protection Devices

The settings on the Protection Devices page take effect on the following actions:

- Generate new diagram for
  - entire grid
  - k-neighbourhood
- Auto-insert element into current diagram
  - K-neighbourhood expansion
  - Edge elements
  - Protection devices

The following options are available in the Protection Devices page:

- **Insert protection devices:** if checked, the Diagram Layout Tool inserts graphical representations of the protection devices. If the option *Auto-insert element into current diagram* → *Protection devices* is selected, the protection devices are inserted anyway.
- **Relays:** to insert graphical representations of relays (*ElmRelay*).
- **CTs and VTs:** to insert graphical representations of current transformers (*Stat*) and voltage transformers (*StaVt*).
- **Custom target layer:** this option is only visible when the option *Auto-insert elements into current diagram* is selected on the *Action* page.  
A layer other than the default can be selected in this field, if there is a user-defined layer in the project. More information on working with layers is available in Section [10.3.7.2](#).

An example of automatically inserted protection devices is shown in Figure [12.6.1](#)

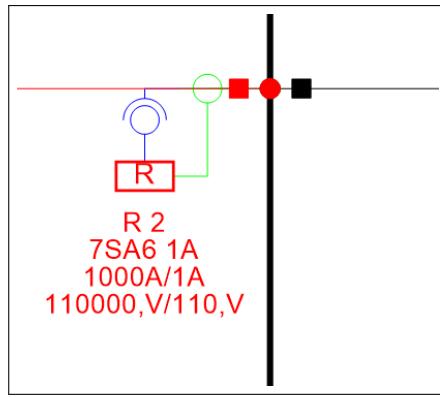


Figure 12.6.1: Protection devices in the single line diagram

If the option *Auto-insert elements into current diagram* or *Assisted manual drawing* is selected on the *Action* page, a link to the Diagram Layer Configuration is available on the *Protection Devices* page to configure the visibility of the fuses.

### 12.6.5 Bays and Sites

The settings on the Bays and Sites page take effect on the following actions:

- Generate complete diagram
  - Entire grid
  - k-neighbourhood
- Auto-insert element into current diagram
  - K-neighbourhood expansion
  - Bays and sites

On the *Bays and Sites* page, the options to show the bay and site representations can be selected independently.

The option *Bays in sections* should be selected to insert bays on section couplers of the substations.

It should be noted that if *Auto-insert element into current diagram* and *Bays and sites* are selected on the Actions page, this will override the main *Insert bays and sites* option on the Bays and Sites page; however, the two individual sub-options Bays and Sites will be observed.

Bays and sites can be added to a layer other than the default *Bays and Sites* using the *Custom target layer* box. A user-defined layer has to be available in the project. More information on working with layers is available in Section 10.3.7.2.

### 12.6.6 Miscellaneous

The settings on the Miscellaneous page take effect on the following actions:

- Generate complete diagram for
  - entire grid
  - k-neighbourhood
- Insert element into current diagram
  - K-neighbourhood expansion
  - Miscellaneous

On this page, the graphical representations of the following elements can be included in an existing diagram, as shown in Figure 12.6.2:

- Feeders
- Meteorological stations
- Areas
- Zones
- Operators
- Owners
- Boundaries
- Paths
- Towers
- Cable systems
- Load flow controllers
- Measurement devices
- User-defined objects (created using Data Extensions)

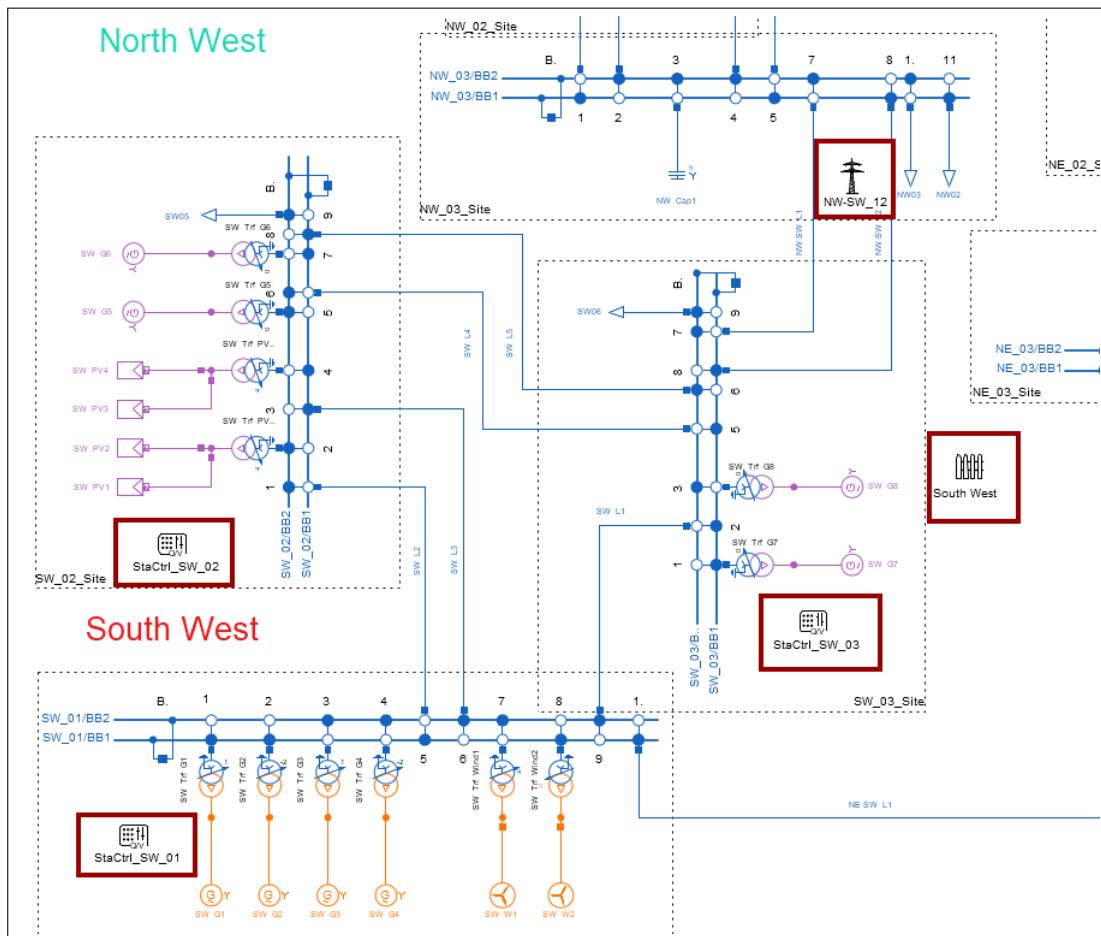


Figure 12.6.2: Single line diagram showing additional elements

The objects inserted using the *Auto-insert elements into current diagram* option on the *Action* page, can be inserted on a layer other than the default *Net elements* by checking the *Custom target layer* box and selecting a user-defined layer. More information on working with layers is available in Section 10.3.7.2.

### 12.6.7 Interchanges

The settings on the Interchanges page take effect on the following actions:

- Generate new diagram for
  - area interchange type

The options on the this page give control over the positioning of elements on the graphic:

- **prefer geographic positions** will use the Geographical Coordinates of the network elements.
- **consider positions from current diagram** is typically used if an overview diagram is currently active, for example.
- **set positions arbitrarily** the layout will be determined automatically.

On the *Show* field, it can be chose what should be displayed in the area interchange diagram: active power (*P*), reactive power (*Q*) or both (*P and Q*).

## 12.7 Other Options for Drawing Existing Elements

### 12.7.1 Drawing Existing Elements using Drag & Drop

As an alternative to using the Diagram Layout tool, it is possible to create graphical objects for existing network elements by using drag & drop:

1. Enable the *Drag & Drop* feature in a Data Manager window by double-clicking the *Drag & Drop* message in the status bar.
2. Select the element in the Data Manager by left clicking on its icon.
3. Hold down the left mouse button and move the mouse to the graphic drawing area (drag it).
4. Position the graphical symbol and release the mouse button to drop the object.
5. A new graphical symbol is created, which is representing the selected element in the diagram. No new data object is created.

### 12.7.2 Drawing Existing Stations using the Neighbourhood Expansion

Starting at any site or substation, the user can draw the neighbouring one by using the *Expand diagram to include neighbour(s)* option. If a site is selected as the starting point, the next site is drawn; in the case of a substation, the next substation is drawn.

The following description refers to substations, but the procedure for sites is the same.

1. Make sure that the diagram is unlocked (i.e. freeze mode deactivated .
2. Right click on a substation within the single line diagram *Graphic Object* → *Expand diagram to include neighbour(s)*.
3. A selection browser pops up showing all substations directly connected to that one. Select the substation you want to include. This one will be automatically drawn in the current diagram.
4. Repeat this step until all required substations are shown in the diagram.

Note that the existing graphical representation will be used, i.e. if the substation was drawn as a composite node, the neighbour substation will also be a composite node.

### 12.7.3 Single Line Diagrams Based on Geographic Diagrams

It is possible to generate a single line diagram starting from a geographic diagram by right-clicking on the diagram and selecting *Create Schematic Diagram*.

A schematic diagram, based on the geographical data, will be automatically created. If the diagram results in some areas being sparse and others too dense, the *Adjust Node Density* option can be used. This option is only visible on unlocked diagrams, and works as follows:

- In an empty part of the diagram, *right-click* → *Adjust Node Density*
- Select one or multiple regions
- Press **E** to expand or **S** to shrink the regions
- The effect of the expansion or shrinking can be adjusted with the keys **+** and **-**

Figure 12.7.1 shows an example of this functionality. On the left side is the single line diagram as generated from the geographic diagram and on the right is the diagram after adjusting the density of the nodes.

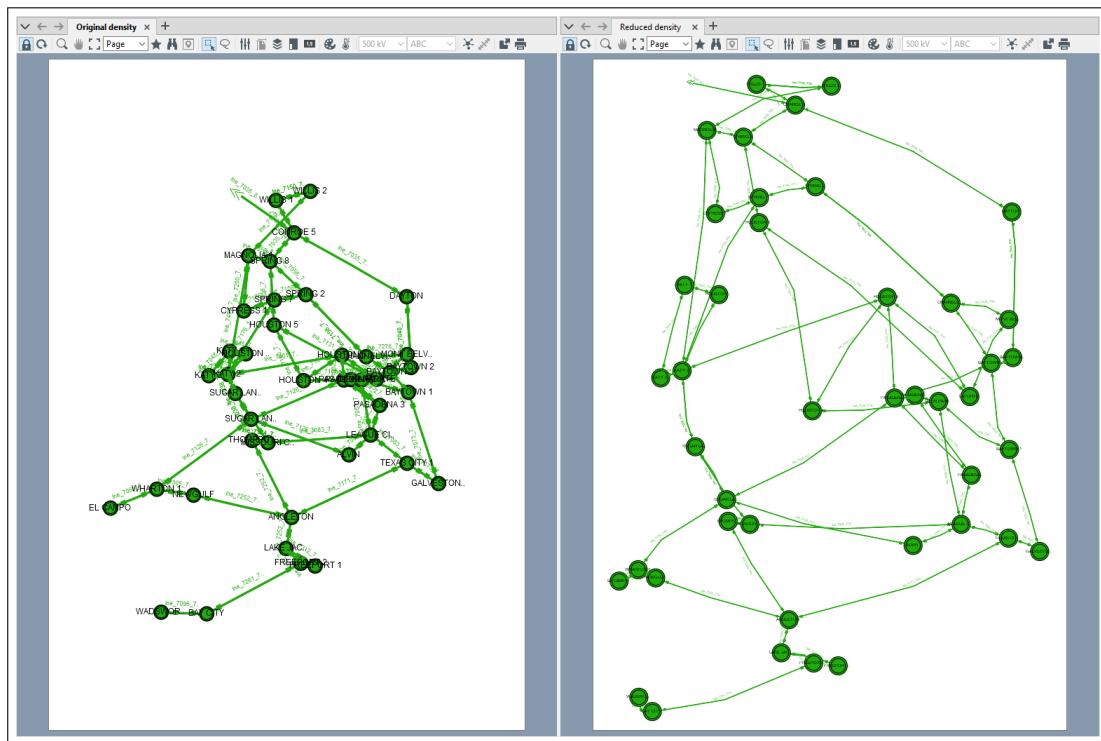


Figure 12.7.1: Single line diagram before and after adjusting the node density

# Chapter 13

## Study Cases

### 13.1 Introduction

The concept of study cases was introduced in Chapter 4 (*PowerFactory* Overview). Study cases (*IntCase*, ) define the studies to be performed on the system being modelled. They store everything created by the user to perform calculations, allowing the easy reproduction of results even after deactivation/reactivation of the project. By means of the objects stored inside them, the program recognises:

- Parts of the network model (grids and expansion stages) to be considered for calculation;
- Calculations (and their settings) to be performed on selected parts of the network;
- Study time;
- Active variations;
- Active operation scenario;
- Calculation results to be stored for reporting;
- Graphics to be displayed during the study.
- The last-selected function toolbox (i.e. which function toolbox was last selected using  on the main toolbar)

A study case with a reference to at least one grid or expansion stage has to be activated in order to enable calculations. A project that contains more than one grid, which has several expansion stages for design alternatives, or which uses different Operation Scenarios to model the various conditions under which the system should operate, requires multiple study cases. All of the study cases in a project are stored inside the study cases folder () in the project directory.

---

**Note:** Only one study case can be active at a given time. When activating a study case, all the grids, Variations and Operation Scenarios that it refers to will also become active.

---

Without study cases, it would be necessary to manually activate the relevant grid and/or expansion stage multiple times in order to analyse the resulting power system configuration. Similarly, it would be necessary to define over and over again the same calculation command setup used to analyse the behaviour of the selected network.

In addition to storing the objects that define a network study, study cases set the units used for the output of calculation results, and allow the definition of specific options for the calculation algorithms.

The following sections describe the main objects stored inside study cases.

## 13.2 Creating and Using Study Cases

When a new project is created, an empty study case is automatically created and activated. This new study case has default settings. The user can later modify these settings using the study case dialog.

The user may define several study cases to facilitate the analysis of projects containing more than one grid, several Expansion Stages, different Operation Scenarios or simply different calculation options. To create a new study case:

- Open the Data Manager and go to the study cases folder. Right-click on the folder and select *New → Study Case...* from the context menu. Enter the name of the new study case in the dialog that pops up and (if desired) modify the default settings.

Only one study case can be active at any given time. To activate or deactivate a study case:

- Open the Data Manager. The active study case and the folder(s) where it is stored are highlighted. Right-click on the active study case and choose *Deactivate* from the context menu. To activate an inactive study case place the cursor on its name, right-click and choose *Activate*. Study cases may also be activated via the Project Overview Window.

A study case can have more than one grid. Only the objects in the active grids will be considered by the calculations. To add an existing grid to the active study case:

- Open the Data Manager and go to the Network Data folder. Right-click the desired grid and select *Activate* from the context menu. The grid will be activated and any relevant graphics will be opened (following a user selection). To remove an active grid, select *Deactivate*.

Variations are considered by a study case when they are activated. The Expansion Stages are applied according to the study case time, which is set by the Study Time stored inside the study case folder. More than one variation can be active for a given study case. However there can be only one recording stage. For further information, refer to Chapter 17 (Network Variations and Expansion Stages). To add (activate) a Variation to the active study case:

- Right-click on the Variation and select *Activate* from the context menu. The Variation will be activated and stages will be highlighted depending on the study time.

An Operation Scenario can be activated or deactivated via the context menu, or by using the option *File → Activate Operation Scenario/Deactivate Operation Scenario* from the main menu. Upon activation, a completeness check is performed (i.e. a check that operational data is available for all components). This is reported in the *PowerFactory* output window. If an Operation Scenario is active, all operational data attributes in property sheets or in the Data Manager are highlighted in blue by default. This indicates that changes to these values will not modify the base component (or Variation) but are recorded by the active Operation Scenario. Upon deactivation, previous operational data is restored. If the Operation Scenario was modified, user confirmation is requested regarding the saving of changes. For further information about working with Operation Scenarios, refer to Chapter 16 (Operation Scenarios).

---

**Note:** Only one study case can be activated at a time. Although network components and diagrams can be edited without an active study case, calculations cannot be performed. Variations and Operation Scenarios used by a study case are automatically activated upon activation of the corresponding study case.

---

### 13.2.1 The Study Case Manager

The Study Case Manager simplifies the management of study cases, by providing an overview of all existing study cases with all active Operation Scenarios, Variations, Grids and Triggers. There is a column for each study case, with the component objects listed on the left, as shown in Figure 13.2.1.

The Study Case Manager can be accessed by clicking on the  icon on the main toolbar or from *Data* → *Study Case Manager* in the main menu. Upon opening the Study Case Manager the active study case will be deactivated, but will be reactivated when the Study Case Manager window is closed.

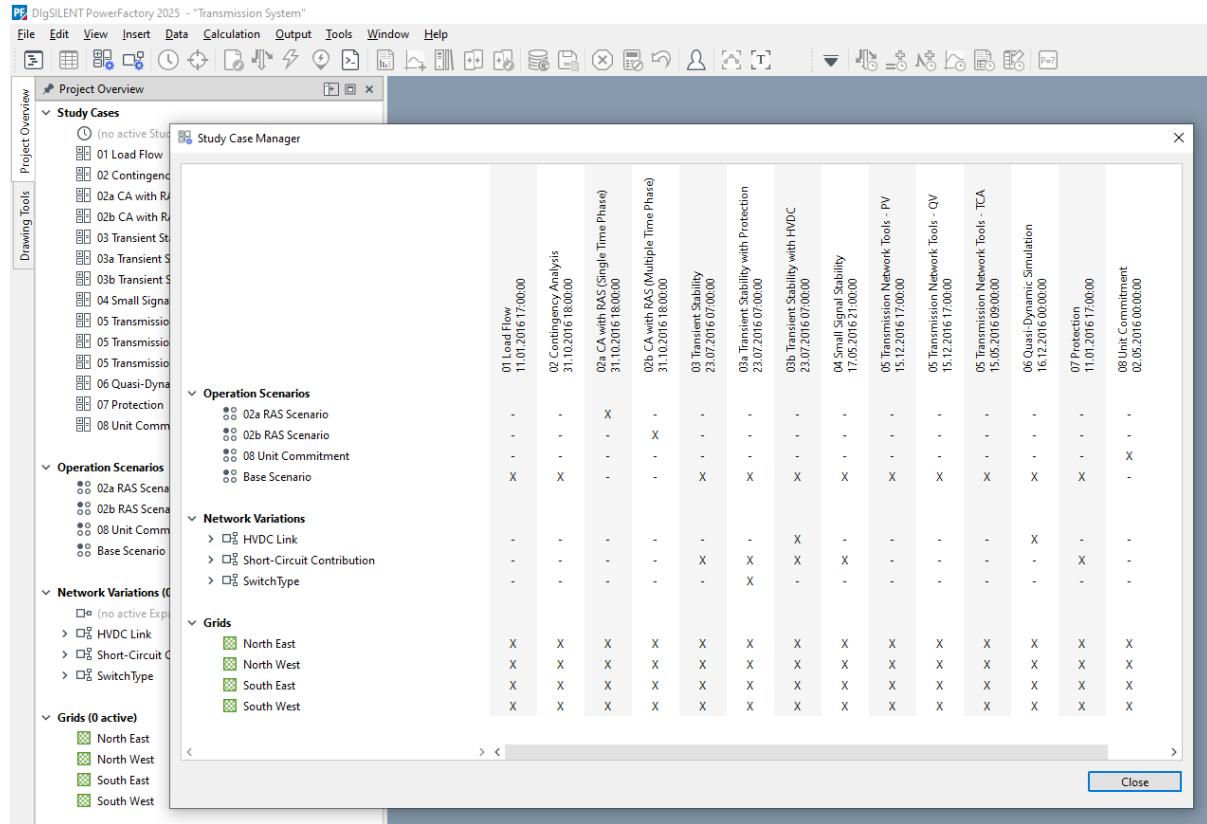


Figure 13.2.1: Study Case Manager

The Study Case Manager not only provides an overview but can also be used to manage the setup of individual study cases, as it allows the activation/deactivation of:

- Operation Scenarios
- Variations
- Grids
- Triggers

simply by double-clicking on the cell entries - without the need to activate the study cases themselves. Since the time of the active study case defines which Expansion Stage is active, it is only possible to activate or deactivate Variations, but not Expansion Stages. Depending on the study time, the recording Expansion Stage will be marked in bold.

**Note:** When folders are used to store study cases, only the study cases in the folder selected (within the Study Case Manager) will be shown.

### 13.3 Summary Grid

The primary task of a study case is to activate and deactivate a calculation target, which is a combination of grids and optionally expansion stages from the network model. The Summary Grid object  holds

references to the grids which are considered in the calculation (i.e. the active grids). Grids may be added to or removed from the currently active study case by right-clicking on them in the database tree or the project overview window and *Activate* or *Deactivate* them. A reference to the activated/deactivated grid is automatically generated in or deleted from the Summary Grid.

A grid cannot be activated without an active study case. With no study case active, the *Activate* action from the context menu of a grid will show a dialog, where a new study case can be created or an existing one can be chosen in order to activate the grid.

## 13.4 Study Time

Study cases have a study time which defines the point in time to analyse.

The study time must be inside the Validity Period of the project, which specifies the time span for which the project is valid (see Chapter 9: Basic Project Definition, Section 9.1.3 (Project Settings)). *PowerFactory* will use the study time in conjunction with time-dependent network expansions (see Chapter 17: Network Variations and Expansion Stages) to determine which network data is applicable at that point in time. The study time may be changed in order to analyse a different point in time. The Expansion Stages will be activated/deactivated in accordance with the study time.

The status bar at the bottom of the *PowerFactory* program window shows the currently study time.

The simplest way to change the study time is:

- Double-click on the *Study Time* shown in the status bar of *PowerFactory*.
- Enter the date and time or press the **Date** and **Time** buttons in order to set the study time to the current time of the computer on which *PowerFactory* is being run.
- Press **OK** and close the window.

There are several alternative ways to edit the study time.

Alternative 1: Edit the study time like a trigger:

- Press the button **Date/Time of Calculation Case** in the main toolbar of *PowerFactory*.
- Enter the date and time or press the **Date** and **Time** buttons in order to set the study time to the current time of the computer on which *PowerFactory* is being run.
- Press **OK** to accept the changes and close the window.

Alternative 2: Edit the study case from within the study case dialog:

- Activate the project and browse for the study case in the Data Manager.
- Right-click on the study case and select *Edit* from the context menu.
- On the *Basic Data* page use the drop-down arrow to bring up a standard calendar menu.
- Set the study time.
- Press **OK** and close the window.

## 13.5 The Study Case Dialog

To edit the settings of a study case, select *Edit* → *Project Data* → *Study Case...* from the main menu, or alternatively right-click on the study case in the Data Manager and select *Edit* from the context menu. A dialog will appear.

### 13.5.1 Basic Data

On the *Basic Data* page, the user can define the name and an owner of the study case. The grids that are linked to a study case may be viewed by pressing the **Grids** button.

The study time can be edited by using the drop-down arrow to bring up a standard calendar menu.

Please note that the study time can also change if the recording expansion stage is set explicitly (see Chapter 17: Network Variations and Expansion Stages).

---

**Note:** To edit the study time one can alternatively press on the “Date/Time of Calculation Case” button . This will open the study time window. In addition, the time of the simulation case is displayed in the lower-right corner of *PowerFactory*. Double-clicking on this field provides access to the same window.

---

### 13.5.2 Calculation Options

The *Calculation Options* page is used to configure the basic algorithm for the study case calculations. The following options are available:

- **General tab:**
  - **Topological processing:** The *breaker reduction* mode determines the internal calculation topology of the grid. In particular, electrically equivalent areas of a detailed substation are identified and merged for an efficient internal treatment. There are four options for the reducing the breakers:
    - \* Enhanced mode: the default option, will also reduce the zero impedance lines when building the electrically equivalent areas.
    - \* Any suitable: will reduce the breakers but not the zero impedances lines.
    - \* Any suitable, except circuit-breakers: similar to the previous one, but circuit breakers will not be reduced.
    - \* None: none of the breakers will be reduced. This option has a dramatic effect on the performance of the calculation and should be avoided.
  - If the check box *Calculate results for all breakers* is ticked, results of reduced elements may then be post-calculated.
  - **Solution of linear equations:** The solution of linear equation systems is an intrinsic part of most calculations in *PowerFactory*, such as load flow, short-circuit or the RMS/EMT simulation. These equation systems can either be solved by a direct factorisation method or an iterative method. The latter method has been developed to meet the increasing demands of modern applications where interconnected, large-scale networks must be analysed. In contrast to traditional direct methods, the implemented iterative solver is able to solve even very large systems with controlled precision.
  - The button **Set all calculation options to default** will restore all default options on the *Calculation Options* page.
  - The other tabs (**Jacobian Matrix**, **Optimisation Matrix**, **Simulation Matrix** and **Advanced**) include additional calculation options to tune the performance and the robustness of the linear system solver.

---

**Note:** Changing the default options on the *Calculation* page is only recommended under the supervision of the *DIGSILENT* support experts.

---

The *Description* page is used for user comments.

## 13.6 Variation Configuration

Similar to the Summary Grid, the Variation Configuration object (*IntAcscheme* ) contains references to the active variations.

## 13.7 Operation Scenarios

A reference to the active Operation Scenario (if any) is always stored in the study case. Similar to Variation Configurations and Summary Grids, when a study case is activated, the Operation Scenario (if any) whose reference is contained, will be automatically activated. The reference to the active Operation Scenario will be automatically updated by the program.

## 13.8 Commands

In *PowerFactory* a calculation (i.e load flow , short-circuit , initial conditions of a time-domain simulation , etc.) is performed via 'Calculation Commands', which are the objects that store the calculation settings defined by the user. Each study case stores its own calculation commands, holding the most recent settings. This ensures consistency between results and calculation commands and enables the user to easily reproduce the same results at a later date. When a calculation is performed in a study case for the first time, a calculation command is automatically created inside the active study case. Different calculation commands of the same class (i.e different Load Flow Calculation commands: objects of the class *ComLdf* or different short-circuit calculation commands: objects of the class *ComShc* ) can be stored in the same study case. This allows the user to repeat any calculation with identical settings (such as fault location, type, fault impedance, etc.) as last performed in the study case. The calculations are only performed on active grids.

Figure 13.8.1 shows a study case, which contains a Load Flow Calculation command () , a command for an OPF calculation , a command for the calculation of initial conditions , and a command to run a simulation . The dialog of each of calculation command in *PowerFactory* is described in the chapter corresponding to that calculation function.

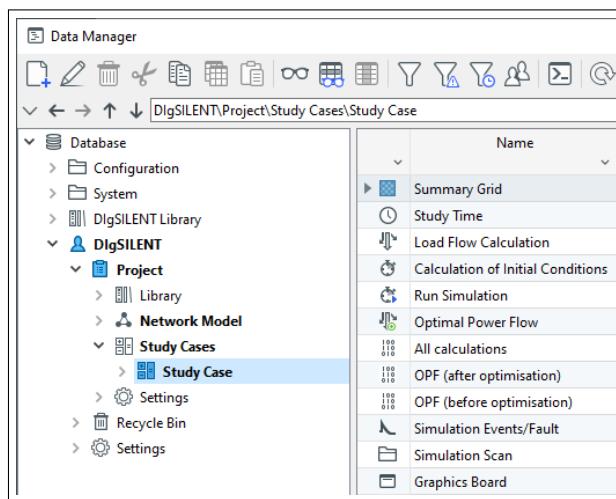


Figure 13.8.1: Calculation commands in a study case

Actions such as generating a report of the actual calculation results or the state of the defined net-

work components are carried out via commands. For information about reporting commands refer to Chapter 19 (Reporting and Visualising Results).

---

**Note:** As with any other object, calculation commands can be copied, pasted, renamed and edited.

---

## 13.9 Events

*Simulation Event* objects are used to define simulation events. For time-domain simulations, events are stored within the *Study Case* → *Simulation Events/Fault* folder (see Chapter 29: RMS/EMT Simulations, Section 29.6 for a general description). For short-circuit studies, they are stored in the *Study Case* → *Short Circuits* folder. For other steady-state calculations that utilise *Simulation Events*, they are stored within the *Operational Library* → *Faults* folder. *PowerFactory* offers several kinds of events:

- Broken Conductor Event (*EvtBrkcond*)
- Dispatch Event (*EvtGen*)
- External Measurement Event (*EvtExtmea*)
- Intercircuit Fault Events (*EvtShcll*)
- Events of Loads (*EvtLod*)
- Message Event (*EvtMessage*)
- Outage of Element (*EvtOutage*)
- Parameter Events (*EvtParam*)
- Save Results (*EvtTrigger*)
- Save Snapshot (*EvtSave*)
- Short-Circuit Events (*EvtShc*)
- Stop Events (*EvtStop*)
- Switch Events (*EvtSwitch*)
- Synchronous Machine Event (*EvtSym*)
- Tap Event (*EvtTap*)
- Power Transfer Event (*EvtTransfer*)

### 13.9.1 Broken Conductor Event

This event type can be used to simulate a broken conductor in a line element. The user can specify which phase is broken and whether specific short-circuit conditions apply on either or both sides of the break. The broken conductor event can be used in the complete short-circuit calculation (if the multiple faults option is enabled in the short-circuit command) and in dynamic simulations.

- **Fault location:** defines at which point of the line the event is applied (seen from terminal i).
- **Conductor interruption:** specifies the phase(s) to be interrupted. Either all phases can be selected or, if this option is disabled, individual phases can be checked/unchecked.
- **Fault at side 1:** check this option to apply a fault at side 1 (towards from terminal i) and configure it accordingly.

- **Fault at side 2:** check this option to apply a fault at side 2 (towards from terminal j) and configure it accordingly.

---

**Note:** In time-domain simulations, to represent the fault location, a dummy terminal is created on the line. For this reason, it is necessary to execute the initial conditions after defining a broken conductor event.

---

### 13.9.2 Dispatch Event

The user specifies the point in time in the simulation for the event to occur, and a generation element (*ElmSym*, *ElmXnet* or *ElmGenstat*) or Ward-Equivalent element (*ElmVac*).

There is the option to define either an incremental change or a move to a specified set-point.

### 13.9.3 External Measurement Event

External measurement events can be used to set and reset values and statuses of external measurements.

### 13.9.4 Intercircuit Fault Event

This type of event is similar to the short-circuit event described in Section 13.9.11. Two different elements and their respective phases are chosen, between which the fault occurs. As for the short-circuit event, four different elements can be chosen:

- Busbar (*StaBar*)
- Terminal (*ElmTerm*)
- Overhead line or cable (*ElmLne*)

---

**Note:** If a line is selected, it is necessary to execute the initial conditions after defining the event. The reason is that to represent the fault location, a dummy terminal is created on the line during the initial conditions.

---

### 13.9.5 Load Event

The user specifies the point in time in the simulation for the event to occur, and a load or set of load element(s) (*ElmLod*, *ElmLodlv*, *ElmLodmv* or *ElmLodlvp*). Optionally a set of loads *SetSelect* can be also selected. The value of the load (s) can then be altered using the load event. The power of the selected load(s) can be changed as follows:

- **Step** Changes the current value of the power (positive or negative) by the given value (in % of the operating power of the load) at the time of the event.
- **Ramp** Changes the current value of the power by the given value (in % of the operating power of the load), over the time specified by the *Ramp Duration* (in seconds). The load ramping starts at the time of the event.

### 13.9.6 Message Event

A message will be printed to the output window at the specified time in the simulation.

### 13.9.7 Outage Event

The *Outage Event* can be used to take an element out of service at a specified point in time. It is intended for use in steady-state calculations (e.g. short-circuit calculations and reliability assessment) and dynamic simulation.

**Out of service:** check/uncheck this flag in order to enable/disable this event.

**Execution Time:** defines the simulation/calculation time when this event is to be executed.

**Element:** the target element of this event is to be assigned here.

**Type of Outage Event:**

- *Take element out of service* - the *Element* is set to out of service at the defined *Execution Time*;
- *Bring element back into service* - the *Element* is put back in service at the defined *Execution Time* (this option is not supported in dynamic simulation);
- *Reinsert all outaged elements* - all elements outaged by previously executed outage events are put back in service (this option is not supported in dynamic simulation);
- *Take element and its controls out of service* - the *Element* is set to out of service at the defined *Execution Time* along with all associated controls. Associated controls are all DSL elements (and only those ones) which are referenced into a slot of a composite model whose *Main slot* is occupied by the target *Element*.

---

**Note:** The event can be used to take elements out of service in time-domain simulations, however it is not possible to bring an outaged element back into service using this event during a transient simulation. This is only possible in steady-state calculations. The following message will be displayed if the user attempts to bring a previously-outaged element back into service using *Outage of Element*:

```
t=000:000 ms - Outage Event in Simulation not available.  
Use Switch-Event instead!
```

---

### 13.9.8 Parameter Event

With this type of event, an input parameter of any element or DSL model can be set or changed. First, a time specifying when the event will occur is specified. An element or set of elements *SetSelect* must then be specified/selected using the *down-arrow* button . Choose *Select...* from the context menu, and insert the name and the new value of the element parameter. In case a selection is used all elements within the selection have to share the same element parameter.

### 13.9.9 Save Results

This event is only used for *PowerFactory* Monitor applications. It cannot be used during time-domain simulations.

### 13.9.10 Save Snapshot

This event is used in a time-domain simulation whenever the current simulation state (a simulation snapshot) is to be saved. Further information on how to configure, save and load a simulation snapshot can be found in Section [29.10](#).

Event options:

- *Out of service* - Check this flag in order to disable its execution. Un-check this flag in order to activate it.
- *Execution time* - define the Absolute/Relative execution time of this event. The Absolute time reference is 0 seconds. The relative time reference is the current simulation time (if the simulation is reset initialised, then the relative time fields are not available).

### 13.9.11 Short-Circuit Event

This event applies a short-circuit on a busbar, terminal or specified point on a line. The fault type (three-phase, two-phase or single-phase fault) can be specified, as can the fault resistance and reactance and the phases which are affected.

Note that during EMT simulation, a short circuit event is ignored if fault resistance or reactance is negative.

For the lines, the location of the short circuit is defined on the event. Internally, a dummy terminal is created on the line to represent the short circuit. For this reason, it is necessary to execute the initial conditions after defining the short circuit event.

The duration of the fault cannot be defined. Instead, to clear the fault, another short-circuit event has to be defined, which will clear the fault at the same location.

### 13.9.12 Stop Event

Stops the simulation at the specified time within the simulation time frame.

### 13.9.13 Switch Event

To create a new switch event, press the  icon on the main menu (if this icon is available), which will open a browser containing all defined simulation events. Click on the  icon in the browser and select *Switch Event*.

After pressing **OK**, the reference to the switch (labelled *Breaker* or *Element*) must be manually set. Any switch in the power system may be selected, thus enabling the switching of lines, generators, motors, loads, etc. The user is free to select the switches/breakers for all phases or only those for one or two phases.

It should be noted that more than one switching event must be created if, for instance, a line has to be opened at both ends. These switch events should then have the same execution times.

### 13.9.14 Synchronous Machine Event

The *Synchronous Machine Event* is used to change the mechanical torque of a synchronous machine (*ElmSym*) in a simple manner. The user specifies the point in time in the simulation for the event to occur, and an active synchronous machine. The user can then define the additional mechanical torque supplied to the generator. The torque can be positive or negative and is entered in per-unit values.

### 13.9.15 Tap Event

The user specifies the point in time in the simulation for the tap event to occur, and a shunt or transformer element (*ElmShnt*, *ElmTr2*, etc). The *Tap Action* can then be specified.

### 13.9.16 Power Transfer Event

The Power Transfer event (*EvtTransfer*) is used to transfer demand from a load object, or output from static generator, to other load objects or static generators respectively. The user specifies the source object (*ElmLod* or *ElmGenstat*) and the destination object or objects, which must be of the same class. Then separate percentage figures are input for active and reactive power. If more than one Power Transfer event is to be used, it is important to consider the order in which they will be executed, as this could affect the final outcome. Power Transfer events may be used in RMS simulations, Outage Management and Faults Cases for contingency analysis.

## 13.10 Simulation Scan

For details of Simulation Scan modules, refer to Chapter 29: RMS/EMT Simulations, Section 29.9.

## 13.11 Results Objects

The results object (*ElmRes* ) is used to store tables with the results obtained after the execution of a command in *PowerFactory*. Results Files are described in chapter ch:ReportingResults: Reporting and Visualising Results, Section 19.7.

For Contingency Analysis, the results object can optionally contain a filter (*SetFilt*), to restrict the recording of results to a specified part of the network. The use of such a filter is described in the Contingency Analysis chapter, in Section 27.10.1.

## 13.12 Triggers

As described in Chapter 18 (Parameter Characteristics, Load States, and Tariffs), parameter characteristics are used to define parameters as ranges of values instead of fixed amounts. The parameter characteristics are set over user defined scales. The current value of the parameter is at the end determined by a trigger object (*SetTrigger*, , which sets a current value on the corresponding scale. For example if the value of a certain parameter depends on the temperature, a characteristic over a temperature scale is set. The current value of the temperature is defined by the trigger. The current value of the temperature determines the current value of the parameter, according to the defined characteristic.

Once a parameter characteristic and its corresponding scale are set, a trigger pointing to the scale is automatically created in the active study case. The user can access the trigger and change its value as required.

*PowerFactory* offers different types of characteristics and scales, and each scale points to a trigger from the active study case. By default, scales are stored in the Scales folder within the *Characteristics* folder in the *Operational Library*, and triggers are stored in the Triggers folder of the study case, the folder being automatically created if required. Information regarding the use and definition of characteristics, scales and triggers is given in Chapter [Parameter Characteristics, Load States, and Tariffs](#).

## 13.13 Desktop

The study case has a folder called the *Desktop* (*SetDesktop*), which contains graphic page objects that are linked to the graphics to be displayed. The desktop is automatically created and maintained and should generally not be edited by the user.

Graphic pages and other pages (tabs), such as the Data Manager or reports, can be freely moved from one docked or floating window to another. The organisation of these tabbed windows is done automatically via Tab Group objects (*SetTabgroup*), also held within the desktop.

The references in the *Desktop* folder are created when the user activates a grid on a active study case. The user can display other graphics by right-clicking in a network diagram and selecting *Mark in Other Graphic or Diagrams* → *Show Detailed Diagram*.

Graphics can be removed by right-clicking the tab at the top of the page and selecting *Close tab*.

More information about the desktop and working with tab groups can be found in sections [4.7.2](#) and [4.7.3](#).

# Chapter 14

## Libraries

### 14.1 Introduction

There are three types of libraries in *PowerFactory*:

- [14.2 DlgSILENT Library](#)
- [14.4 Project Library](#)
- [14.3 Custom Global Library](#)

The organisation of the libraries is similar for all the library types. Therefore in the following sections, after the main libraries are described, each of the subfolders included in the *DlgSILENT* and project library are described.

### 14.2 DlgSILENT Library

A global library (called “*DlgSILENT Library*”) is supplied with *PowerFactory*; it contains a wide range objects and is accessible to all users. When a database is migrated to a new *PowerFactory* version, the *DlgSILENT* Library will be refreshed with all the new and updated information and if a user were to have made changes and additions, all that information would be lost, which is why it is not recommended to modify this library.

The *DlgSILENT* Library contains:

- Composite model frames, DSL and Modelica models, DSL macros (i.e. transfer functions and logic blocks, etc.) and Modelica model types. Section [Dynamic Models Library](#).
- Type data for standard components such as conductors, motors, generators, and transformers. Section [Equipment Type Library](#).
- Harmonic current spectrums and frequency characteristics. Section [Harmonics Library](#).
- Operational data (e.g. predefined characteristics for loads, coincidence definitions). Section [Operational Library](#).
- Models and type data for protection devices. Section [Protection Devices Library](#).
- Quasi-Dynamic models. Section [Quasi-Dynamic Models Library](#).
- Pre-defined report templates. Section [Reports Library](#).
- Standard scripts. Section [Scripts Library](#).

- Pre-defined model templates. Section [Templates Library](#).

### 14.2.1 Versioning in the *DIGSILENT* Library

All the main objects (types, models etc.) in the *DIGSILENT* Library are managed using versions. The versioning distinguishes between major and minor versions. For example a type object might be at v002.1, where the main version number (002 in this example) is incremented when a comprehensive change to the object is made, and the minor version number (.1 in this example) is incremented when a small change that will not affect calculation results is made. Elements that are no longer maintained by *DIGSILENT* are marked as obsolete.

A version page shows the version number and a *Change Log* to indicate the modification history.

When the *DIGSILENT* global library is updated, for example due to the installation of a new version of *PowerFactory*, all minor version changes will be implemented automatically. This means that a new minor version will replace the previously released version of an element. If a new major version of an element is released, the corresponding previous versions will be automatically retained in subfolders.

## 14.3 Custom Global Library

When several users work for the same company, there is often a need to share a custom library between these users. Instead of copying the library from user to user, a more convenient approach is to create an additional library in the global area which will be visible for all the users.

The custom library should be defined by the Administrator as follows:

- Right-click on the “Database” folder
- Select *New* → *Library...* from the context menu
- Choose a library name and click **OK**

The new library is now created at the same level of the hierarchy as the *DIGSILENT* global library.

The user groups with access to the library, and their level of access, are defined on the *Sharing* page of the edit dialog of the library.

Several libraries can be created, but only one can be edited by a user at a time. In order to edit the library, it has to be active; the library is activated using *right-click* → *Activate*.

Once an additional global library is added, it should be set in the User Settings as *Used Library* (see chapter User Settings, Section [7.2](#)).

The versioning of the Custom Global Library works in exactly the same way as for the *PowerFactory* project, described in Section [21.2](#): Project Versions.

## 14.4 Project Library

The project library stores the following categories of data:

- Dynamic models (see Sections [14.10](#): Dynamic Models Library and [14.11](#): Quasi-Dynamic Models Library)
- Equipment types (see Section [14.5](#): Equipment Type Library)
- Operational data (see Section [14.6](#): Operational Library)

- Reports (see Section [14.7: Reports Library](#))
- Scripts (see Section [14.8: Scripts Library](#))
- Templates (see Section [14.9: Templates Library](#))

Unlike global libraries, which are accessible to all users, the project's local library stores project-specific data. It can only be used by the project owner and users sharing the project.

## 14.5 Equipment Type Library

The *Equipment Type Library* is used to store and organise type data for each class of network component.

In the *DlgSILENT* Library, folder *Equipment Types*, the following element types are available:

- Busbar Trunking Systems
- Busbars
- CTs (current transformers)
- Cables
- Conductors
- Induction Machines
- Loads
- PV Panels
- Switches
- Synchronous Machines
- Towers
- Transformers
- VTs (voltage transformers)

For the project, once a new project is created, an *Equipment Type Library* is automatically set by the program within the *Library* folder inside the project.

To create or edit a folder in the *Equipment Type Library* of an active project:

1. Right-click on the *Equipment Type Library* folder in the left pane of the *Data Manager* and select *New → Project Folder...* from the context menu (or to edit an existing folder, right-click the folder and select *Edit*). The project folder edit dialog is displayed.
2. In the *Name* field, enter the name of the new folder.
3. In the *Folder Type* field, select *Generic*.
4. In the *Class Filter* field, write the name of the type class(es) to be allowed in the folder (case sensitive). If more than one class is to be allowed, write the class names separated by commas. An asterisk character (\*) can be used to allow all classes.
5. In the *Icon* field, select *Library*.

To create new type objects in the library folders, select the *New Object* icon  and the appropriate type class; alternatively, types can be copied from other projects or the *DlgSILENT* library.

---

**Note:** If a class filter is applied to the library folder, only the objects belonging to the class will be shown in the *New Object* dialog. An error message is displayed when pasting objects with a class different than the defined one the type class.

Figure 14.5.1 shows the equipment library of a project containing generators, loads, and transformers types, sorted using library sub-folders.

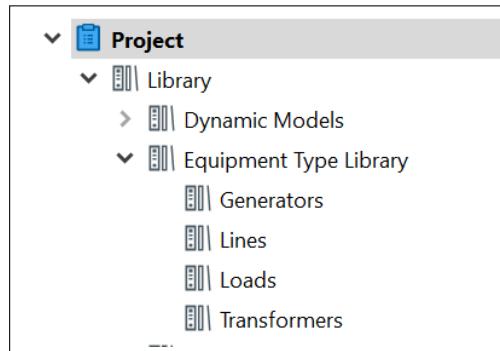


Figure 14.5.1: The Equipment Library

There are two options available for defining the type data for network components:

1. *Select Type*: the *Data Manager* pointing to the *DlgSILENT* Library, shown in Figure 14.5.2, is launched. The **Project Library** button can be used to quickly switch between the global and local libraries.

**Note:** If an additional library is defined in the global area (see Section 14.3) and set in the User Settings (see Section 7.2), an additional button will be available in the dialog

2. *New Project Type*: a new type will be defined and automatically stored in the local *Equipment Type Library*.

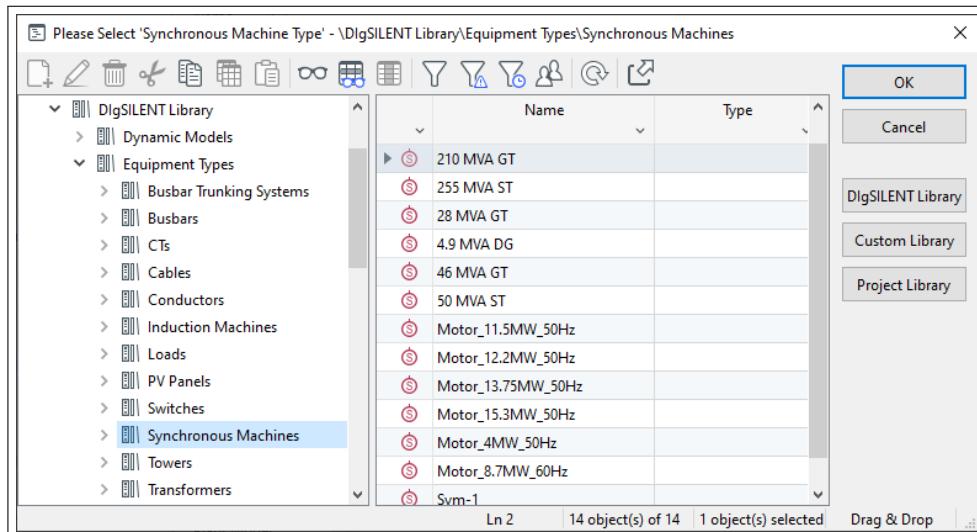


Figure 14.5.2: Libraries buttons shown when assigning a type

## 14.6 Operational Library

The *Operational Library* is used to store and organise operational data for application to a number of elements, without the need to duplicate operational information.

To illustrate, consider an example where there are two generators, “G1” and “G2”. The units have slightly different type data, and thus unique type models. The Mvar Limit Curves for these units are identical, and so the user wishes to create only a single instance of the capability curve. By defining a *Capability Curve* in the *Operational Library*, a single capability curve can be linked to both generators.

Similarly, various circuit breakers may refer to the same short-circuit current ratings. A *Circuit Breaker Rating* object can be defined in the *Operational Library* and linked to relevant circuit breakers.

Characteristics can also be assigned to many element parameters. The use of Characteristics is described in detail in Chapter 18. Such characteristics are normally held in the Characteristics folder of the Operational Library.

The *DIGSILENT* Library includes a number of standard load characteristics within the folder *Operational Data* → *Characteristics* → *BDEW Profiles*, as well as coincidence definitions for electrical vehicles, households and heatpumps within the folder *Operational Data* → *Coincidence Definitions*.

The Operational Library inside the project library contains the following folders:

- CB Ratings (see Section 14.6.1: [Circuit Breaker Ratings](#))
- Characteristics (see Chapter 18: [Parameter Characteristics, Load States, and Tariffs](#))
- Coincidence Definitions (see Section 14.6.2: [Coincidence Definitions](#))
- Common Mode Failures (see Section 46.3.1.6: [Common Mode Stochastic Model \(StoCommon\)](#))
- Demand Transfers (see Section 14.6.3: [Demand Transfers](#))
- Faults (see Section 14.6.4: [Fault Cases and Fault Groups](#))
- Generator Cost Curves (see Section 40.5.2.1, Chapter 40: [Unit Commitment and Dispatch Optimisation](#))
- Generator Efficiency Curves (see Section 40.6.1, Chapter 40: [Unit Commitment and Dispatch Optimisation](#))
- Mvar Limit Curves (see Section 14.6.5: [Capability Curves \(Mvar Limit Curves\) for Generators](#))
- Outages (see Section 14.6.6: [Planned Outages](#))
- Probabilistic Assessment (see Chapter 45: [Probabilistic Analysis](#))
- QP-Curves (see Section 14.6.8: [QP-Curves](#))
- QV-Curves (see Section 14.6.9: [QV-Curves](#))
- Remedial Action Schemes (RAS) (see Section 14.6.10: [Remedial Action Schemes \(RAS\)](#))
- Running Arrangements (see Section 14.6.11: [Running Arrangements](#))
- Tariffs (see Chapter 18: [Parameter Characteristics, Load States, and Tariffs](#))
- Thermal Ratings (see Section 14.6.12: [Thermal Ratings](#))
- V-Control-Curves (see Section 14.6.13: [V-Control-Curves](#))

### 14.6.1 Circuit Breaker Ratings

Circuit Breaker Ratings objects  (*IntCbrating*) contain information that define the rated short-circuit currents of circuit breakers (objects of class *ElmCoup*). They are stored inside the *CB-Rating* folder in the *Operational Library*. Any circuit breaker (*ElmCoup*) defined in the Network Model can use a reference to a Circuit Breaker Rating object in order to change its current ratings.

The parameters defined by a circuit breaker rating are:

- Three phase initial peak short circuit current.
- Single phase initial peak short circuit current.
- Three phase peak break short circuit current.
- Single phase peak break short circuit current.
- Three phase RMS break short circuit current.
- Single phase RMS break short circuit current.
- DC time constant.

Additionally, the values for the *Bay Infrastructure Ratings* can be defined. These represent the ratings for the structural components of the circuit breaker mounting.

- Three phase initial peak short circuit current
- Single phase initial peak short circuit current
- Three phase RMS break short circuit current
- Single phase RMS break short circuit current

---

**Note:** The complete ratings for circuit breakers are not used for any calculations in *PowerFactory*. The entries are intended for informational and scripting purposes. Similarly, this applies to the parameters belonging to the *Complete Ratings* option of Switch Types (*TypSwitch*).

---

To create a new circuit breaker rating in the operational library:

- In the Data Manager open the *CB Ratings* folder.
- Click on the *New Object* icon .
- Select Circuit Breaker Rating (*IntCbrating*) and press **Ok**.
- The new circuit breaker rating dialog will then be displayed. Set the corresponding parameters and press **Ok**.

To assign a circuit breaker rating to a circuit breaker (*ElmCoup* object) from the network model:

1. Go to the *Short-Circuit Complete* page of the element's dialog.
2. In the Ratings field click on the  button to select the desired rating from the *CB Ratings* folder.

The parameters defined in the circuit breaker ratings can be made to be time-dependant by means of variations and expansion stages stored inside the *CB Ratings* folder.

For information regarding short-circuit calculations, refer to Chapter [Short-Circuit Analysis](#). For further information about variations and expansion stages, refer to Chapter [Network Variations and Expansion Stages](#).

---

**Note:** Variations in the CB Ratings folder act “locally”, they will only affect the circuit breaker ratings stored within the folder. Similarly, the variations of the Network Model will only affect the network components from the grids.

---

**Note:** Circuit breaker elements (*ElmCoup*) must be distinguished from Switch objects (*StaSwitch*); the latter are automatically created inside cubicles when connecting an edge element (which differs to a circuit breaker) to a terminal. Ratings can also be entered in the *TypSwitch* object.

---

#### 14.6.1.1 Example Time-Dependent Circuit Breaker Rating

Consider an example where a substation circuit breaker “CB” operates with different ratings depending on the time of the year. From 1st January to 1st June it operates according to the ratings defined in a set of parameters called “CBR1”. From 1st June to 31st December it operates with the ratings defined in a set of parameters called “CBR2”.

This operational procedure can be modelled by defining a circuit breaker rating “CBR” in the *CB Ratings* folder, and a variation “CB\_Sem\_Ratings” containing two expansion stages. The first expansion stage should activate on the 1st January and the second on the 1st June. The first task is the definition of the time-dependant circuit breaker rating “CBR”. To set the parameters of “CBR” for the first period:

1. Set a study time before the 1st June to activate the first expansion stage (the Variation “CB\_Sem - Ratings” must be active).
2. Edit the parameters of “CBR” (previously defined) according to the values defined in “CBR1”. The new parameters will be stored in the active expansion stage.
3. To set the parameters of “CBR” for the second period:
4. Set a study time after the 1st June to activate the second expansion stage;
5. Edit “CBR” according to the values of “CBR2”. The new parameters will be stored in the active expansion stage.

Once the ratings for the two expansion stages have been set, and the circuit breaker rating “CBR” has been assigned to the circuit breaker “CB”, the study time can be changed from one period to the other to apply the relevant ratings for “CB” (note that the variation must be active).

#### 14.6.2 Coincidence Definitions

*Coincidence Definitions* (objects class *IntCoincdef*) describe the simultaneity of load demand, for example of households or of electric vehicle charging. *Coincidence Definitions* are typically used in low voltage power system analysis to assess maximum load demands and associated voltage drops, see Section 42.5.

#### 14.6.3 Demand Transfers

Note that Demand Transfers make use of the *IntOutage* object, which has now been superseded by the new *IntPlannedout* object described in Section 14.6.7. Therefore, users wishing to create *IntOutage* objects will need to enable a project setting: on the Project Settings, Miscellaneous page select *Create IntOutage (obsolete)*.

The active and reactive power demand defined for loads and feeders in the network model can be transferred to another load (or feeder) within the same system by means of a *Demand Transfer* (objects

class *IntOutage*). This transfer only takes place if it is applied during a validity period defined by the user (i.e. if the current study time lies within the validity period).

To create a new load demand transfer:

1. In the Data Manager, open the *Demand Transfer* folder.
2. Click on the *New Object* icon 
3. Select *Planned Outage (IntOutage)* and press **Ok**.
4. Set the validity time, the source and target loads/feeders and the power transfer.

---

**Note:** If there is a demand transfer, which transfers load between two loads (*ElmLod*) belonging to different feeders (*ElmFeeder*), then the same MW and Mvar value is transferred from one feeder to the other.

---

A demand transfer is only possible if an active operation scenario (to record the changes) is available. The **Apply all** button will automatically apply all transfers that are stored in the current folder and which fit into the current study time. Before execution, the user is asked if the current network state should be saved in a new operation scenario. The same demand transfers can be applied as many times as desired during the validity period.

If a non-zero power transfer has been executed and the source's power is less than zero, a warning is printed to the output window indicating that the power limit has been exceeded. The applied transfers can be reverted by using the **Reset all** button.

When the current operation scenario is deactivated, all load transfers executed while the operation scenario was active will be reverted.

For information about operation scenarios refer to Chapter [Operation Scenarios](#).

## 14.6.4 Fault Cases and Fault Groups

This section discusses the data structure of the *Faults* folder, and the objects contained within it. The functionality of Event objects is described in Section [29.6: Events \(IntEvt\)](#).

The *Faults* folder stores two types of subfolders:

1. *Fault Cases* folders, which in turn store objects that represent *Simulation Events* or *Faults*   
*Simulation Events/Fault* may contain a number of individual *Events* (*Evt\**), e.g. short-circuits events, switching events.
2. *Fault Groups* folders store *Fault Groups* (*IntFaultgrp*) objects, which in turn reference *Fault Cases*.

The following sections provide a details of how to define *Fault Cases* and *Fault Groups*.

### 14.6.4.1 Fault Cases

A fault case can represent a fault in more than one component, with more than one event defined. There are two types of Fault Cases:

1. **Fault cases without switch events (Type 1):** Independent of the current topology and only stores the fault locations. The corresponding switch events are automatically generated by the contingency analysis tools.

2. **Fault Case with at least one switch event (Type 2):** A Fault Case of Type 2 predefines the switch events that will be used to clear the fault. No automatic generation of switch events will take place.

To create new fault case (object of class *IntEvt*):

1. Multi-select the target components on a single line diagram.
2. Right-click and select *Operational Library* → *Fault Cases* from the context menu.
3. Select from the following options:
  - **New Single Fault Case:** This creates a single simultaneous fault case including all selected elements. A dialog box containing the created fault case is opened to allow the user to specify a name for the fault case. Press **Ok** to close the dialog and saves the new fault case.
  - **New Multiple Fault Cases, n-1:** This creates an n-1 fault case for each selected component. Therefore the number of fault cases created is equal to the number of components selected. The fault case is automatically created in the database after selection.
  - **New Multiple Fault Cases, n-2:** This creates an n-2 fault case for each unique pair among the selected components. Therefore the number of fault cases is  $(b \cdot (b - 1)/2)$  where "b" is equal to the number of selected components. The fault case is automatically created in the database after selection.
  - **New Mutually Coupled Lines/Cables, n-k:** This creates fault cases considering the simultaneous outage of each coupled line in the selection.

The fault cases created will consist of short-circuit events applied to the selected components. All breakers (except for circuit breakers, which are used to model a circuit breaker failure) will be ignored.

- If only breakers are included in the selection, an error message will be issued.
- If a simple switch (not a circuit breaker) is included in the selection, a warning message will be issued that this switch will be ignored.
- If a circuit breaker is contained in the selection, then an *Info* message will be issued, that the CB will be used for modelling a CB failure and will not be handled as a fault location.

---

**Note:** In the case that a branch is selected, the short-circuit event is generated for a (non-switch device with more than one connection) component of the branch. The component used in the event is: "Connection 1" if suitable, otherwise "Connection 2" if suitable, otherwise a suitable random component of the branch (line, transformer ...).

---

#### 14.6.4.2 Fault Groups

New *Fault Groups* are created in the Data Manager as follows:

1. Open the target *Fault Groups* folder and select the *New Object* icon .
2. Select *Fault Group* from the list and click **OK**.
3. In the edit dialog, specify the name and use the button **Add Cases** to add Simulation Events/Fault (*IntEvt*) to the *Fault Group*.

#### 14.6.5 Capability Curves (Mvar Limit Curves) for Generators

Reactive Power operating limits can be specified in *PowerFactory* through definition of *Capability Curves*  (*IntQlim*). They are stored in the *Operational Library*, within the *Mvar Limit Curves* folder. Synchronous generators (*ElmSym*) and static generators (*ElmGenstat*) defined in the network model can

use a pointer to a capability curve from the *Load Flow* page of their edit dialog. When executing a *Load Flow* (with *Consider Reactive Power Limits* selected on the *Basic Options* page) the generators' reactive power dispatch will be limited to within the extends of the defined capability curve. For information about the dispatch of synchronous generators, refer to the synchronous machine technical reference in the [Technical References Document](#). For information on Load Flow calculations and reactive power limits, refer to Chapter 25 (Load Flow Analysis).

---

**Note:** If *Consider active power limits* is selected on the *Basic Options* page of the Load Flow Calculation command, active power is limited to the lesser of the *Max. Active Power Operational Limits* and the *Max. Active Power Rating* specified on the synchronous machine *Load Flow* page.

---

#### 14.6.5.1 Capability Curves *IntQlim*

##### Basic Data

The *Basic Data* page consists of tables to enter the reactive power capabilities as well as an automatically generated graphical representation of the given curves. The input table changes based on whether the voltage dependency of the reactive power capability is considered or not. This can be set via *Configuration* → *Consider voltage dependent limits*. The values for the curves can either be entered as absolute values (i.e. MW and Mvar) or in per-unit, which can be selected on the *Configuration* page as well.

- **Without voltage dependent limits:** In this case, the user can provide the minimum and maximum reactive power limits at certain active power setpoints in a table with the title *Capability Curve*. The active power must be entered in ascending order. The reactive power limits between two row entries are linearly interpolated.
- **With voltage dependent limits:** If this setting is activated, the user can enter the capability curve using the *Matrix for Qmin* and the *Matrix for Qmax*. The rows containing the voltage reference points in p.u. and columns containing the active power reference points are set on the *Configuration* page. The capability curve is graphically depicted for each voltage level. The limits are linearly interpolated depending on the operating voltage and active power of the network element.

---

**Note:** The breakpoints in the curve at the voltage levels set in the reactive power capability curve definition are smoothed for load flow calculations. This may lead to slight deviations close to the reference voltages. Additional precision can be achieved by adding supporting voltages close to the reference voltage with the respective linearly interpolated reactive power limits.

---

##### Configuration

This page contains the configuration for the data input on the *Basic Data* page.

- **Input Model:** This switches the input for the values of the active and reactive power defined on the *Basic Data* page between absolute (i.e. MW and Mvar) or per-unit (p.u.) values.
- **Consider voltage dependent limits:** This option allows to consider the voltage dependency of the reactive power capability curves. By activating this setting, the user can enter the voltage reference points in ascending order as *Rows*. Additionally, the active power set points, which were previously set on the *Basic Data* page, are now entered on the *Configuration* page as *Columns*.

---

**Note:** The per-unit values for the active and reactive power of the capability curves refer to the nominal apparent power of the respective network element. If the element consists of multiple parallel units

(i.e. Number of parallel units (*nnum*) on the *Basic Data* page of the generator greater than one) the absolute power limit is applied to each individual unit.

---

#### 14.6.5.2 Defining Capability Curves

To define a new generator *Capability Curve*:

1. Open the folder *Mvar Limit Curves* from the *Operational Library*.
2. Click on the *New Object* icon  and select *Capability Curve*. The new capability curve dialog will be displayed.
3. Right-click on *Append Row(s)* to add the required number of rows to the table and enter data points to define the generation limits.
4. To apply a *Capability Curve* to a generator:
  - Locate the *Reactive Power Limit* section on the *Load Flow* page of the synchronous machine's or static generator's dialog.
  - Press  next to *Capability Curve*.
  - Choose *Select* and then select the required curve in the *Mvar Limit Curves* folder of the *Operational Library* (the required curve can also be created at this step by selecting the *New Object* icon .
5. Select a capability curve and press **OK**.

Capability curves are included in operation scenario subsets; meaning that if a capability curve is selected/reset from a generator when an operation scenario is active, the change will be stored in the operation scenario. Once the operation scenario is deactivated, the assignment/reset of the curve is reverted. For information on working with operation scenarios, refer to Chapter 16 (Operation Scenarios).

To enter a capability curve for information purposes only (i.e. a capability curve which is not to be considered by the calculation), enter it on the Advanced tab of the *Load Flow* page. Then select *User defined Capability Curve* and enter the curve as a series of points in the table. Right-click on the rows to append, delete or insert new rows.

#### 14.6.5.3 Defining a Variation of a Capability Curve

Similar to circuit breaker ratings (see Section 14.6.1 (Circuit Breaker Ratings)), *Capability Curves* can become time-dependant by means of variations and expansion stages stored inside the *Mvar Limit Curves* folder.

To create a time-dependent variation for a *Capability Curve*, navigate to the *Mvar Limit Curves* folder in the left pane of a Data Manager window. Right-click on the folder and select *New → Variation...*. Name the variation, press **OK**, name the Expansion Stage, and press **OK**. Changes to Capability Curves are recorded in the active expansion stage.

To activate a variation of a *Capability Curve*, open the Data Manager. Right-click the *Variation object*  in the *Mvar Limit Curves* folder and select *Activate*.

For general information about variations and expansion stages refer to Chapter 17 (Network Variations and Expansion Stages).

### 14.6.6 Planned Outages

Planned Outage objects (*IntPlannedout*) are normally stored in the Outage folder of the Operational Library. They can be applied to an active study case to model expected outages of network elements for maintenance, network expansion etc. Figure 14.6.1 shows the dialog box of a Planned Outage object, illustrating the following features:

- Start and End date of the period for which the Planned Outage is valid.
- Outaged components.
- Buttons to apply and reset the outage, view the events and record additional events.

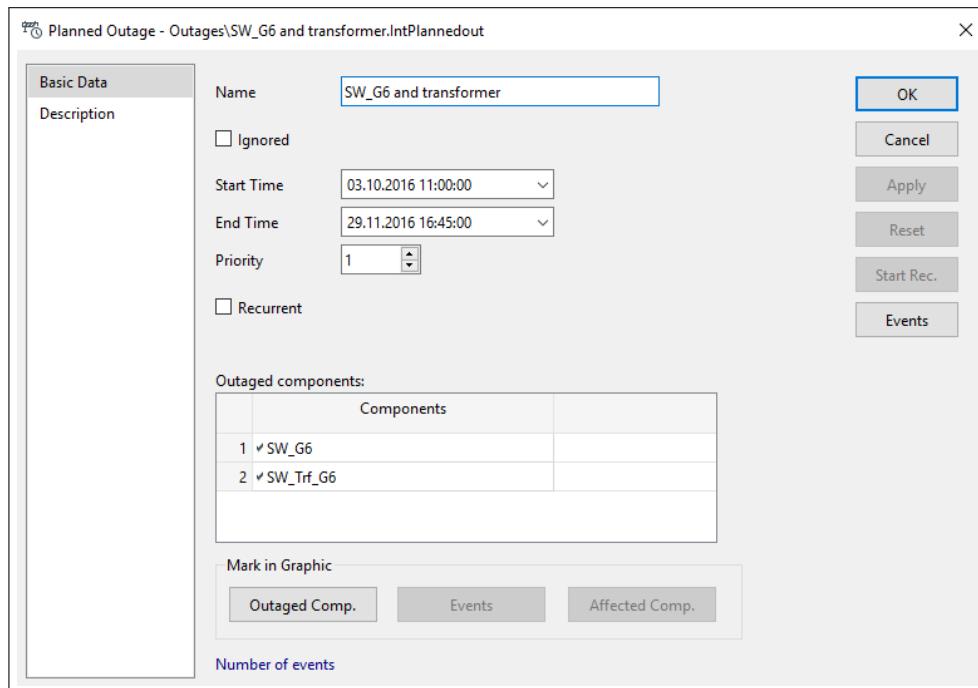


Figure 14.6.1: Planned Outage object dialog

Changes to switch positions and other parameters resulting from the application of Planned Outages will be taken into account for all calculations but are only effective as long as the study case is active. A new toolbar has been provided for the handling of Planned Outages. Please see Chapter 43 for how to create Planned Outage objects and handle them via the toolbar.

### 14.6.7 Planned Outages *IntOutage*

Note that this subsection refers to the original *IntOutage* object, which is now superseded by the *IntPlannedout* object described in Section 14.6.6. Users wishing to create *IntOutage* objects will need to enable a project setting: on the Project Settings, Miscellaneous page select *Create IntOutage (obsolete)*.

A *Planned Outage* is an object used to check and/or apply an *Outage of Element* or *Generator Derating* over a specified time period. *Planned Outages* are stored within the *Operational Library* in the *Outages* folder.

- For the *Outage of Element type*, *PowerFactory* automatically isolates the referenced components. The switches connecting the target elements with the other network components are open and the terminals connected to the elements are earthed (if the *Earthed* option in the terminal (*ElmTerm*) dialog is checked). Note that the target element can only be earthed if it is directly connected

(without switches in the cubicle) to terminals, which are then connected through switches to the network terminals.

- For a *Generator Derating*, a reference to the generator which is to be derated and the magnitude of the *MW reductions* is specified. For the *Generator Derating*, the maximum active power that can be dispatched (defined on the *Load Flow* page of the generator element dialog, in the section *Operational Limits*) is recalculated as the difference between the maximum active power (section *Active Power: Ratings*) and the *MW reductions*.

---

**Note:** If a Planned Outage object is defined in the Outages folder of the Operational Library, only the outage types Outage of Element and Generator Derating are enabled. Similarly if outage objects are defined in the Demand transfer folder, only the outage type Demand Transfer is enabled.

---

#### 14.6.7.1 Defining Outages and Deratings

To create a new *Element Outage* or *Generator Derating*:

1. In the Data Manager, open the *Outages* folder.
2. Click on the *New Object* icon , select *Planned Outage* and press **Ok**.
3. The *Planned Outage* dialog will be displayed. In the *Outage Type* frame of the dialog, the options *Outage of an Element* and *Generator Derating* will be enabled. Set the desired *Outage Type*, *Start Time* and *End Time*.
4. The definition of a *Planned Outage* requires reference(s) to relevant network components. To create a reference:
  - Press the **Contents** button of the outage object.
  - In the data browser that is displayed, create a reference to the target element by selecting the *New Object* icon (*IntRef*).
  - Press the  button in the *Reference* field to select the target element.
  - Press **Ok** to add the reference.
5. (*Generator Derating* only) Specify the *MW Reduction* (see previous section for details) for the generator derating.
6. To apply the *Planned Outage*, press the **Apply** button (the **Apply** button is only available if the study time lies within the outage period, and an *Operation Scenario* is active).

Applied outages and generator deratings can be reset using the **Reset** button.

#### 14.6.7.2 Checking Outages and Deratings

The **Check All** button in the *Planned Outage* dialog is used to verify if the actions defined for the target element(s) have been performed (right-click a *Planned Outage* and select *Check* to perform an individual check). Only the outages within a valid period are considered. Outages marked as *Out of Service* are not regarded (even if the study time lies within the outage period).

For an *Outage of Element*, the energising state is always determined by a connectivity analysis. Any component that is connected to an External Grid or a reference Generator is considered to be energised. All other components are considered to be de-energised (if circuit breakers are open). A de-energised component is earthed if a topological connection to a grounding switch or an earthed terminal exists (terminal with the *Earthed* option checked).

**Note:** If the outaged element is a Branch Element (*ElmBranch*), all contained elements are checked. If any of these elements is not correctly outaged, the whole branch is reported as not correctly outaged.

The fulfilment of programmed outages can also be checked via the use of the colour representation function available within the single line graphic by setting the *Colouring* option to *Outage Check* from the colour representation dialog . The following states are coloured, according to user preferences:

- Components that are energised, but should be outaged.
- Components that are de-energised and not earthed, but should be outaged.
- Components that are de-energised and earthed, but should NOT be outaged.
- Components that are de-energised, not earthed and should be outaged.
- Generators that are not derated, but should be outaged.
- Generators that are derated, but should NOT be outaged.

#### 14.6.8 QP-Curves

Q(P)-curves (*IntQpcurve*) are used by generators (*ElmSym*, *ElmGenstat* and *ElmAsm*), loads (*ELmLod*) and station controllers (*ElmStactrl*) when the controller mode Q(P)-Characteristic is selected. In this mode, the reactive power control follows a user-specified characteristic, so that the reactive power setpoint is adapted according to the active power output.

A detailed description of the Q(P)-curves is provided in [Technical References Document](#) (Q(P)-Curve *IntQpcurve*).

#### 14.6.9 QV-Curves

Q(V)-curves (*IntQvcurve*) are used by static generators (*ElmGenstat*), station controllers (*ElmStactrl*) and loads (*ELmLod*) when the controller mode Q(V)-Characteristic is selected. In this mode, the reactive power control follows a user-specified characteristic, so that the reactive power setpoint is adapted according to the voltage.

A detailed description of the Q(V)-curves is provided in [Technical References Document](#) (Q(V)-Curve *IntQvcurve*).

#### 14.6.10 Remedial Action Schemes (RAS)

Remedial Action Schemes (*IntRas*) and Remedial Action Scheme groups may be used during contingency analysis. If the user creates Remedial Action Schemes and groups, they will be stored here in the Operational Library. See Chapter 27, Section 27.11 for more information.

#### 14.6.11 Running Arrangements

*Running Arrangement* objects  store operational data (switch status) for a single substation. As shown in Figure 14.6.2, a *Running Arrangement* uses a reference to the substation object (*ElmSubstat*) whose switch statuses are stored. A *Start Time* and *End Time* is used to specify the validity period of the *Running Arrangement*. Running arrangements are stored in the *Running Arrangements* folder in the Operational Library.

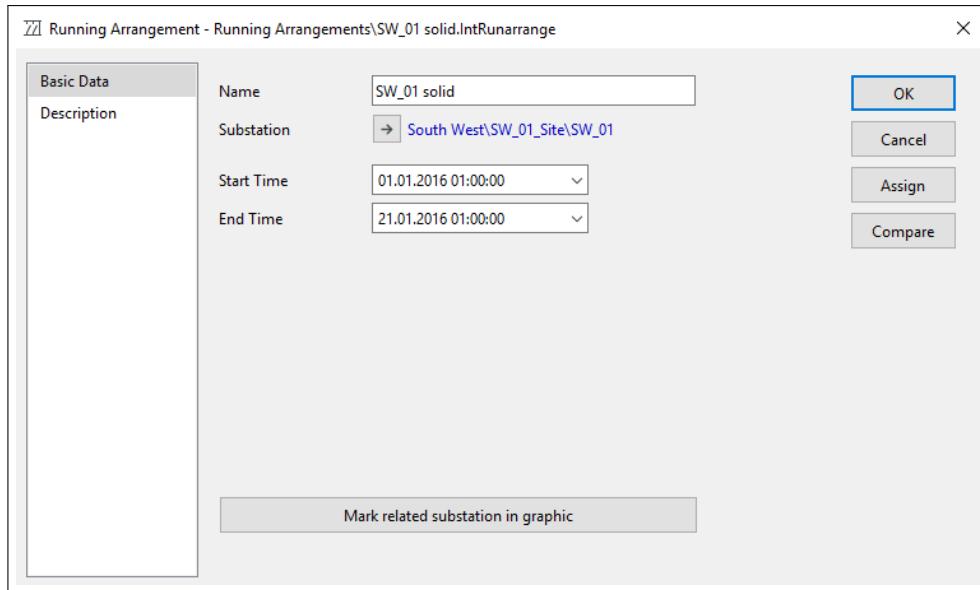


Figure 14.6.2: RA object dialog

Different configurations of the same substation can be defined by storing the corresponding switch statuses in *Running Arrangements*. Different *Running Arrangements* can then be easily selected during a study. If a running arrangement is selected for a substation, the status of the substation switches cannot be modified (i.e. they become read-only). If there is no setting for a switch in a *Running Arrangement* (i.e. the *Running Arrangement* is incomplete), the switch will remain unchanged but its status will also be set to read-only. If the current *Running Arrangement* is deselected, switch status will be reverted to the status prior to application of the *Running Arrangement*, and write-access will be re-enabled. Running arrangements are defined and selected in the substation object dialog *Basic Data* page.

Further details of how to create, select, apply, and assign Running Arrangements are provided in the following sections.

#### 14.6.11.1 Creating a Running Arrangement

To store the current status of the switches in a substation, a *Running Arrangement* object must be created. To create and save a new *Running Arrangement* (RA):

1. Click on an empty place in the substation graphic, and from the context menu choose *Edit Substation*. Open the substation dialog.
2. Click **Save as** to store the switch settings of the substation as a new RA. This button is only available if there is currently no RA selection active.
3. In the new RA dialog is displayed, specify a name and time period, and press **Ok**. The new RA is automatically stored in the *Running Arrangements* folder in the *Operational Library*.

An **Overwrite** button is available in the substation dialog (if no RA is selected), to store current switch statuses to an existing RA.

The **Assign** button contained in the *Running Arrangement* (RA) dialog facilitates the selection of the RA as the one currently selected for the corresponding substation. This action is also available in the context menu in the Data Manager (when right-clicking on an RA inside the Data Manager). It should be noted that assignment is executed immediately and cannot be undone by pressing the cancel button of the dialog.

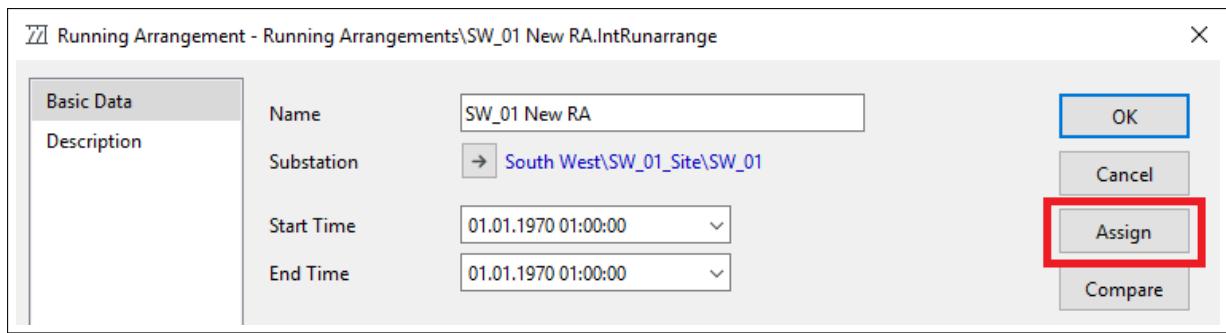


Figure 14.6.3: Running Arrangement Dialog

A **Mark related substation in graphic** button is provided on the Running Arrangement object. This can be used to display the related substation diagram or find the related substation in an overview graphic.

It is also possible to do this using the *Mark in Graphic* option in the context menu displayed when right-clicking on a Running Arrangement in a Data Manager.

#### 14.6.11.2 Selecting a Running Arrangement

A Running Arrangement (RA) can be selected in the Basic Data page of a substation dialog:

1. Open the substation dialog.
2. In the *Running Arrangement* frame of the Substation dialog, select from a list of previously defined RA's.
3. Select the desired RA. This selection is immediately reflected in the substation graphic.

While an RA is selected, the switch statuses of a substation are determined by this RA and cannot be changed by the user (i.e. they are read-only).

If there is no setting for a switch in an RA (i.e. the RA is incomplete), such a switch will remain unchanged but its status is also set to read-only.

Furthermore, there is a button **Select by Study Time** (also available via the context menu when right-clicking on the Data Manager), which selects a valid RA automatically according to the study time. If there are multiple RAs valid for the current study time, or if there is no valid one, a warning is printed to PowerFactory's output window (nothing is selected in this case).

#### 14.6.11.3 Applying and Resetting a Running Arrangement

An active *Running Arrangement* (RA) can be applied to a substation by pressing the **Apply and Reset** button from within the substation dialog. This action copies the statuses stored in the RA directly in the substation switches. It is only available only if an RA is selected. The RA will be deselected afterwards. An RA can be directly set as the substation's selected RA, using the **Assign** button (from within the RA dialog).

The following functional aspects must be regarded when working with running arrangements:

- An RA can be selected for each substation. If an operation scenario is active, the selection of an RA in a substation is recorded in the operation scenario (i.e. the RA selection is part of the operational data included in the operation scenario subset).
- If a variation is active (and there is no active operation scenario), the selection of the RA is stored in the recording expansion stage.

- While an RA is selected, the switch statuses of the corresponding substation are determined by the RA and can not be modified. Any attempt to change such a switch status will be rejected and a warning message will be printed to the output window. The switch statuses preceding the activation of an RA remain unchanged and are restored when deselecting the RA.
- The switch statuses stored in the RA could be incomplete due to the activation of a variation or a modification made to the network model. For example, if an RA was defined and then deactivated, and then later new switches were added to a substation. In this case if the RA is re-activated, a warning would be printed to the output window and the current switch statuses, which depend on the base network, active variations and active operation scenario, remain unchanged. Missing switch statuses will be added only when performing the **Save as** or **Overwrite** functions (available in the substation dialog).
- Switch statuses stored in the RA, and which are currently not required (depending on expansion stages) are ignored and remain unchanged. In this case a summary warning is printed during the RA activation.
- It is not possible to add a new switch to a substation while a running arrangement is selected. Additionally, it is not possible to delete an existing switch from this substation. In both cases the action is blocked and an error message is issued.

For information regarding operation scenarios and their application refer to Chapter 16 (Operation Scenarios).

### 14.6.12 Thermal Ratings

*Thermal Ratings* objects  (*IntThrating*) allow the definition of continuous and post-fault operational ratings for certain branch elements. *Thermal Ratings* objects are stored in the *Thermal Ratings* folder in the *Operational Library*. Depending on their configuration, one-, two- or three-dimensional matrices can be defined for the continuous ratings that take the ambient temperature, wind speed and solar irradiance into account. Similarly, short term post-fault ratings can be defined which additionally take the preload and duration into account. The meteorological dependencies use the settings from the meteorological station (*ElmMeteostat*, see chapter [Meteorological Stations](#)) that is referenced by the respective branch element. If the branch does not reference such a station, the following default values are assumed:

- Ambient temperature: 20 °C
- Wind speed: 0 m/s
- Solar irradiance (GHI): largest defined value in W/m<sup>2</sup>

The following section provide an overview on how to define thermal ratings object. A detailed description is provided in [Technical References Document](#) (Thermal Ratings *IntThrating*).

#### 14.6.12.1 Defining and Using Thermal Ratings

References to *Thermal Ratings* are defined on the *Basic Data* page of the dialog of the target branch elements. Elements that can use references to *Thermal Ratings* are:

- Transmission lines (*ElmLne*)
- Transformers (*ElmTr\**)
- Series reactors (*ElmSind*)
- Series capacitors (*ElmScap*)
- Bays (*ElmBay*)

To create a new *Thermal Ratings* object:

1. Open the folder *Thermal Ratings* from the Operational Library.
2. Click on the *New Object* icon  and select *Thermal Ratings*.
3. The new object dialog is displayed.
4. Depending of the options set on the *Configuration* page, there will be tables on the *Continuous Ratings* and *Short Term Ratings* pages.

The values of a thermal rating object can be edited at any time by double-clicking on it to open the *Thermal Ratings* dialog. Similar to *Circuit Breaker Ratings* and *Capability Curves*, *Thermal Ratings* objects can be made to be time-dependant by means of variations and expansion stages stored inside the *Thermal Ratings* folder (refer to the *Circuit Breaker Ratings* section for an explanation on how to define time-dependant operational objects).

When a contingency analysis (*ComSimoutage*) is configured, the user can define a post-contingency time. According to the pre-fault loading found by the load flow used to calculate the base case, and the post-contingency time (if specified), the ratings to be used in the contingency load flow are determined (based on the referred *Thermal Ratings* object). The loading of the branch elements after the contingency load flow are calculated with respect to the new ratings. For information about contingency analysis refer to Chapter 27 (Contingency Analysis).

#### 14.6.13 V-Control-Curves

The curves in this folder, i.e. V-Control-Curve (*IntVctrlcurve*) are used by transformers, if the user wishes to define the voltage setpoint according to the power or current flow through the transformer.

### 14.7 Reports Library

The *DlgSILENT* Library contains the templates for the predefined reports used in *PowerFactory*. There are two folders inside the Reports library; these are:

**Documents:** contains the output reports (pdf format), grouped according to the calculation function that they relate to. Information about using the standard reports can be found in Section 19.5.

**Tables:** contains the tabular reports, grouped according to the calculation function that they relate to.

The Reports folder in a project is used for storing user-defined output reports templates and tabular reports. Information about creating user-defined output and tabular reports can be found in sections [Output Reports](#) and [Tabular Reports](#).

### 14.8 Scripts Library

In the Scripts Library within a project, all the Python and DPL scripts created by the user are stored.

The *DlgSILENT* Library offers a list of predefined scripts that can be copied or directly used by all users. The scripts in this library are updated when changes that affect them are made in the *PowerFactory* development and in this case, a new version of the script is generated.

The scripts available and maintained in the *DlgSILENT* Library are:

- **Efficiency ratio calculation:** calculates the efficiency ratio (EffR) of a selection of expansion stages, expressing the benefits due to grid extension with respect to the investment costs [in %]
- **EvaluateTececoCalcForStudyCases:** evaluates the techno-economical effects of different study cases compared to one dedicated base study case.

- **HermanBeta:** calculates the voltage drop at single buses or at all buses of a feeder using the Herman Beta algorithm.
- **Impedance Loci:** scans a user-defined impedance loci representing the short-circuit impedance of an external network and determines by means of an iterative Harmonic Load Flow calculation those R-X values resulting in the highest voltage harmonic distortion at each harmonic frequency.
- **RadialFeedersReport:** creates a tabular report of feeder calculation parameters.
- **RunRelForStudyCases:** activates the selected study cases and runs the reliability command stored inside the cases. A result file with the results of the runs is created. Plots and reports are created automatically.
- **Stochastic Switching:** executes a multi-run EMT simulation for line switching analysis. It varies the dispersion and scatter times and reports result extrema and probabilities.

More information about how to use the scripts and is available in the *Description* page of each script (DPL Command).

More information about working with scripts is available in chapter [Scripting](#).

## 14.9 Templates Library

The *Templates* folder is used to store and organise templates of network components (or groups of components) for re-use in a power system model. Components from templates are created using the graphical editor. There are two types of templates:

1. General Templates: used to group a set of network elements, described in detail in Section [14.9.1](#).
2. Templates of composite elements: templates of substations, busbars or composite branches containing all the internal elements (including diagrams) . See sections [14.9.2](#), [14.9.3](#) and [14.9.4](#) for more information.

The *DIGSILENT* Library contains a wide range of pre-defined model templates, including:

- Distributed Energy Resources
- FACTS
- Grid-forming Converters
- HVDC
- Loads
- Photovoltaic
- Plant Controllers
- Steam/Gas/Diesel Power Plants
- Storage Systems
- Variable Speed Drives
- Wind Turbines (including DFIG, Fully Rated, IEC and WECC)

More information about the predefined templates is available in the [DIGSILENT Library Templates Overview](#).

### 14.9.1 General Templates

*General Templates* are used for grouping and packaging objects into a data container that can be stored within the project library or a global library for future use. *General Templates* are intensively used when a fully packaged power equipment model is defined, with one such model containing various functional components, for example:

- Built-in models (e.g. static generators, PWM converters);
- User defined load flow / Quasi-dynamic simulation models (e.g. QDSL models);
- Measurement devices for dynamic simulation (e.g. voltage and current measurement objects);
- User-defined dynamic controller models (e.g. power and voltage controllers, fault-ride through controllers).

Furthermore, all external references of the contained objects can be additionally packed, thus making sure that all required elements are stored in the *General Template*. Consequently, *General Templates* enable *PowerFactory* users to group and pack all of the above objects into a single data container that can later on be easily and safely shared across *PowerFactory* project boundaries.

The dialog of the *General Template* object displays the following information in the Basic Data page:

- *General Template*'s name;
- *Short Description*: a one line text field that may contain a short textual description of the template;
- *Used for*
  - QDS: if this flag is checked, then this template is intended to be used within Quasi-Dynamic Simulations
  - RMS balanced: if this flag is checked, then this template is intended to be used within balanced RMS simulations
  - RMS unbalanced: if this flag is checked, then this template is intended to be used within unbalanced RMS simulations
  - EMT (average): if this flag is checked, then this template is intended to be used within EMT simulations. The model intends to represent an average behaviour of the contained system.
  - EMT (detailed): if this flag is checked, then this template is intended to be used within EMT simulations. The model intends to represent a detailed behaviour of the contained system.

The dialog of the *General Template* object displays the following information in the Description page:

- *Foreign key*: the foreign key of the template;
- *Description*: A text based description field;
- *Documentation File*: A file path entry that can point to a user defined (external) documentation file. If a valid file path is assigned to this field, then the dialog button **Open doc.** is enabled.

The dialog of the *General Template* object displays the following information in the Version page:

- *Version*: A one line text field where the template's version description can be entered;
- *Change log*: A text based change log field is available.

The dialog of the *General Template* object provides the following buttons:

- **OK**: Close dialog and save changes;
- **Cancel**: Close dialog and discard changes;
- **Pack**: Pack external references, if any. If no external references are identified, then this button is disabled. The following options are available when packing:

- External locations: Individually choose which external locations shall be considered when packing:
    - \* DlgSILENT Library
    - \* Other global objects
    - \* Current user
    - \* Other users
  - Options:
    - \* Restrict to Types only
  - Click **OK** to pack the external references according to the configured options.
  - Click **Cancel** to close this dialog and discard any changes.
- **Check:** verify whether any contained objects have references to objects located outside the template.
  - **Open doc.:** Open the external documentation file, as referenced in the *Description* page. If no valid file path is provided, then this button is disabled.
  - **Contents:** display the contents of the template.

In order to create a *General template*, do the following:

- The source data for a *General Template* is an already operational and configured model. Therefore, once a power equipment model / *User-defined* model is configured and operational in a project, go to the single line diagram of the grid in which this model is existing.
- Make sure that the diagram is unlocked (i.e. freeze mode deactivated 
- Select all graphically defined elements shown in the grid's single line diagram which belong to the model.
- Right-click on the selected elements and choose *Define Template*. As a result, a copy of each selected element is created in a newly created *General Template object* (*IntTemplate*) located within the project library (*Library* → *Templates*).
- If the selected elements are part of a dynamic simulation model (i.e. they have a referenced *Composite Model* to parameter *c\_pmod* (if existing), then a copy of the referenced *Composite Model* is added to the template. Other objects with references, such as *Station Controllers* or *Power-Frequency Controllers*, which are not always shown graphically, are also included if the selected elements are referenced in them. Lastly, note that the copied elements maintain their references to other objects within the template (i.e. the references and connections to objects within the template are not lost).

---

**Note:** The template contains by default only the directly copied elements (as previously detailed) but without storing any possibly existing library type objects.

When defining a template, care should be taken to include all the measurement points of the measurement devices. If a measurement point is not included, the measurement location in the corresponding measurement device will be reset. It will then be necessary to select the measurement point manually after the template has been added to the network.

- 
- If the library objects shall be included in the template (recommended action, in order to completely separate the contained model from other source project folders), then open the Edit dialog of the *General Template* and click the **Pack** button. For more information, refer to the already provided information on *Pack* functionality in this section. A subfolder is created within the template containing all referenced library objects.
  - Click on **Check** button to verify whether any contained objects have references to objects located outside the template.

- Click on **Contents** button to display the contents of the template. If needed, further elements/objects can be manually added to the template e.g. DPL scripts, further library objects etc.
- Once the template has been created, it can be used within the existing project or exported outside *PowerFactory* (as a \*.pdf file) or to other projects within the same *PowerFactory* database.

In order to export a *General Template* outside *PowerFactory*, do the following:

- Open the *Data Manager* and deactivate the current project (if any).
- Navigate and locate the *General template* of interest (by default, found in the *Library* → *Templates* folder of a *PowerFactory* project).
- Select the *General Template* and click the **Export Data** button  . Alternatively, from the file menu click *File* → *Export* → *Data (\*.pdf, \*.pfds)* . . .
- Follow the normal export procedure. The template is now saved in a \*.pdf file which can be shared outside *PowerFactory*.
- The \*.pdf file can now be imported in other *PowerFactory* programs. When importing the template, it should always be imported into the *Library* → *Templates* subfolder of the destination project. The General Template can also be imported into a Custom Global Library.

In order to copy a *General Template* from one project to another one located in the same *PowerFactory* database, do the following:

- Open the *Data Manager* and deactivate the current project (if any).
- Locate and copy the *General Template* of interest
- Activate the project in which the template is to be added.
- Navigate to the project's *Library* → *Templates* and paste the template there.

In order to make use of a *General template* (e.g. deploy the contained model into a network), do the following:

- Show the single line diagram of the Grid in which the model shall be deployed. Make sure that the diagram is un-frozen, such that the *Drawing Tools* toolbar is shown.
- Click on the *General templates* 
- A selection window with all available project templates is shown (the “Project Templates” option on the left side pane of the window, corresponds to the *Library* → *Templates* project folder).
- Alternatively, it is possible to choose a template from one of the following folders:
  - *DIGSILENT Library Templates*: pre-defined templates provided with the *PowerFactory* installation
  - *Library Busbar System*: templates for substations
  - *Custom Global Library*: predefined templates stored in the Custom Library. It will only be shown if the library is set as *Used Library* in the User Settings (see chapter User Settings, Section 7.2).
- Click once on the *General Template* of interest in order to select which template shall be deployed.
- Click once in the single line diagram in order to activate the *General Template* drawing mode. If done so, the cursor will change appearance by overlaying the template's graphic. The template is not deployed at this stage.
- The mouse cursor can now be moved to an area in the single line diagram of the grid in order to conveniently deploy the *General Template*.
- When appropriately positioned, click once to deploy the template into the grid at the current location, containing all template's elements (including the non-graphical ones).

### 14.9.2 Substation Templates

Existing substations can be used as “models” to define templates, which may be used later to create new substations. A new substation template is created by right-clicking on one of the busbars of the detailed substation single line diagram and selecting *Define substation template* from the context menu. This action will copy the substation together with all of its contents (including its diagram even if it is not stored within this substation) in the Templates folder.

---

**Note:** In case of creating templates which contain graphical information the default settings of the names and result boxes defining their graphical representation (font, frame, size,...) are copied into the template diagram so that they appear as in the source object(s).

---

For further information about working with substation templates, refer to sections [Creating a substation from predefined templates](#) and [Creating a substation from user defined templates](#).

### 14.9.3 Busbar Templates

Similar to creating substation templates, existing busbars can be used as a “models” to create user-defined templates, which may be used later to create new busbars. A new busbar template is created by right-clicking on the detailed single line diagram or simplified diagram of busbar and selecting *Define substation template* from the context menu. This action will copy the busbar together with all of its contents (including detailed and simplified diagrams) in the Templates folder. If the detailed busbar configuration has been modified, it is possible to right-click the (existing) simplified representation in the main single line diagram and select *Update representation*.

Busbars that have been created by the user in this way can be added to the single line diagram by selecting the *General Busbar System* icon (). Note that for a busbar to be accessible from this icon, both detailed and simplified diagrams must be included within the busbar template, as in the previously described method.

### 14.9.4 Composite Branch Templates

Composite Branch templates can be defined as follows:

1. To create a Branch template, navigate to the *Library* → *Templates* folder in the Data Manager.
2. Right-click on the right pane of the Data Manager and select *New* → *Branch...* from the context menu.
3. In the branch edit dialog, define the name of the branch and press **Ok**.
4. A new (empty) single line diagram will be displayed. Draw the required elements (for example, a terminal with two lines connected, with each line connected at one end only).
5. To create an instance of the Branch from the newly created Branch template, navigate back to the main grid diagram, then select the Composite Branch () icon and connect the branch to existing terminals on the Single Line Diagram.

Alternatively, composite branches can be defined in the Data Manager as described in Chapter 12 ([Building Networks](#)), Section 12.5.4 ([Defining Composite Branches in the Data Manager](#)).

## 14.10 Dynamic Models Library

The *Dynamic Models* folder in the *DIGSILENT* Library contains a large number of predefined dynamic model types, which are available to all users and can either be directly used from this library or copied to the project library and modified. Models of the following categories are included:

- **DSL Macros:** contains a selection of DSL macro types used to build DSL models. The complete list of DSL macros in *PowerFactory* is available in the [DSL Macros Documentation](#), which can also be accessed via the Help menu: *Help* → *Technical References* → *DSL Macros*.
- **FACTS:** contains DSL model types and composite model frames for several flexible alternating current transmission systems (FACTS).
- **HVDC:** contains composite model frames and DSL model types for HVDC models (high voltage direct current).
- **Inverter Based Resources/Storage:** contains DSL model types and composite model frames with regard to inverter based resources such as wind and photovoltaic generation and storage systems.
- **Modelica:** contains a selection of Modelica model types, which can be used as building blocks to create higher level Modelica models.
- **Motor Loads:** contains composite model frames, motor driven machines (objects class *Elm-Mdm\**) and DSL model types regarding motor loads.
- **Synchronous Generator Power Plants:** contains several DSL model types including frames, excitation systems, governors, power system stabilisers and limiters etc. for synchronous generator power plants.
- **Various:** contains further DSL model types, e.g. of PLL elements (phase-locked loop) or generating unit protection.

Dynamic models from the following sources are available:

- **DIGSILENT:** These models were developed by *DIGSILENT* and do not necessarily comply with any standard.
- **CGMES:** DSL model types according to CGMES (Common Grid Model Exchange Standard) Profile 2.4.15 and Profile 3.1, defined by ENTSO-E (European Network of Transmission System Operators for Electricity). CGMES Profile 3.1 models are at the same time compatible with IEC 61970 – Part 457: Common Information Model (CIM) for Dynamics Profile, Edition 2.0. The complete list of models and their description is available in the [Synchronous Generator Power Plants Reference](#).
- **IEC:** DSL model types that comply with IEC standards, in particular for wind turbine modelling according to the [31] standard.
- **IEEE:** DSL model types that comply with IEEE standards or technical reports. More information about which standard is used and the model itself is available in the *Description* page of each DSL model type.
- **PSS/E compatible:** DSL model types that were developed to be compatible with the ones available in the software PSS/E. More information about the PSS/E Interface is available in chapter Interfaces, section [24.4](#).
- **WECC:** DSL model types according to WECC documentation (Western Electricity Coordination Council).

Within the project, dynamic models are stored inside the project library *Dynamic Models*. More information about how to develop and work with dynamic models is available in chapter [Models for Dynamic Simulations](#).

## 14.11 Quasi-Dynamic Models Library

The *Quasi-Dynamic Models* folder in the *DIGSILENT* Library contains predefined quasi-dynamic models types that can be used during a quasi-dynamic simulation. The list includes:

- Battery Type 1 (Voltage Measurement)
- Battery Type 2 (Power Measurement)
- Battery Type 3 (Current Measurement)
- Electric Vehicle (Load)

Within the project, quasi-dynamic models are stored inside the project library *Dynamic Models*. More information about quasi-dynamic simulation and how to develop and modify quasi-dynamic models is available in chapter [Quasi-Dynamic Simulation](#), sections [28.3](#) and [28.5](#).

## 14.12 Protection Devices Library

The *Protection Devices* folder in the *DIGSILENT* Library is a relay model library which contains models of both generic and manufacturer-specific relays.

For more information on these blocks please refer to the [Protection Devices Library](#), where all the protection devices are listed and additional detailed documentation is provided.

## 14.13 Harmonics Library

The Harmonics library within the *DIGSILENT* Library contains data that can be used during harmonic analysis, the available folders are:

- **Frequency Characteristics:** includes several frequency polynomial characteristic which represent inductances and resistances dependent on the frequency. Information about how to use these characteristic is available in chapter [Power Quality and Harmonics Analysis](#), Section [36.4.3](#).
- **Harmonic Current Spectrums:** includes types used to define current spectrum of harmonics. Mode information about the definition of harmonics injections is available in chapter [Power Quality and Harmonics Analysis](#), Section [36.4.1](#).

# Chapter 15

## Grouping Objects

### 15.1 Introduction

This chapter describes the management and functionality of the objects used to group network components.

Among the various applications of grouping objects is their use for visualisation in network diagrams. For this reason, many of these grouping object classes include the possibility of associating colours with the objects. For general information about configuring colours, see Section [4.7.8](#).

### 15.2 Areas

To facilitate the visualisation and analysis of a power system, elements may be allocated into areas (*ElmA* ). The single line diagrams can then be coloured according to these areas and special reports after load flow calculations ('Area summary report' and 'Area interchange report') can be generated. Area objects are stored inside the *Areas* folder () within the *Network Data*.

New areas can be defined directly from the single line diagram, using the drawing tools, as follows:

- Click on the *Area* icon from the drawing tools ()
- With the left mouse button held down, circle the elements that should belong to the area. Make sure that at least one terminal is within the selection.
- A graphical representation of the area will be automatically added to the diagram as shown in Figure [15.2.1](#).

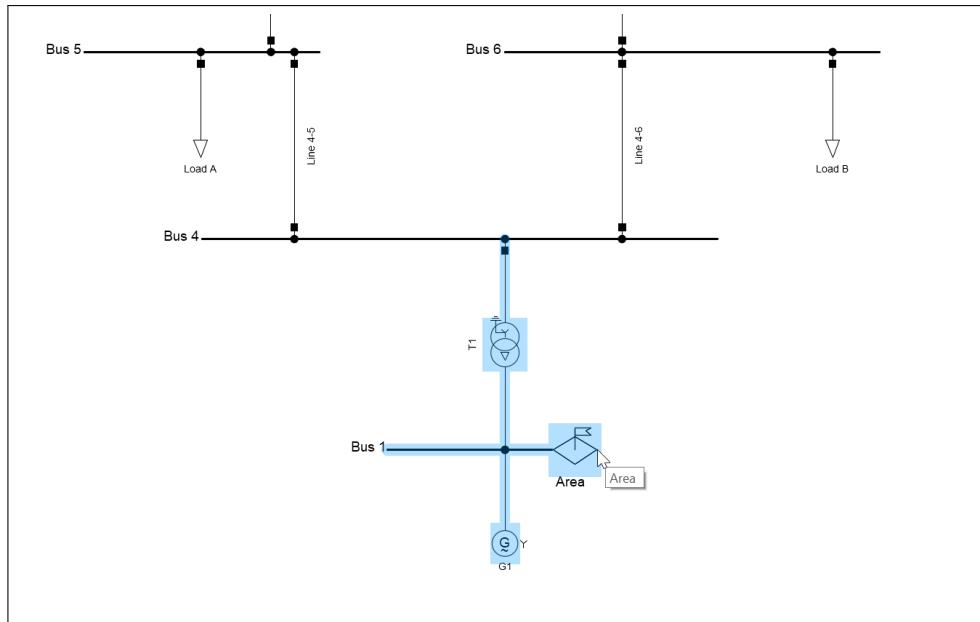


Figure 15.2.1: Graphical representation of an area)

To add elements to an existing area:

- Multi-select the components belonging to the new area (in the Data Manager or in a single line diagram).
- Right-click on the selection and select *Network Groupings* → *Area* → *Add to...* from the context menu.

**Note:** The context menu of the elements can also be used to define an area: *Network Groupings* → *Area* → *New...*

New areas can also be defined in the Data Manager; however, using this option will not automatically create a graphical representation of the area. Graphical representations of existing areas can be inserted in a single line diagram using the Diagram Layout Tool.

In the edit dialog of the new area you can select a colour to represent the area in the single line diagrams. Using the **Edit Elements** button you can have access to all the element belonging to that area in a data browser, then you can edit them. The **Mark in Graphic** button may be used to locate the components of an area in one or more single line diagrams.

### 15.3 Virtual Power Plants

Virtual power plants are used to group generators in the network, in such a way that the total dispatched active power is set to a target value. The dispatch of each generator (the *Active Power* field available in the *Dispatch* section of the *Load Flow* page in the generator element dialog) is scaled according to the virtual power plant rules (must run, merit of order, etc.), in order to achieve the total target value.

Virtual power plant objects (*ElmBmu* ) are stored inside the *Virtual Power Plants* folder ( ) within the *Network Data* directory.

### 15.3.1 Defining and Editing a New Virtual Power Plant

A new virtual power plant can be defined using the drawing tools, as follows:

- Click on the *Virtual Power Plant* icon from the drawing tools (G)
- With the left mouse button held down, circle the generators that should belong to the virtual power plant

Alternatively a virtual power plant can be defined by multi-selecting the generators in a single line diagram and selecting *Network Groupings* → *Virtual Power Plant* → *New...* from the context menu.

In both cases, a graphical representation of the virtual power plant will be automatically added to the diagram.

The rules that determine the dispatch of the selected generators are set in the virtual power plant dialog. The total active power to be dispatched is set in the field *Active Power Setpoint*. The dispatch of the belonging generators (variable `e:pgini` from the Load Flow page of the generator) is set by pressing the **Apply** button. If the 'Maximal active power sum' of the included generators (sum of the maximal active power operational limit of the generators) is smaller than the active power to be dispatched, an error message will be displayed. Otherwise the dispatch is set according to the defined *Distribution Mode*:

**According to merit order** Distribution of the dispatched active power is done according to the priorities given to each generator in the Merit Order column of the 'Machines' table (this value can also be set in the *Automatic Dispatch* tab of the *Load Flow* page of the generators dialog). Lower values have higher priority. Generators with the option 'Must Run' checked are dispatched even if they have low priority (high value). It is assumed that the merit of order of all generators in the virtual power plant is different. If not an error message appears after the **Apply** button is pressed.

**According to script** The rules for the dispatch are set in user defined DPL scripts, which are stored inside virtual power plant object. To create new scripts or to edit the existing ones you must open a data browser with the **Scripts** button.

### 15.3.2 Applying a Virtual Power Plant

Check that the active power set for the virtual power plant is less than or equal to the maximum power. Press the **Apply** button.

### 15.3.3 Inserting a Generator into a Virtual Power Plant

Generators are added to an existing virtual power plant in their edit dialog, by adding a reference on the *Automatic Dispatch* tab of the *Load Flow* page. Note that a generator can only belong to one virtual power plant. Merit order and must run properties can be defined as required.

Generators can also be added to a virtual power plant by right-clicking on the element (in the single line diagram or in the Data Manager) and choosing *Network Groupings* → *Virtual Power Plant* → *Add to...* from the context menu.

## 15.4 Boundaries

Boundaries are used in the definition of network reductions and to report the interchange of active and reactive power after a load flow calculation. Boundary objects (*ElmBoundary* ) may define topological

regions by specifying a topological cut through the network.

Boundaries are defined by the cubicles that determine the cut through the network, these cubicles together with the orientations define the corresponding “Interior Region”. Topologically, the interior region is found searching through the network starting at each selected cubicles towards the given direction. The topological search continues until either an open switch or a cubicle that is part of the boundary list is found. Any open switch that is found by this search is considered to be part of the interior region.

Boundaries can be defined using the *Boundary Definition Tool*, directly on the branch elements by right-clicking on them and selecting *Network Groupings* → *Boundary* → *New...* or using the *Boundary* icon from the drawing tools (█).

Boundaries can be adapted by modifying the *Boundary Definition* table within the boundary object or by adding additional cubicles by right-clicking on them and selecting *Network Groupings* → *Boundary* → *Add to...*.

The boundaries are stored in the project folder *Boundaries* (█) within the *Network Data*.

### 15.4.1 Boundary Definition Tool

The Boundary Definition Tool (█) is located within the *Additional Tools* toolbox as shown in Figure 15.4.1.

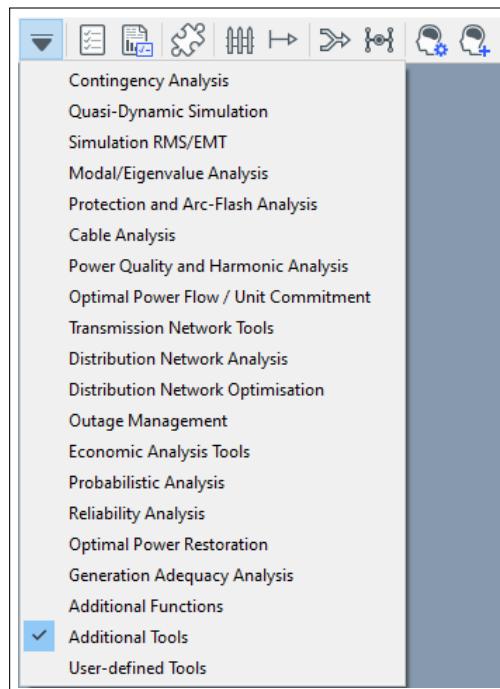


Figure 15.4.1: Additional Tools toolbox

The following options are available when using the Boundary Definition Tool command:

#### 15.4.1.1 Basic data

##### Based on regional elements

Zones, areas, grids and even existing boundaries can be used to define a boundary. The selection supports multiple elements of the same type.

When regional elements are used, some additional options are available for the user:

- One common boundary: single boundary containing all the interior elements of the composing regions.
- Separate boundary for each region: a number of boundaries corresponding to the number of regions will be defined with corresponding interior elements.
- All boundaries between neighbouring regions: each combination between selected regions is considered and corresponding boundary is defined.

### Based on branch elements

Branch elements (e.g. lines, transformers) can be used to define a boundary, *PowerFactory* will perform a topological search to define the interior elements. To finishing defining the interior region, the user can check the *Assign selected branch elements to interior region* checkbox on the *Basic Data* page of the command dialog.

In addition the *Boundary Definition Tool* offers the possibility to define the boundary only if it splits the network into two separated regions.

#### 15.4.1.2 Advanced

The options available on this page depend on whether the boundary is based on regional elements or branch elements.

##### Use fictitious border network

This option is available when the boundary is being created based on regional elements. It is intended specifically for networks that have a “fictitious border grid”, which contains the nodes marking the borders between adjacent grids. This is a typical configuration in European networks to facilitate CIM (CGMES) data interchange. The option is relevant when one common boundary is to be created. If this option is checked, the fictitious border grid must also be selected. Then, those fictitious border grid elements which connect adjacent regions that have been selected to define the common boundary will also be included.

##### Choice of the border inside a branch (\*.ElmBranch)

The option *Prefer the longest line modelled as distributed parameters* is available when the boundary is being created based on branch elements, and is relevant when the user is defining a boundary for the purpose of carrying out a co-simulation calculation (see Section 29.11). It gives the user control over the positioning of the boundary border within an *ElmBranch* so as to ensure that the co-simulation regions are connected via lines of sufficient length.

## 15.4.2 Element Boundary

The element boundary *ElmBoundary* edit dialog is accessible by double clicking on the boundary element, using either the Data Manager or the Network Model Manager.

To add cubicles to an existing Boundary:

- In the Boundary dialog, right-click on the table (on the number of a row) that lists the included cubicles.
- Select *Insert rows*, *Append rows* or *Append n rows* from the context menu.

- Double click on the *Boundary Points* cell of the new line.
- Select the target cubicle using the data browser that pops up.

After selecting the desired cubicle, the terminal and the branch element connected to it are added to the *Busbar* and *Branch* cells on the table. By default the *Orientation* (direction used to determine the interior region) is set to the branch; you can change it in order to direct the definition of the internal region to the connected terminal.

Cubicles can be retired from a boundary by selecting *Delete rows* from the context menu of the table in the element dialog.

The selected colour underneath the boundary name is used when representing the boundary in single line diagrams (●). Each element in the graphic is coloured according to the following criteria:

- If it uniquely belongs to one interior region of a boundary to be drawn, its colour will be assigned to that specific boundary colour.
- If it belongs to exactly two of the interior regions of the boundaries to be drawn, its will be represented with dashed lines in the specific boundary colours.
- If it belongs to exactly more than two of the interior regions of the boundaries to be drawn, its will be represented with dashed lines in black and the colour selected for multiple intersections.

#### 15.4.2.1 Boundary object buttons

The **Edit Interior Elements** button can be used to list in a data browser all the components included in the internal region. The **Mark Interior Region** button marks all the components of the interior region in the selected network diagram.

Topological changes in the network that affect the defined interior regions are automatically detected by the program.

The **Check Split** button can be used to check whether or not the boundary is a closed boundary which splits the network into two parts.

Related to the Check Split is an option *Topological search: stop at open breakers*. Some boundaries may only split the network because particular breakers are open, i.e. they effectively rely on these breakers to ensure that they are “splitting” boundaries. By selecting the *Topological search: stop at open breakers* option, this ensures that they are taken into account. In some cases, a boundary may be “splitting” only if the *Topological search: stop at open breakers* option is selected; in such a case the user can find out which switches are critical by using the **Report open switches making boundary split** button to get a list of these switches.

The **Colour graphic according to this boundary** will set the colouring option of the currently active graphic according to the Boundary Definition of the boundary in question. This is to help users visualise large boundaries in particular, as they create or modify them. (Note that if the original colouring scheme needs to be restored subsequently, this will have to be done manually.)

## 15.5 Circuits

Circuits are objects of class *ElmCircuit* (○), and are used to group branches in order to clarify which branches are connected galvanically. Each branch (*ElmBranch*) can have a reference to any defined circuit object. This feature allows branches to be sorted according to the circuit to which they belong.

To create a new Circuit:

- In the Data Manager open the Circuits folder from the Network Model.
- Click on the *New Object* icon.
- The edit dialog of the new Circuit pops up. Give a name to the new object and press **Ok**.

Branches are added to a circuit using the pointer from the 'Circuit' field of the branch dialog. The button **Branches** in the Circuit dialog opens a data browser listing the branches that refer to that circuit.

---

**Note:** Circuits that are created or deleted when a recording expansion stage is active; become available/not available only if the corresponding variation is active and the expansion stage activation time is earlier than the current study time.

---

## 15.6 Feeders

When analysing a system it is often useful to know where the various elements are receiving their power supply from. In *PowerFactory* this is achieved using Feeder (*ElmFeeder* ).

A feeder is defined at a line or transformer end, and then the feeder definition algorithm searches the system from the definition point to determine the extent of the feeder. The feeder ends when:

- An open breaker is encountered; or
- The end of a line of supply is encountered; or
- 'Terminate feeder at this point' is enabled in a cubicle (optional); or
- A higher voltage is encountered (optional).

Once a feeder has been defined it may be used to scale the loads connected along it according to a measured current or power, to create voltage profile plots or to select particular branches and connected objects in the network. Following load flow calculations, special reports can be created for the defined feeders. To distinguish the different feeder definitions, they can be coloured uniquely in the single line graphic. All feeder objects are stored in the Feeders folder () within the *Network Data* folder.

A new feeder is created by right-clicking on a cubicle (that is, with the cursor close to the breaker in the single line diagram) and selecting *Network Groupings* → *Feeder* → *New...*

Alternatively, the *Feeder* icon from the drawing tools () can be used to define a feeder at the end of a line/transformer. In both cases, the graphical representation of the feeder will be added to the diagram.

The feeder dialog presents the following fields:

### Name

**Cubicle** Is a reference to the cubicle where the Feeder was created. It is automatically set by the program once the Feeder is created.

---

**Note:** The reference cubicle must be set manually if the feeder is created from the Data Manager.

---

**Zone** Reference to the Zone (if any) to which the feeder belongs. A Feeder is assigned to the zone of the local busbar/terminal.

**Colour** Sets the colour be used when the Feeder Definitions colouring mode () is engaged in the single line diagram.

**Terminate feeder when...** A feeder will, by default, terminate when a higher voltage level is encountered, however, this may not always be desirable. This may be prevented by un-checking this option. The feeder will now continue 'past' a higher voltage level and may be terminated at a user defined cubicle if desired. To manually terminate a feeder right-click a branch element above the breaker (to select the desired cubicle where the feeder is going to end) and select *Edit Cubicle*. The cubicle dialog will be presented, and the 'Terminate feeder at this point' option may be checked.

**Orientation** The user may select the direction towards the feeder is defined. 'Branch' means that the feeder starts at the cubicle and continues in the direction of the connected branch element. 'Busbar' means that the Feeder is defined in the direction of the connected Terminal.

**Load Scaling** In any system some loads values may be accurately known whilst others are estimated. It is likely that measurement points exist for feeders in the system as well, and thus the power that is drawn through this feeder is also known. The load scaling tool assists the user in adjusting these estimated load values by scaling them to match a known feeder power or current that has been measured in the real system. More information about the use of the Load Scaling Function is given below.

**Elements** The **Mark in Graphic** button may be used to show all the elements of a Feeder in one or more single line diagrams. The **Edit** button is used to list all the elements belonging to a Feeder in a data browser.

To use the Load Scaling tool first define which loads may be scaled by enabling the 'Adjusted by Load Scaling' option on the Load-Flow tab of the load dialog. All of the loads in a feeder may also be quickly viewed by editing the feeder from the feeders folder.

Load scaling is now performed by the load flow calculation function when:

- At least one feeder is defined with load scaling according to a current or power.
- The option 'Feeder Load Scaling' is enabled in the load-flow command dialog (basic options).
- At least one load exists in the feeder area for which
  - A change in operating point affects the load-flow at the feeder position
  - The option 'Adjusted by Load Scaling' has been enabled.

The load-flow calculation will then adjust the scaling of all adjustable loads in the feeder areas in such a way that the load-flow at the feeder equals the current or power set-point.

The feeder setpoint is influenced by the zone scaling. This means that the current or power flow as calculated by the load-flow could differ from the setpoint in the feeder dialog when the busbar where the feeder is defined is part of a zone.

For instance, a feeder has a set-point of 1.22 MVA. The busbar is in a zone and the zone-scale is set to 0.50. The flow at the feeder position will thus be 0.61 MVA.

For information about colouring the single line graphic according to feeder definitions refer to Chapter 10: Network Graphics. For information about voltage profile plots, refer to Chapter 19 (Reporting and Visualising Results).

### Defining Feeders from a Terminal Element

Often it is useful to be able to quickly setup a feeder or many feeders from a 'source' bus within the system. There is a specific methodology within *PowerFactory* for this purpose. The procedure is as follows:

1. Right-click the target terminal where the feeder/s should be defined from.
2. Choose the option *Network Groupings* → *Feeder* → *New...* from the context menu that appears. This step is illustrated in Figure 15.6.1.

3. PowerFactory will automatically create Feeder objects for each of the connected two terminal elements, for example lines and transformers. The list of created feeders is displayed in a pop-up window. The default name for each feeder is the concatenation of the terminal name and the connected object.

4. Adjust the feeder colours and definitions as required and remove any unwanted feeders.

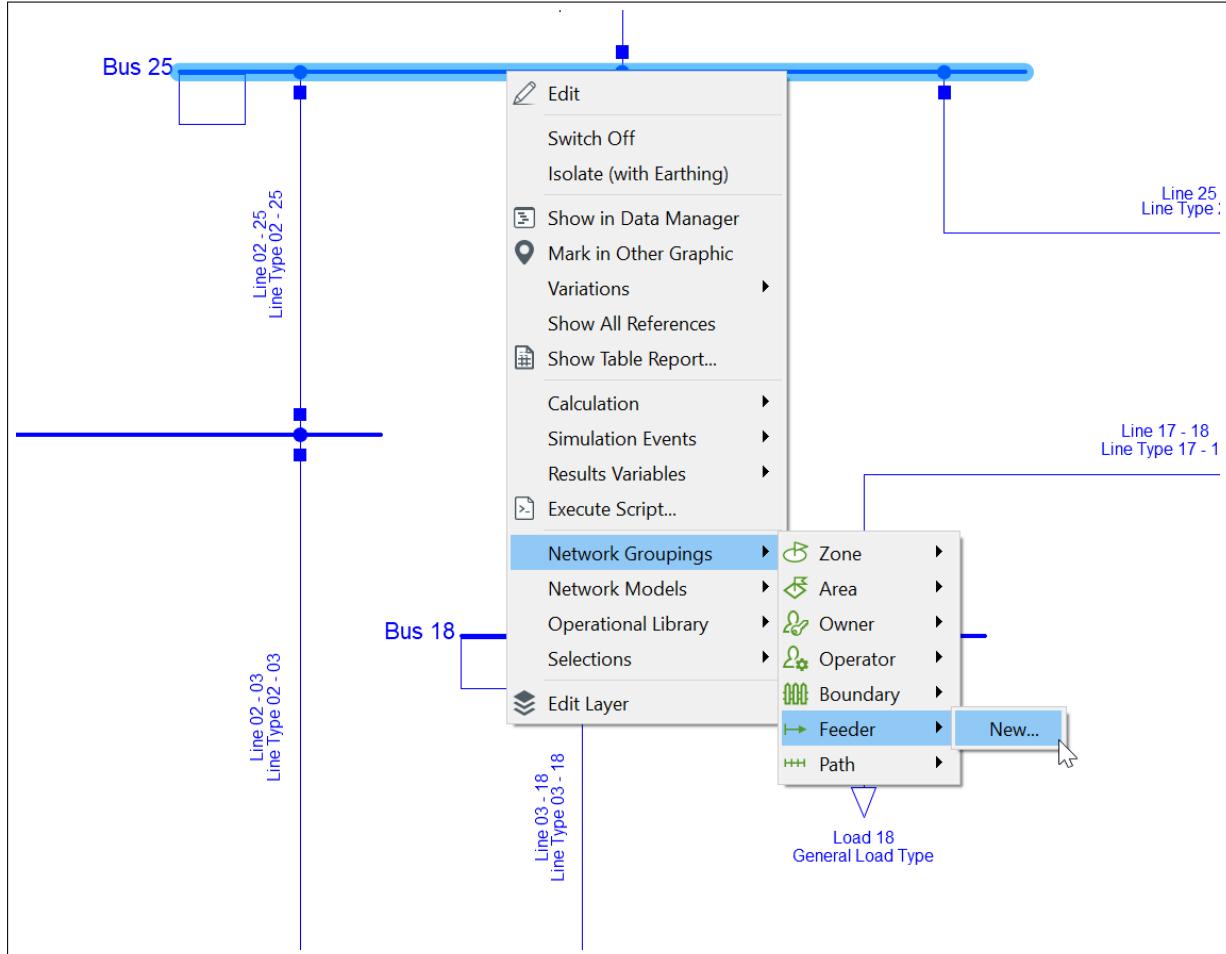


Figure 15.6.1: Definition of Feeders from a terminal by right-clicking the terminal

**Note:** The Load Scaling options are part of the operation scenario subsets; therefore they are stored in the active operation scenario (if available). The Load Scaling options are stored in the active expansion stage (if available) if no active operation scenario is active. Feeders that are created or deleted when a recording expansion stage is active; become available/not available only if the corresponding Variation is active and the expansion stage activation time is earlier than the current study time.

### 15.6.1 Feeder Tools

Feeder Tools is a set of three tools that can be used only in radial systems to change voltage, technology from a particular point downwards.

**Note:** Additional functions for feeders like *Backbone Calculation* or *Phase Balance Optimisation* are available in the module *Distribution Network Tools*, described in Chapter 42

### 15.6.1.1 Voltage Change Tool

The Voltage Change Tool automatically changes type data (for transformers, lines, loads and motors) and element data such that the primary voltage can be changed to a specified voltage value. The tool will change the voltage from a particular point downwards but is limited to the HV side. This will enable the voltage level of a network to be changed for planning studies, taking into account all downstream equipment.

### 15.6.1.2 Technology Change Tool

The Technology Change Tool automatically changes type data (for transformers, lines, loads, motors) and element data such that the primary number of phases or neutrals (commonly referred to as 'technology') can be changed to a specific number of phases/neutrals. The tool will change the technology from a particular point downwards but is limited to the HV side.

**Note:** If a device such as a transformer or shunt device is no longer compatible (number of phases and/or phasing is not supported) then the device is set out of service and is reported to the user.

### 15.6.1.3 Feeder Tool Command

Feeder Tools is a built-in command (*ComFeedertool*) in *PowerFactory* and can be started via the right-mouse context menu, by clicking on an element of a feeder as shown in Figure 15.6.2. A radial feeder must be defined prior to using the command.

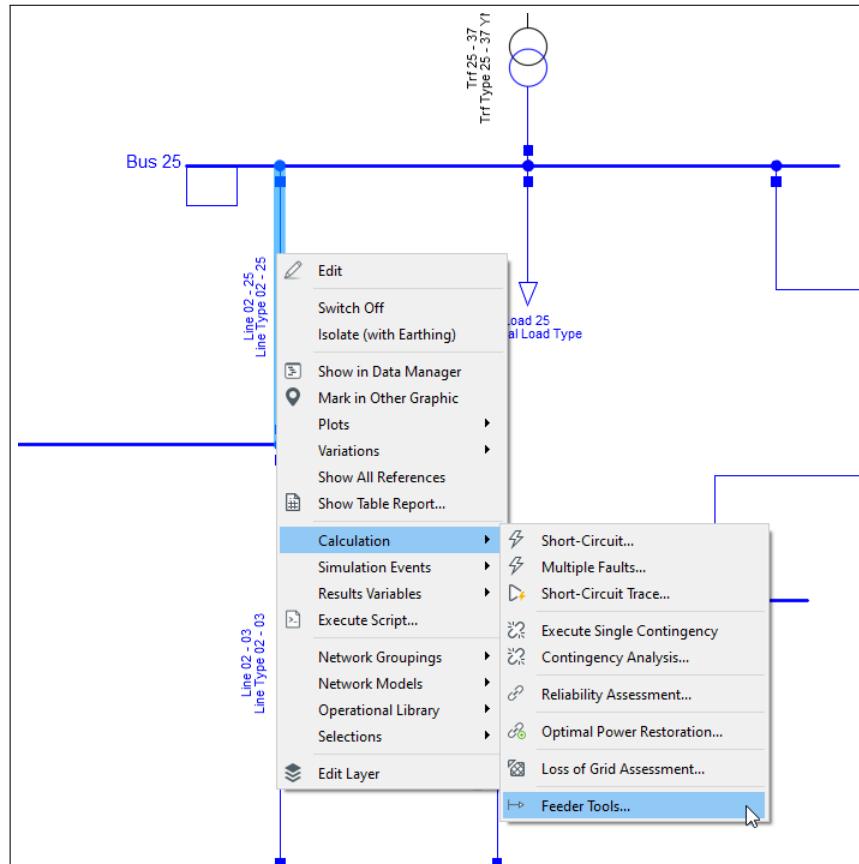


Figure 15.6.2: Feeder Tool

The voltage, technology and balancing tools are all related and are integrated in *PowerFactory* as one command having different options for enabling/disabling each individual tool. Any combination of the three tools can be used. For example, a user may want to evaluate the alternative where an existing 19 kV SWER line is to be changed to a 22 kV three-phase line. In this case, the line type voltage, phasing and technology will all need to change. The transformers should then be changed to equivalent single- or dual-phase transformers (depending on their original secondary technology) with 22 kV phase-to-phase connected primary windings.

Since Voltage and Technology Tools are more intrinsically related than the Auto Balancing Tool, the first tools are meshed into one algorithm. The Auto Balancing Tool runs independently of Voltage and Technology Tools but requires a convergent load flow. If the user wishes to apply all tools in one run (Voltage, Technology and Balancing), then the algorithm of Voltage and Technology Tools is performed followed by execution of the Auto Balancing Tool.

#### 15.6.1.4 How to use the Voltage and Technology Tool

When selecting the Voltage Change Tool, the user should specify the voltage level in kV (*Previous Voltage*) that will be replaced, and the *New Voltage*. Both voltages should be specified as phase-phase voltages, even if there is no phase-phase voltage available; for example when the previous or new technology is 1 PH or 1 PH-N. When selecting the Technology Change Tool, the user should specify the *New Technology* from the drop-down list and then proceed as follows:

1. A radial feeder must be defined.
2. A *Start Element* (terminal or line) must be selected:
  - If the *Start Element* is a terminal, then this is defined as the *Start Terminal*.
  - If the *Start Element* is a line, then the *Start Terminal* is defined as:
    - For the Voltage Tool: the line terminal nearest to the feeder definition point
    - For the Technology Tool: the line terminal more distant from the feeder definition point.

**Note:** The algorithm uses a top-down approach: working from the *Start Terminal* downwards to the *Stop Point*

3. Definition of *Stop Point* for Voltage/Technology Tools: The voltage/technology changes will stop at transformers or open points. For transformers, only the primary voltage/technology is changed. This means that the transformer secondary voltage/technology and secondary network remains unchanged. If the transformer technology (three phase, single phase or SWER) is not compatible with the new primary technology, then the transformer will be disconnected. This will occur when a three-phase primary network supplies a three-phase transformer and the primary technology is changed to a non-three-phase technology. In this case, the transformer will be disconnected. Likewise, three-phase machines cannot be connected to a non-three-phase technology. (Note: Out-of-service elements are not Stop Points for Voltage/Technology Tools.)
4. Setting the new type/element voltage: If *Voltage Change Tool* is selected, the new voltage is equal to the *New Voltage* specified by the user. A voltage change can be performed independent of a technology change. If *Technology Change* is disabled, the voltage change will be associated with the existing technology.
5. Setting the new type/element technology: If *Technology Change Tool* is selected, the new technology is that of the *New Technology* specified by the user. A technology change can be performed independent of the voltage (the voltage will not be changed).
6. A *Linking Object* must be provided.  
The selection of a new Type is not automated as there may be several types that could be compatible with a particular scenario. The solution for this is a linking database object. This linking object stores the relationships between old and new types for different new voltage and/or technology changes. This linking object can be saved in a project or library. It should be added to and modified each time a technology/voltage change is performed.

If for any network element a new type that should match a specific new voltage/technology is not found, the user can choose how the program should proceed by selecting one of the following Linking Object options:

- Prompt for new type selection: the user should manually select which type should be used. If the selected type is still not valid (see item 7: Validation rules for types), the program will present new options: (i) the user can select a new type again, (ii) ignore replacing this type, or (iii) interrupt algorithm execution. Otherwise, if the selected type is valid (see item 7: Validation rules for types), the existing record in Linking Object is updated or in the event that it does not exist, a new one will be created.
- Automatically creates new type: a new type that matches the required voltage/technology is automatically created and the existing record in the Linking Object is updated, or in the event that it does not exist, a new one will be created.
- Do not change the old Type: the old type is not replaced and the corresponding element is put out-of-service. Changes, if necessary, should be manually performed after the command execution.

An example of a *Linking Object* is shown in Figure 15.6.3. The voltage tolerance (parameter *vtol*) for comparison between type voltage and new voltage can be optionally specified. The default value is 30 %. Records in *Linking Object* should be unique for each combination of Old Type, New Voltage and New Technology. Validation rules (see item 7) are applied when the user presses the **OK** button or/and automatically (i.e. within the algorithm).

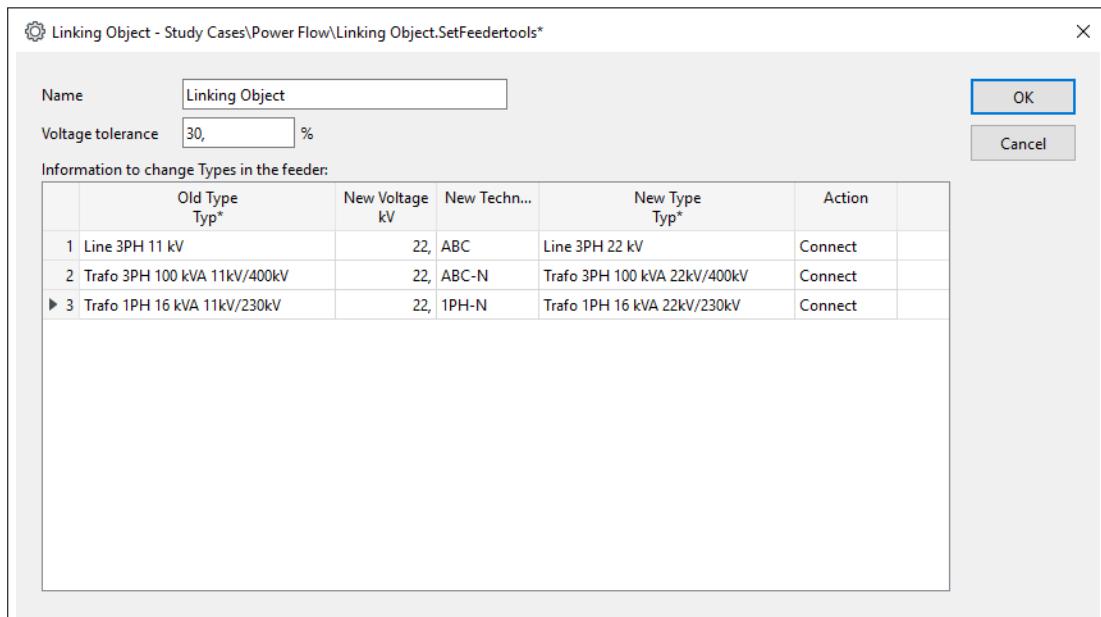


Figure 15.6.3: Linking Object

#### 7. Validation rules for types:

- The new type voltage must be equal to the new voltage (item 4) or inside its tolerance specified by parameter *vtol* in the Linking Object (item 6). If this rule fails, the message “The Type selected voltage is incompatible with the new voltage” is shown. This rule does not apply to the load type (*TypLod*).
- For the motor type (*TypAsmo*, *TypAsm*), the number of phases must be equal to or greater than the old number of phases. If this condition is not met the message “Cannot supply a three-phase motor from a single or bi/dual phase” is shown.
- For the 2-winding transformer type (*TypTr2*), the number of phases must be equal to or greater than the old number of phases. If this condition is not met the message “Cannot supply a three-phase secondary network from a single or bi/dual phase primary” is shown.
- As stated in item 3, the voltage/technology changes will stop at the primary of a transformer so that the secondary network remains unchanged. Despite this, the algorithm does not limit the selected 2-winding transformer type to the same secondary voltage/technology.

- e. For the line type (*TypLne*) the number of phases and neutrals must be equal to the one required by the new technology specified by the user. For the 2-winding transformer type (*TypTr2*), motor type (*TypAsmo*, *TypAsm*) or load type (*TypLod*), the number of phases and neutrals must be less than or equal to the one required by the new technology. If this condition is not met the message “The type selected technology is incompatible with the new technology” is shown.
- f. The algorithm will change a type object when:
  - The actual type voltage/technology is not compatible with the new voltage/technology;
  - The actual type voltage/technology is compatible with the new voltage/technology but the change operation is specified in the Linking Object.If the type is incompatible with the new voltage/technology and a new type has not yet been specified in the *Linking Object* for a particular old type, *New Voltage* and *New Technology* combination, then one of the following predefined actions will occur:
  - Prompt for new type selection: during this process the user can select or create a new type, where the default parameters are the same as those for the old type. Once a new type has been selected or created, it will be checked to ensure that it is compatible with the new voltage/technology. If it is still incompatible, the user will be asked whether to select again or to put this element out-of-service. The changes/additions are stored in the Linking Object so that they are available for future reuse.
  - Automatically creates a new type: a compatible type is created where the default parameters are the same as those for the old type. The changes/additions are stored in the Linking Object so that they are available for future reuse.
  - Do not change the type: no new type is selected and the element is set out-of-service.

## 15.7 Meteorological Stations

*Meteorological Stations* (*ElmMeteostat*)  hold information about the meteorological conditions of ambient temperature, wind speed and global horizontal irradiance for a certain location. This information can be used by other objects. An example for this is the usage by a thermal rating object for the determination of continuous and short-term ratings. Thus, meteorological stations can be linked to certain network elements such as lines and generators, but also to sites and substations to be valid for all the contained elements within these objects. For objects for which meteorological stations can be individually selected, those selections will take precedence over that of the parent object. The meteorological stations are stored in the folder *Network Data*.

Meteorological stations can be defined either via the element that is to be part of the meteorological station (from any of the generator elements described in Section 49.4), via the single line diagram by right-clicking on an appropriate element and selecting *Network Groupings* → *Meteorological Station* → *New...* from the context menu or by using the *Meteorological Station* icon from the drawing tools ().

---

**Note:** The ability to define a *Meteorological Station* for generators is dependent upon whether at least one of the “member” generators has the options *Generator* and *Wind Generator* selected on its *Basic Data* page. If these options are not selected, the context menu entry is not visible.

---

---

**Note:** A graphical colouring mode exists for Meteorological Stations, so that they can be visualised in the single line graphic.

---

## 15.8 Operators

For descriptive purposes, it is useful to sort network components according to their operators. Additionally, system operators may find it advantageous to generate summary reports of the losses, generation, load, etc. according to their designated region(s). *PowerFactory* allows the definition of operators, the assignment of network components to these operators, and the identification of operators on single line diagrams by means of operator elements (*ElmOperator*, ), these are stored in the *Operators* folder () within the *Network Data*.

There are different ways to define a new operator:

1. From the single line diagram:
  - Right click on the element/s
  - Select *Network Groupings* → *Operator* → *New...*
2. Using the drawing tools
  - Click on the *Operator* icon from the drawing tools ()
  - With the left mouse button held down, circle the elements where the operator should be assigned
3. From the Data Manager:
  - Open the Operators folder from the Network Data
  - Click on the *New Object* icon
  - Select operator

The first two options will also insert a graphical representation of the operator in the single line diagram. The Data Manager option will not automatically create a graphical representation of the operator. Graphical representations of existing operators can be inserted in a single line diagram using the Diagram Layout Tool.

Network elements (class name *Elm\**) such as terminals, switches, lines, generators, transformers, relays, composite models, substations and Branches can be assigned to an operator by means of the reference *Operator* from the *Description* page of their edit dialog.

## 15.9 Owners

For descriptive purposes it is useful to sort network components according to their owners. Additionally, for network owners it may prove advantageous to generate summary reports of the losses, generation, load, etc. for their region(s). Similar to Operators, *PowerFactory* allows the definition of network owners, and the assignment of network components to them, by means of Owner objects.

The Owner objects (*ElmOwner*, ) are stored in the 'Owners' folder () in the *Network Data* directory. They are created following the same procedure described for operators. Network elements (class name *Elm\**) such as terminals, switches, lines, generators, transformers, relays or composite models (*ElmComp*), Substations (*ElmSubstat*) and Branches (*ElmBranch*) can be assigned to an operator by means of the reference 'Operator' from the Description page of their dialog.

## 15.10 Paths

A path (*SetPath*, ) is a set of two or more terminals and their interconnected objects. This is used primarily by the protection module to analyse the operation of protection devices within a network.

The defined paths can be coloured in a single line graphic using the colouring function. New paths are stored inside the *Paths* folder (\*\*\* in the *Network Data* directory.

To create a new Path:

- In a single line diagram select a chain of two or more terminals and their inter-connecting objects.
- Right-click on the selection.
- Select the option *Network Groupings* → *Path* → *New...* from the context menu.
- The dialog of the new path pops up, give a name and select the desired colour for the corresponding colour representation mode in the single line diagram. The references to the objects defining the Path (First/Last Busbar First/Last Branch) are automatically created by the program, according to the selection.
- After pressing **Ok** the new path is stored in the Paths folder of the Network Data.

Alternatively, the path can be created as follows:

- In the single line diagram, select the first element to be included in the path definition by pressing left mouse button and select the last element in combination with **Ctrl+Shift** keys. As a result, the shortest path between the first and last element will be highlighted.
  - In order to constrain the route, additional elements can also be selected in combination with **Ctrl** key.
  - The paths with multiple end terminals can be created by selecting the multiple end terminals using left mouse button in combination with **Ctrl+Shift** keys.
  - In order to consider the state of switches and to define a path consisting of closed switches only, select the last element of the path using left mouse button in combination with **Ctrl+Shift+S** keys.
- Next, right-click on the selection.
- Select the option *Network Groupings* → *Path* → *New...* from the context menu.
- Consequently, a new path is created. The names of the first and the last elements are used for defining the corresponding path name. However, the name can be further edited, if required.

By using the **Elements** button of the Path dialog you can have access to all the element belonging to the path in a data browser, there you can edit them. The **Select** button may be used to locate the components of the path in a single line diagram. With the **Toggle** button you can invert the order of the objects limiting the path (First/Last Busbar First/Last Branch). This order is relevant when evaluating directional protective devices.

In cases where a path forms a closed ring the **First Busbar** button of the SetPath dialog can be used to specify at which busbar the path should be considered to begin and end. This can be particularly useful when displaying the path on a time distance diagram.

New objects can be added to a path by marking them in a single line diagram (including one end of the target path and a busbar as the new end) right-clicking and selecting *Network Groupings* → *Path* → *Add to...* from the context menu. Objects can be removed from a Path (regarding that the end object of a Path must be always a busbar) by marking them in the single line diagram, right-clicking and selecting *Network Groupings* → *Path* → *Remove Partly* from the context menu. The option *Network Groupings* → *Path* → *Remove* will remove the firstly found path definition of which at least one of the selected objects is a member.

For information about the colouring function refer to Chapter 10: Network Graphics. For information about the use of the path definitions for the analysis of the protective devices, refer to Chapter 33 (Protection).

## 15.11 Routes

Routes are objects which are used to group line couplings (tower elements). Each coupling (*ElmTow*) can have a reference to any defined route (*ElmRoute*, ). Each route has a colour that can be used to identify it in single line diagrams, when the corresponding colouring function is enabled.

For information regarding line couplings refer to the technical reference for the transmission line model (see [Technical References Document](#)).

## 15.12 Zones

Components of a network may be allocated to a zone object (*ElmZone*, ) in order to represent geographical regions of the system. Each zone has a colour which can be used to identify the elements belonging to it in the single line graphic if the colouring is set to *Groupings → Zones*. These elements can be listed in a browser format for group editing; additionally all loads belonging to the zone can be quickly scaled from the zone edit dialog. Using the 'Scaling Criteria'-Button, the total active or apparent power demand can be set as absolute values in MVA or MW. The zone scaling factor is then calculated accordingly. Another special scaling factor is provided for all generators, its plant category is set to 'Wind' and which are not part of other controllers: The Wind Generation Scaling Factor.

Reports for the defined zones can be generated following calculations.

Upon being defined, zones are by default stored inside the Zones folder () in the *Network Data* folder.

New zones can be defined directly from the single line diagram, using the drawing tools, as follows:

- Click on the *Zone* icon from the drawing tools ()
- With the left mouse button held down, circle the elements that should belong to the zone, making sure that at least one terminal is within the selection.
- A graphical representation of the zone will be automatically added to the diagram.

The option *Network Groupings → Zone → Add to...* can be selected when a zone(s) have already been defined. Single-port elements are automatically assigned to the zone of the node that they are connected to. For multi-port elements (such as lines or transformers), one of the available terminals has to be chosen, from which the zone assignment is inherited.

---

**Note:** The context menu of the elements can also be used to define a zone: *Network Groupings → Zone → New...*

---

New zones can also be defined in the Data Manager; however, using this option will not automatically create a graphical representation of the zone. Graphical representations of existing zones can be inserted in a single line diagram using the Diagram Layout Tool.

# Chapter 16

## Operation Scenarios

### 16.1 Introduction

Operation Scenarios are used to store operational data such as generator dispatch, load demand, and network line/switch status. Individual Operation Scenarios are stored within the **Operations Scenarios** folder, and can be easily activated and deactivated. This Chapter describes *PowerFactory* operation scenarios.

---

**Note:** Parameter Characteristics can also be used to modify network operational data - see Section [18.2 \(Parameter Characteristics\)](#) for details.

---

### 16.2 Operation Scenarios Background

*Operation Scenarios* are used to store network component parameters that define the operational point of a system. Examples of operational data include generator power dispatch and a load demand. Operational data is typically distinguished from other component data because it changes frequently. Compare for instance, how often a generator changes its power set-point, with how often the impedance of the generator transformer changes.

Storing recurrent operation points of a network and being able to activate or deactivate them when needed accelerates the analyses of the network under different operating conditions. *PowerFactory* can store complete operational states for a network in objects called operation scenarios (*IntScenario*, ).

Operation scenarios are stored inside the operation scenarios folder () in the project directory. You can define as many operation scenarios as needed; each operation scenario should represent a different operational point.

Figure [16.2.1](#) shows a project containing six operation scenarios, and the contents of the 'High Load - No Infeed' scenario (i.e. its subsets) are shown in the right pane of the Data Manager.

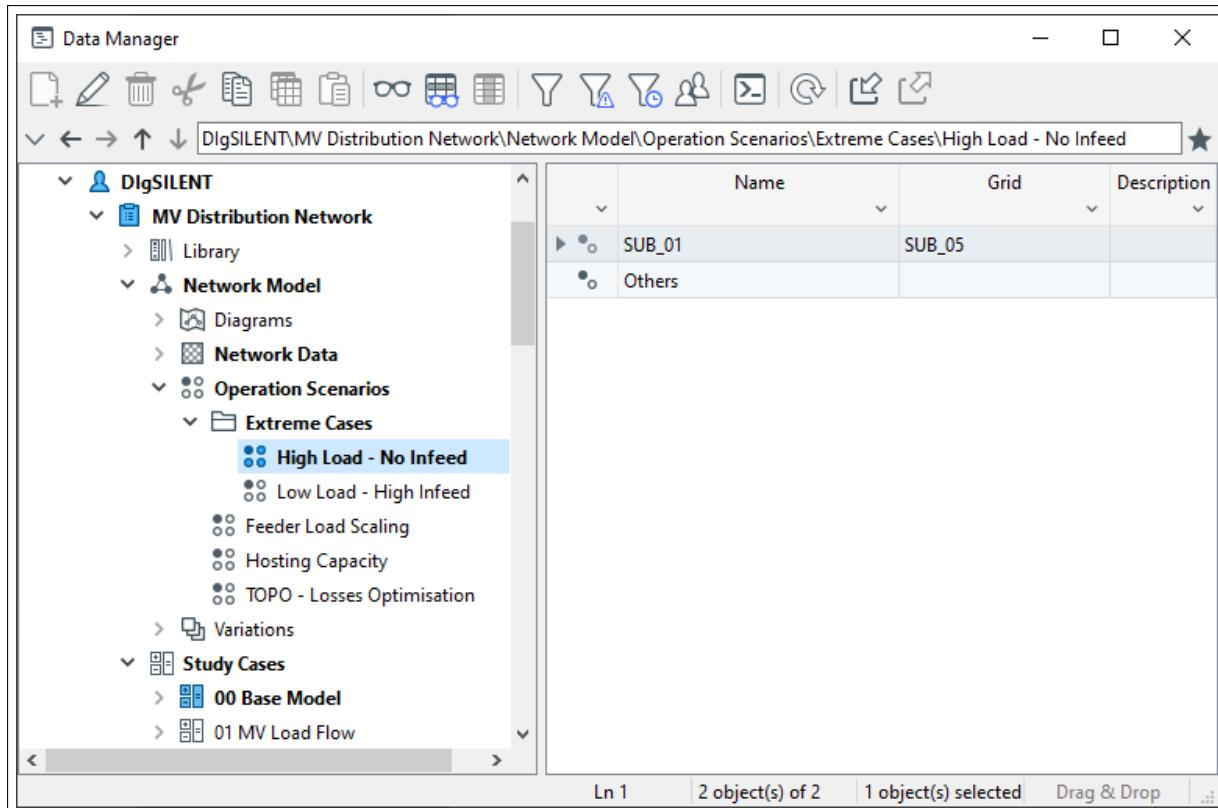


Figure 16.2.1: Operation Scenarios and Subsets

A new operation scenario is defined by saving the current operational data of the active network components. Once they have been created, operation scenarios can be activated to load the corresponding operational data. If an operation scenario is active and certain operational data is changed, these changes are stored in the active operation scenario (if you decide to save the changes). If the current operation scenario is deactivated, the active network components revert to the operational data that they had before the activation of the operation scenario (this is the 'default' operational data). Changes made to the 'default' operational data do not affect data within existing operation scenarios.

Operation scenario data stored within each operation scenario is separated into subsets, with one subset of operational data created for every grid in the network model. It is possible to 'exclude' the operational data for individual grids. This prevents the operation scenario from saving the operational data for any subset where this option is active. For example, you might be working with a network model with four grids, say North, South, East and West. Perhaps you do not wish to store operational data for the 'West' grid because the models in this grid have fixed output regardless of the operational state. By excluding the operational data subset for this grid, the default data can be used in all cases, even though the operational data is different in the other three grids.

When working with active operation scenarios and active expansion stages, modifications on the operational data are stored in the operation scenario whereas the expansion stage keeps the default operational data and all other topological changes. If no operation scenarios are active and new components are added by the current expansion stage, the operational data of the new components will be added to the corresponding operation scenario when activated.

---

**Note:** When an operation scenario is active, the operational data can be easily identified in network component dialogs and in a Network Model Manager because it is shown with a blue background. The colouring is configurable by the user: see Section 7.8

---

## 16.3 How to use Operation Scenarios

This sub-section explains how to complete the common tasks you will need when working with operation scenarios. The most common tasks are creating a new operation scenario, saving data to an operation scenario, Activating an existing operation scenario, Deactivating an operation scenario and identifying parameters stored within an operation scenario.

### 16.3.1 How to create an Operation Scenario

There are two ways to create an operation scenario.

#### Method 1

Follow these steps:

1. In the Data Manager, right-click on the *Operation Scenarios* folder in the active project.
2. Select *New → Operation Scenario...* from the context menu as shown in Figure 16.3.1. The dialog of the new operation scenario pops up.

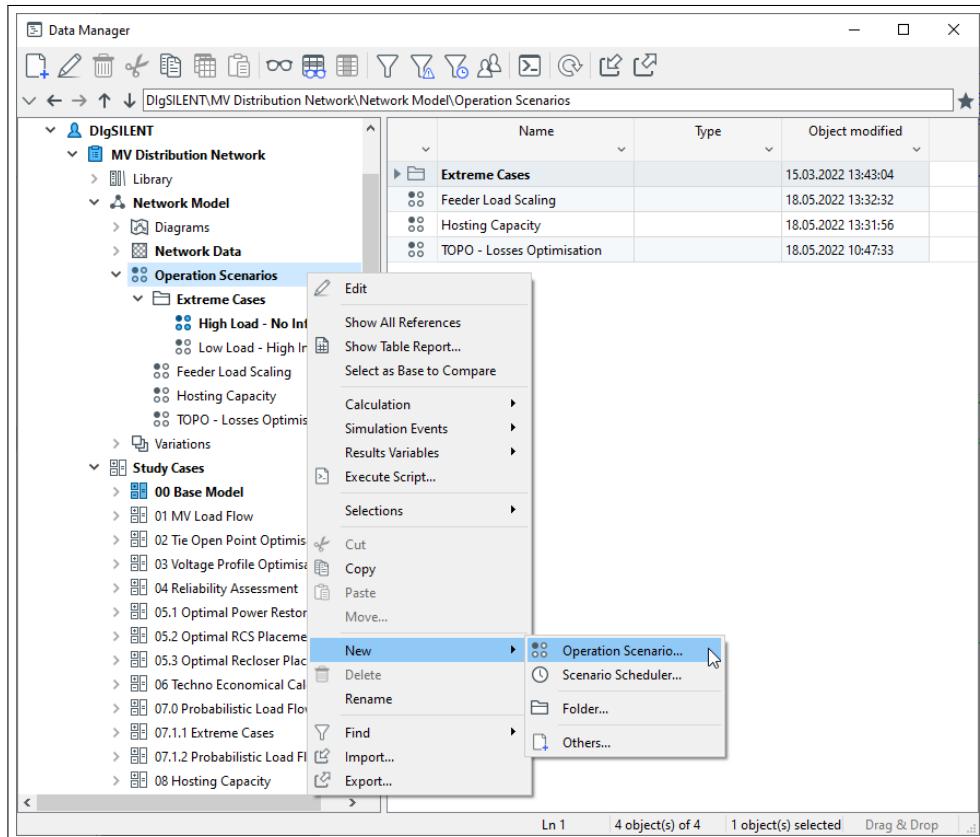


Figure 16.3.1: Creating a new operation scenario object using the Data Manager.

3. Enter the name for the operation scenario in the name field.
4. Press **OK**. The operation scenario will appear as a new object within the Operation Scenarios folder.

#### Method 2

Follow these steps:

1. From the main *PowerFactory* menu go to the File menu and select *File → Save Operation Scenario as...*. The dialog of the new operation scenario pops up.
2. Enter the name for the operation scenario in the name field.
3. Press **OK**. The new operation scenario is created within the operation scenarios' project folder and automatically activated and saved.

### 16.3.2 How to save an Operation Scenario

#### Why do you need to save Operation Scenarios?

Unlike all other *PowerFactory* data, changes to operational data are not automatically saved to the database if an operation scenario is active. So, after you update an operation scenario (by changing some operational data) you must save it. If you prefer automatic save behaviour, you can activate an automatic save option setting - see Section [16.6.1](#).

#### How to know if an Operation Scenario contains unsaved data

If any operational data (of a network component) is changed when an operation scenario is active, the unsaved status of it is indicated by an asterisk (\*) next to the icon for the operation scenario as shown in Figure [16.3.2](#). The other situation that causes an operation scenario icon to appear with an asterisk is when new network components are added to the model. Any operational parameters from these models are not incorporated in the active operation scenario until it is saved.

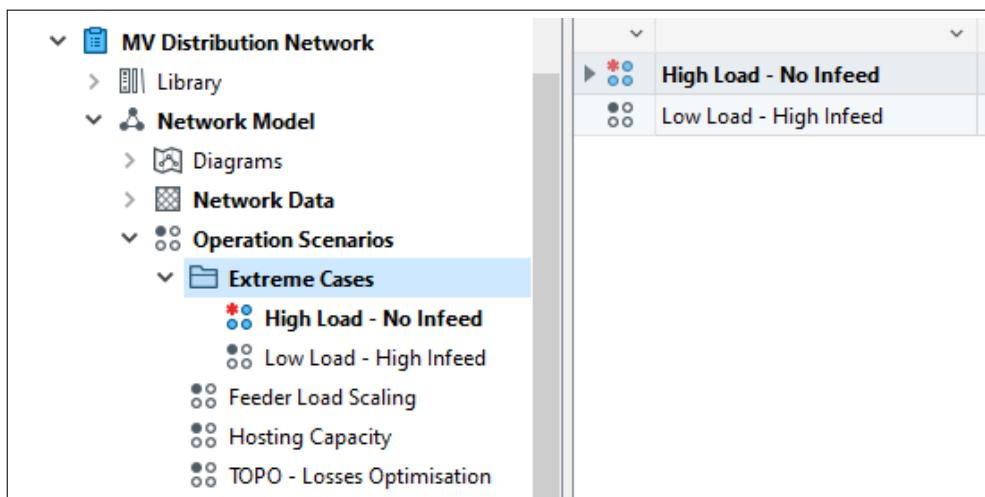


Figure 16.3.2: An asterisk indicates unsaved changes in operation scenarios

#### Options for saving an Operation Scenario

There are four ways to save a modified operation scenario to the database. They are:

- The menu entry *Save Operation Scenario* in *PowerFactory*'s main file menu.
- The button **Save** in the dialog window of the operation scenario.
- The button *Save Operation Scenario* () in the main icon bar.
- The context menu (right mouse button) entry *Action -> Save* of the operation scenario (see Figure [16.3.3](#)).

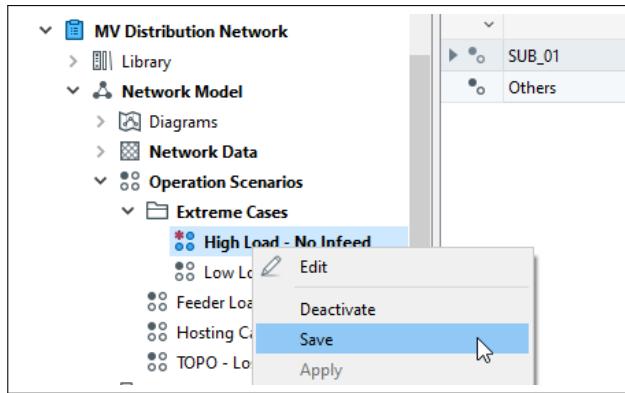


Figure 16.3.3: Saving an operation scenario using the context menu

**Note:** The button **Save as** from the operation scenario dialog (only available for active operation scenarios) can be used to save the current operational data as a new operation scenario. The new operation scenario is automatically activated upon being created.

### 16.3.3 How to activate an existing Operation Scenario

Switching between already available operation scenarios is a common task. There are two methods for activating an existing operation scenario.

#### Method 1

Follow these steps:

1. Go to the operation scenarios' folder within your project using the Data Manager.
2. Right-click the operation scenario that you wish to activate. The context menu will appear.
3. Choose the option *Activate* from the menu. If a currently active operation scenario contains unsaved data, you will be prompted to save or discard this information.

#### Method 2

Follow these steps:

1. From the main file menu choose the option *Activate Operation Scenario*. A pop-up dialog will appear, showing you the available operation scenarios.
2. Select the operation scenario you wish to Activate and press **OK**. If a currently active operation scenario contains unsaved data, you will be prompted to save or discard this information.

**Note:** The active operation scenario can be displayed in the status bar. To do this right-click the lower right of the status bar and choose *Display Options* → *Operation Scenario*.

### 16.3.4 How to deactivate an Operation Scenario

There are two ways to deactivate an active operation scenario.

#### Method 1

Follow these steps:

1. Go to the 'operation scenarios' folder within your project using the Data Manager.
2. Right-click the operation scenario that you wish to deactivate. The context menu will appear.
3. Choose the option deactivate from the menu. If the operation scenario contains unsaved data, you will be prompted to save or discard this information.

## Method 2

From the main file menu choose the option *Deactivate Operation Scenario*. If the operation scenario contains unsaved data, you will be prompted to save or discard this information.

**Note:** On deactivation of an operation scenario, previous operational data (the 'default' operational data) is restored.

### 16.3.5 How to identify operational data parameters

Because the operation scenario only stores a subset of the network data, it is useful to know exactly what data is being stored by the operation scenario. This is relatively easy to see when you have an active scenario. Data that is stored in the operation scenario is shown with a blue background. This appears in both the object dialogs and the Data Manager browser as shown in Figures 16.3.4 and 16.3.5.

The colouring is configurable by the user: see Section 7.8.

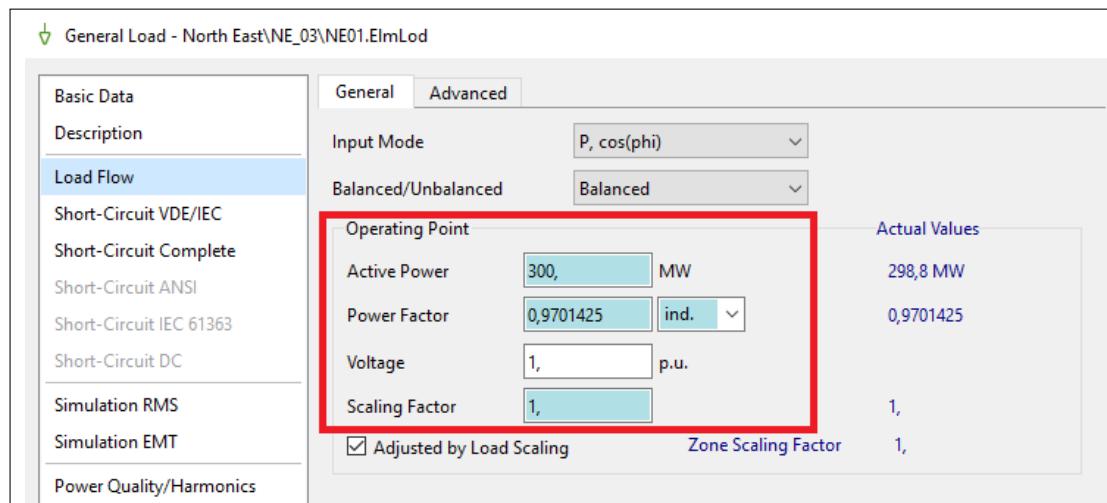


Figure 16.3.4: Blue highlighted operational data in an element dialog

The screenshot shows the Network Model Manager interface with a tree view on the left and a table view on the right. The tree view includes categories like Network Components, Load Flow Controller Models, and Protection Devices. The table view lists components with columns for Name, In Folder, Grid, Input M..., Balance..., and Act.Pow. MW. A red box highlights the 'Act.Pow. MW' column.

	Name	In Folder	Grid	Input M...	Balance...	Act.Pow. MW
NE01	NE_03	North East	PC	0	300,	
NE02	NE_03	North East	PC	0	400,	
NE03	NE_02	North East	PC	0	400,	
NE04	NE_04	North East	PC	0	800,	
NW01	NW_02	North West	PC	0	400,	
NW02	NW_03	North West	PC	0	700,	
NW03	NW_03	North West	PC	0	500,	
SE01	SE_02	South East	PC	0	400,	
SE02	SE_01	South East	PC	0	480,	
SE03	SE_03	South East	PC	0	400,	
SE04	SE_03	South East	PC	0	70,	
SW05	SW_02	South West	PC	0	400,	
SW06	SW_03	South West	PC	0	800,	

Figure 16.3.5: Blue highlighted operational data in a browser window

## 16.4 The Operation Scenario Manager

The Operation Scenario Manager is a tool which has been introduced to enable users who work with operation scenarios to compare data between scenarios and easily copy data values from one scenario to another.

It will be noted that some of the “How to” tasks covered in Section 16.5 also enable the user to view and copy data. In other words, there can be different ways of achieving the same result, but the choice of method will in the end depend on the precise task at hand. The Operation Scenario Manager is particularly useful for viewing the same data attributes for various scenarios together.

### 16.4.1 Accessing the Operation Scenario Manager

The Operation Scenario Manager can be accessed via the Network Model Manager (see Chapter 11), presenting itself as a special *Scenarios* tab (the second of four “special” tabs) within the network model manager itself. Alternatively, the Operation Scenario Manager can be started from the context menu accessed by right-clicking on the title heading of the “Operation Scenarios” section in the Project Overview window.

When the Scenarios tab is used for the first time in a project, there will be little to see, because no scenarios have been selected to be viewed, and the configuration of variables to be shown for each element class has not yet been set up.

## 16.4.2 Selecting scenarios and setting up variable configurations

In the descriptions below, the references such as “(1)” refer to this figure, 16.4.1:

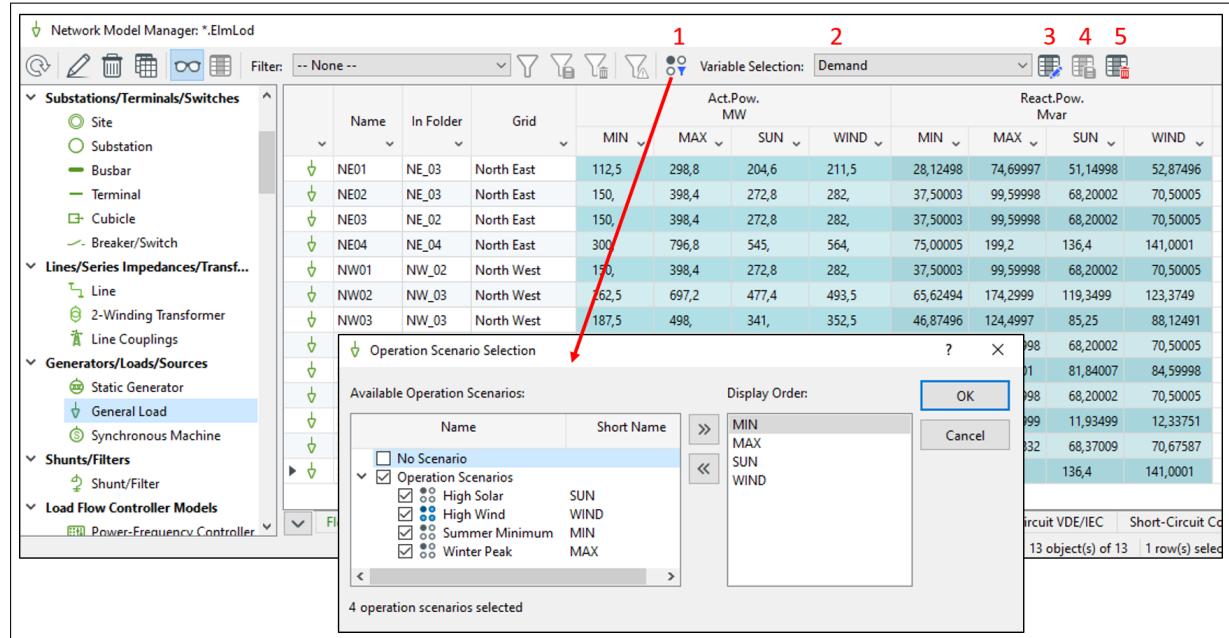


Figure 16.4.1: The Operation Scenario Manager

### 16.4.2.1 Selecting scenarios

The *Select Operation Scenarios* button (1) is used to specify for which scenarios the data shall be displayed. An option to see the “No Scenario” data i.e. the underlying values, is also available.

It can be seen in the *Operation Scenario Selection* dialog that a *Short Name* is listed in addition to the full scenario name. This scenario attribute is used for the column headings, as a way of avoiding the practical difficulties of having many scenarios with long names only differing by characters at the end of the name. If the user has not populated this field, the full name will be used instead.

### 16.4.2.2 Defining variable configurations

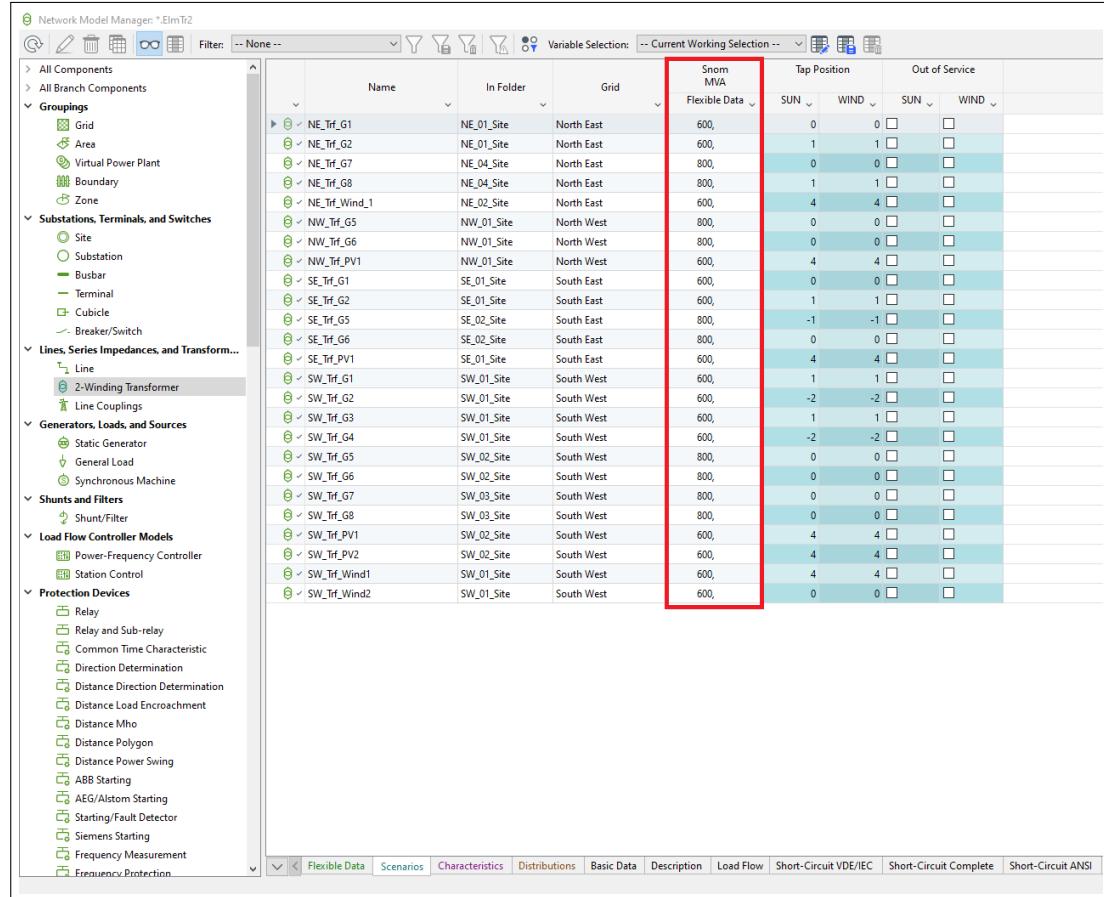
The variables to be displayed must also be selected. The concept is that the Operation Scenario Manager can be used with a number of different customised Variable Selection configurations, which the user first sets up and can then select as required from the drop-down Variable Selection menu (2).

When the Operation Scenario Manager tab is opened for the first time, the Variable Selection field will show “- - None - -”. The user can then select an element class from the left hand pane of the network model manager and click on the *Edit Variable Selection* button (3) to start setting up the configuration. Variables are selected via standard variable selection (\*.IntMon) dialogs, and the configuration can include one or more element classes.

**Note:** The Variable Selection object (\*.IntMon) is described in Section 19.3: [Variable Selection](#).

When the variable selection dialog is displayed using the operation scenario manager, the variables are filtered by default to show only operational variables. This filter can be deselected using the *Scenario relevant only* checkbox.

If non-operational variables are selected, then the operation scenario manager will display an additional column for each selected variable with the column header *Flexible Data* assigned as highlighted in Figure 16.4.2.



The screenshot shows the Network Model Manager interface with a tree view on the left and a table view on the right. The table has columns for Name, In Folder, Grid, Snom MVA, Tap Position (SUN, WIND), and Out of Service (SUN, WIND). A red box highlights the 'Flexible Data' column, which contains numerical values (e.g., 600, -1, 4) corresponding to the selected variables. The table also includes rows for various components like Trf\_G1, Trf\_G2, etc., across different sites and grids.

Figure 16.4.2: Flexible Data Column in Operation Scenario Manager

Once the user starts setting up the configuration, this then becomes the “- - Current Working Selection - -”. When all the required variables have been selected, the configuration can be saved (4) with a user-specified name.

Configurations can be edited at any time, or deleted (5) as required.

### 16.4.2.3 Location of variable configurations within the project

The user can set up a number of variable selection configurations, and these are stored within the project Settings folder. Within this folder, there is a *Scenario Manager Selection Set*, and this can contain any number of folders, each of which will then have one or more \*.IntMon variable definitions, as shown in Figure 16.4.3 below. It is these folders which are the named configurations that are selected via the drop-down menu seen in Figure 16.4.1 (2).

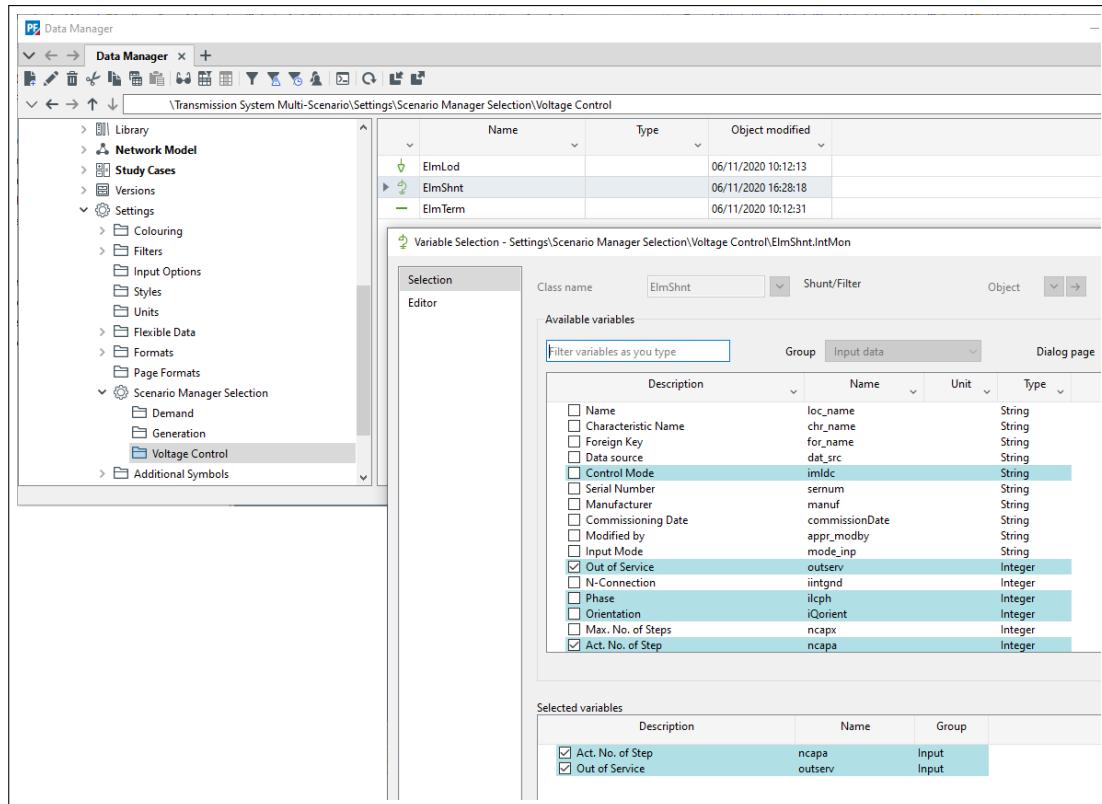


Figure 16.4.3: Variable Selection configurations for the Operation Scenario Manager

### 16.4.3 Viewing and editing data within the Operation Scenario Manager

When scenario data is viewed in the Operation Scenario Manager, the data is grouped into blocks, one for each data attribute selected in the configuration. Within each block, the values for each of the selected scenarios are seen. The usual sorting and filtering options are available.

In addition to being able to view the data, the user can also modify values as expected in the Network Model Manager, including copying and pasting from one column to another.

---

**Note:** It is very important for users to realise that when scenario data is changed in any scenario which is not currently active, that change will be automatically saved in the scenario. Only in the currently-active scenario will the normal option remain, to save or discard data changes. Care must be taken therefore, when making any changes to data in the Operation Scenario Manager.

---

## 16.5 Working with Operation Scenarios

In this section the operation scenario administrative tasks are explained. This includes reporting operational scenario data status, comparing operation scenarios, viewing the non-default running arrangements, applying data from one operation scenario to another (copying), updating the base network model, excluding grids from the operation scenario and creating a time based operation scenario.

### 16.5.1 How to view objects missing from the Operation Scenario data

When you add a component to a network, the data is not automatically captured in the active operation scenario until you save the scenario. The operation scenario appears with an asterisk next to its name in the Data Manager. If you want to get a list of all the objects that have operational data that is missing from the active scenario, then you need to print the operation scenario report. To do this, follow these steps:

1. Open the active operation scenario dialog by finding the operation scenario in the Data Manager right-clicking it and selecting *Edit* from context menu.
2. Press the **Reporting** button. A list of objects with data missing from the operation scenario is printed by *PowerFactory* to the output window.

---

**Note:** If you double click a listed object in the output window the dialog box for that object will open directly allowing you to edit the object. You can also right-click the name in the output window and use the function 'Mark in Graphic' to find the object.

---

### 16.5.2 How to compare the data in two operation scenarios

It is sometimes useful to compare data in two separate operation scenarios so that key differences can be checked. To compare two operation scenarios:

1. Deactivate all operation scenarios that you wish to compare. Only inactive operation scenarios can be compared.
2. Open the first operation scenario dialog by finding the operation scenario in the Data Manager right-clicking it and selecting edit from context menu.
3. Press the **Compare** button. A data window browser will appear.
4. Choose the second operation scenario and press **OK**. A report of the operation scenario differences is printed by *PowerFactory* to the output window.

### 16.5.3 How to view the non-default Running Arrangements

Any running arrangements that are assigned to substations will be stored as part of the operational data. The operation scenario has a function that allows you to view any substations with active running arrangements that are different from the default running arrangement for that substation. The default running arrangement is determined by the running arrangement that is applied to the substation when no operation scenarios are active. To view all the non-default Running Arrangements follow these steps:

1. Open the active operation scenario dialog by finding the operation scenario in the Data Manager, right-clicking it and selecting edit from context menu.
2. Press the **Reporting RA** button. *PowerFactory* prints a report of the non-default Running Arrangements to the output window.

---

**Note:** Most of these actions are also available in context menu when right-clicking on an operation scenario (*Action* → ...).

---

### 16.5.4 How to transfer data from one Operation Scenario to another

As explained in the chapter introduction, within each operation scenario there is a subset of operation scenario data for each grid in the network model. Therefore, there are two options when transferring data from one operation scenario to another, either copying all the operation scenario data at once, or only copying a subset of data for an individual grid. Both methods are explained within this section.

Furthermore, whether operational data is to be transferred for the whole scenario or one grid only, it is also possible to be selective about which data is transferred, by setting up and using *Scenario Apply Configurations*.

#### 16.5.4.1 Transferring operational data from one grid only

To transfer the operational data from a single grid subset to the same grid subset of another operation scenario follow these steps:

1. Activate the target operation scenario.
2. Right-click the source operation scenario subset.
3. From the context menu select *Apply*. A pop-up dialog will appear asking you if you really want to apply the selected operational data to the active operation scenario.
4. Click **OK**. The data is copied automatically by *PowerFactory*. Warning, any data saved in the equivalent subset in the active scenario will be overwritten. However, it will not be automatically saved.

#### 16.5.4.2 Transferring operational data from a complete operation scenario

To transfer the operational data from a complete operation scenario to another operation scenario follow these steps:

1. Activate the target operation scenario.
2. Right-click the source operation scenario.
3. From the context menu select *Apply*. A pop-up dialog will appear asking you if you really want to apply the selected operational data to the active operation scenario.
4. Click **OK**. The data is copied automatically by *PowerFactory*. Warning, any data saved in the active scenario will be overwritten. However, it will not be automatically saved.

#### 16.5.4.3 Transferring selective operational data using *Scenario Apply Configurations*

If the user wants to be selective about which data is transferred, this can be done by setting up *Scenario Apply Configurations* within the project, then using these in conjunction with the *Apply* command.

##### **Creating Scenario Apply Configurations**

If creating *Scenario Apply Configurations* for the first time in a project, first carry out this step in the *Settings* folder of the project:

- Click on the *Settings* folder and use the *New Object* icon to create an object *Scenario Apply Configurations (SetOpselection)*.

Once this is created, one or more scenario apply configuration folders may be created within it. Each will define a set of data items to be copied when that folder is selected in the *Apply* command. To create and populate each folder, follow these steps:

- First use *New Object* icon to create a folder within the *Scenario Apply Configurations (SetOpdselection)*. Give it a meaningful name such as “Generator MW”.
- Within the folder, use *New Object* icon to create one or more *Variable Selection (IntMon)* objects. These are used to specify an element class (e.g. *ElmSym*) and which variables (e.g. *pgini*) are to be copied for this element class when this folder is selected.

#### Using *Scenario Apply Configurations*

Once the above configurations have been created, the folder names will appear as options when the *Apply* command is used to copy data from one scenario to another, so instead of applying all the data, the user would have option, using the example above, of just copying the generator MW set points by selecting “Generator MW”.

#### 16.5.5 How to update the default data with operation scenario data

As a user, sometimes you need to update the default operational data (the operational data parameters that exist in the network when no operation scenario is active) with operational data from an operation scenario within the project. To do this:

1. Deactivate any active operation scenario.
2. Right-click the operation scenario that you want to apply to the base model.
3. From the context menu select *Apply*. A pop-up dialog will appear asking you if you really want to apply the selected operational data to the base network data
4. Click **OK**. The data is copied automatically by *PowerFactory*. Warning, any data saved in the base network model will be overwritten.

#### 16.5.6 How exclude a grid from the Operation Scenario data

##### Background

By default, each operation scenario contains several subsets, one for each grid in the network model. For example, you might be working with a network model with four grids, say North, South, East and West. In such a case each operation scenario would contain four subsets. Now it might be the case that you do not wish to store operational data for the ‘West’ grid because the models in this grid have fixed output etc. regardless of the operational state. By excluding the operational data subset for this grid, the default data can be used in all cases, even though the operational data is different in the other three grids.

##### How to exclude a Grid from the Operation Scenario

1. Select an operation scenario using the Data Manager.
2. Double-click the subset of the grid that you wish to exclude (you can only see the subsets in the right panel of the Data Manager). A dialog for the subset should appear.
3. Check the ‘Excluded’ option and the operational data from this grid will not be included within the operation scenario the next time it is saved.

#### 16.5.7 How to create a time-based Operation Scenario

##### Background

By default, operation scenarios do not consider the concept of time. Therefore, when you activate a particular operation scenario, the operational parameters stored within this scenario are applied to

network model regardless of the existing time point of the network model. However, sometimes it is useful to be able to assign a 'validity period' for an operation scenario, such that if the model time is outside of the validity period, then the changes stored within the operation scenario will be ignored and the network model will revert to the default parameters.

The concept of validity periods can be enabled in *PowerFactory* by using the *Scenario Scheduler*. There are two tasks required to use a 'Scenario Scheduler'. Firstly, it must be created, and secondly it must be activated. These tasks are explained below.

### How to create a Scenario Scheduler

To create a Scenario Scheduler follow these steps:

1. Go to the operation scenarios folder within your project using the Data Manager.
2. Click the *New Object* icon . An object selection window will appear.
3. From the displayed list, choose the Scenario Scheduler (*IntScnsched*).
4. Press **OK**. The scenario scheduler object dialog will appear as shown in Figure 16.5.1. Give the scheduler a name.

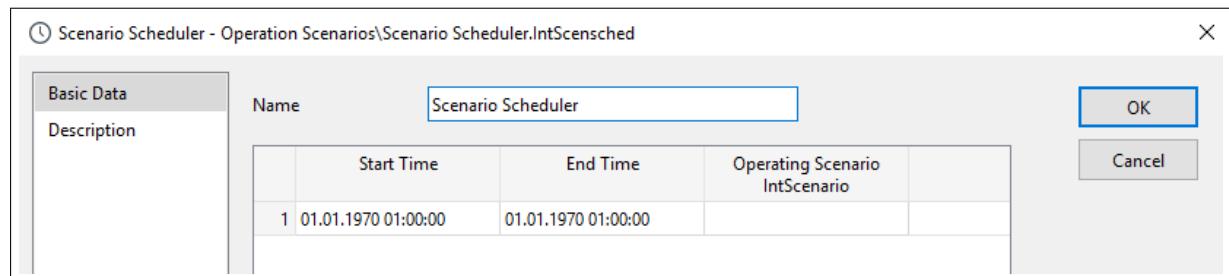


Figure 16.5.1: The Scenario Scheduler (*IntScnsched*) dialog

5. Double-click on the first cell within the operation scenario. A scenario selection dialog will appear.
6. Choose an operation scenario to schedule.
7. Adjust the start time of the schedule by double clicking the cell within the *Start Time* column.
8. Adjust the end time of the schedule by double clicking the cell within the *End Time* column.
9. Optional: To add more scenarios to the scheduler, right-click an empty area of the scheduler and *Append Rows*. Repeat steps 5-9 to create schedules for other operation scenarios.

### How to Activate a Scenario Scheduler

When first created, a scenario scheduler is not automatically activated. To activate it, follow these steps:

1. Go to the operation scenarios' folder within your project using the Data Manager.
2. Right-click the scenario scheduler object that you wish to activate and choose the option *Activate* from the context menu. The operation scenario validity periods defined within the scenario scheduler will now determine whether an operation scenario is activated automatically based on the study case time.

---

**Note:** It is possible to create more than one scenario scheduler per project. However, only one may be active. Also, if you have defined overlapping validity periods for operation scenarios within the scenario scheduler, then the operation scenario listed first (lowest row index) in the scenario scheduler will be activated and all other scenarios ignored.

---

## 16.6 Advanced Configuration of Operation Scenarios

This sub-section describes some advanced configuration options for the operation scenarios. This includes adjusting the automatic save settings and modifying the data that is stored within the operation scenarios. Note for new users, it is recommended to use the default settings.

### 16.6.1 How to change the automatic save settings for Operation Scenarios

As mentioned in Section [16.3.2](#), by default operation scenarios do not automatically save your modifications to the network data operational parameters at the time the changes are made. As a user, you can enable automatic saving of operation scenario data and you can alter the automatic save interval. It is also possible to change the save interval to 0 minutes so that all operational data changes are saved as soon as the change is made. To change the save interval for operation scenarios, follow these steps:

1. Open the *PowerFactory* User Settings by clicking the ( icon on the main toolbar).
2. Select the Data Manager page.
3. In the operation scenario section of the page, enable the option *Save active Operation Scenario automatically*.
4. Change the *Save Interval* time if you would like to alter the automatic save interval from the default of 15 minutes. Setting this value to 0 minutes means that all operation scenarios will be saved automatically as soon as operational data is modified.

---

**Note:** If an operation scenario is active any changes to the network model operational parameters are stored within such an scenario. If no operation scenario is active, then the changes are stored within the network model as usual, within a 'grid' or within a 'recording expansion stage'. A changed operation scenario is marked by a "\*" next to the operation scenario name in the status bar. In the Data Manager the modified operation scenario and operation scenario subset are also marked ().

---

### 16.6.2 How to modify the data stored in Operation Scenarios

#### Background

*PowerFactory* defines a default set of operational data for each object within the network model. This is the information that is stored within the operation scenarios. However, it is possible to alter the information that is stored to a limited extent by creating a *Scenario Configuration*. The procedure is divided into two tasks. Firstly, a special *Scenario Configuration* folder must be created and then the object definitions can be created within this folder.

#### Task 1: Creating a Scenario Configuration Folder

To create a scenario configuration folder follow these steps:

1. Go to the *Settings* folder within the project using the Data Manager.
2. Click the *New Object* icon . A object selection window will appear.
3. Choose the *Scenario Configuration (SetScenario)*. A scenario configuration dialog will appear. You can rename it if you like.
4. Press **OK**.

**Task 2: Defining the Operational Data Parameters**

Once you have created the scenario configuration folder (task 1 above), then you can create the object definitions that determine which parameters are defined as operational data. Follow these steps:

1. Deactivate any active operation scenario.
2. Open the edit dialog of the *Scenario Configuration* folder using the Data Manager.
3. Press the **Default** button. *PowerFactory* then automatically creates the object definitions according to the defaults.
4. Open the object definition that you would like to change by double clicking it. The list of default operational data parameters is shown in the *Selected Variables* panel of the dialog box that appears.
5. You can remove an operational parameter of this object by double clicking the target parameter from the *Selected Variables* panel. Likewise, a variable can be added to this list by clicking the *black triangle* underneath the cancel button and then adding the variable name to the list of parameters.
6. Once you have altered the defined parameters, click **OK**.
7. Repeat steps 4-6 for as many objects as you would like to change.
8. If an object is not in the default list of objects, it is also possible to create a new variable selection for it as follows:
  - Click on the Scenario Configuration folder in the Data Manager and then on the *New Object* icon .
  - A Variable Selection dialog will open, use the drop down menu to select the class of the object.
  - Select the variables to be included in the operation scenario.
  - Click **OK** to save the changes.
9. Open the scenario configuration object again (step 2) and press the **Check** button. *PowerFactory* will notify you in the output window if your changes are accepted.

---

**Note:** Some variables cannot be removed from the default operational parameters due to internal dependencies. If you need to remove a certain variable but the *check* function doesn't allow you to, it is suggested that you contact *DlgSILENT* support to discuss alternative options.

---

# Chapter 17

# Network Variations and Expansion Stages

## 17.1 Introduction

As introduced in Chapter 4 (*PowerFactory* Overview), Variations and Expansion Stages are used to store changes to network data, such as parameter changes, object additions, and object deletions. This Chapter describes how to define and manage Variations, and presents an example case. The term “Variation” is used to collectively refer to Variations and Expansion Stages.

The use of Variations in *PowerFactory* facilitates the recording and tracking of data changes, independent of changes made to the base Network Model. Data changes stored in Variations can easily be activated and deactivated, and can be permanently applied to the base Network Model when required (for example, when a project is commissioned).

The concept of having a “permanent graphic” in *PowerFactory* means that graphical objects related to Variations are stored in Diagrams folders, and not within Variations. When a Variation is inactive, its graphic (if applicable) is shown on the Single Line Graphic in yellow. Turning on *Freeze Mode* (🔒) hides inactive variations graphics.

When a project uses Variations, and the user wants to make changes to the base network model directly, Variations should be deactivated, or the Study Time set to be before the activation time of the first Expansion Stage (so that there is no recording Expansion Stage).

In general there are two categories of data changes stored in Variations:

1. Changes that relate to a future project (e.g. a potential or committed project). The changes may be stored in a Variation to be included with the Network Model at a particular date, or manually activated and deactivated as required by the user.
2. Changes that relate to data corrections or additions based on the current (physical) network. The changes may be stored in a Variation in order to assess the model with and without the changes, to track changes made to the model, and to facilitate reversion to the original model in case the changes are to be revised.

Notes regarding Variations and Expansion Stages:

- General:
  - The user may define as many Variations and Expansion Stages as required.
  - Variations and Expansion Stages cannot be deleted when active.
  - Variations may also be used to record operational data changes, when there is no active Operation Scenario.

- Expansion Stages are by default sorted according to their activation time in ascending order.
- To quickly show the recording Expansion Stage, project name, active Operation Scenario, and Study Case, hover the mouse pointer over the bottom right corner of the *PowerFactory* window, where (by default) the project name is shown. To change this to display the recording Expansion Stage, choose *Display Options* → 'Recording' Expansion stage.
- Activating and deactivating Variations:
  - Active Variations and Expansion Stages are shown with blue icons in the Data Manager and Project Overview.
  - The Activation Time of Expansion Stages can only be modified when the parent Variation is inactive.
  - To activate or deactivate single or multiple Variations in the Data Manager, navigate to the "Variations" folder, select and right-click on the Variation(s) and choose to activate or deactivate the selected Variation(s).
  - In the active Study Case, the "Variation Configuration" object stores the status of project Variations. It is automatically updated as Variations are activated and deactivated.
- Recording changes:
  - Whenever network data changes are recorded, an information box will appear for 10 seconds. This tells the user where (in which Variation and Expansion Stage) the changes are being recorded.
  - Elements in *PowerFactory* generally include references to Type data. Changes to Type data are not recorded in Expansion Stages. However, changes to Element Type references are recorded.
  - When there are multiple active Expansion Stages, only the 'Recording' Expansion Stage stores changes to Network Data (shown with a red circle icon and bold text). There can be only one recording Expansion Stage per study case.
  - With the exception of objects added in the active 'Recording' Expansion Stage, objects (e.g. Terminals in the base network model) cannot be renamed while there is a 'Recording' Expansion Stage.
- DPL:
  - Deleted objects are moved to the *PowerFactory* Recycle Bin, they are not completely deleted until the Recycle Bin is emptied. If a DPL script is used to create an Expansion Stage, and Expansion Stage objects are subsequently deleted, re-running the DPL script may first require the deleting of the Expansion Stage objects from the Recycle Bin. This is to avoid issues with references to objects stored in the Recycle Bin.

## 17.2 Variations

To define a new Variation (*IntScheme*):

1. First, either:
  - From the Main Menu, select *Insert* → *Variation*.
  - In a Data Manager, right-click on the *Variations* folder () and from the context menu select *New* → *Variation*....
  - In a Data Manager, select the *Variations* folder and click on the *New Object* icon . Select *Variation* (*IntScheme*) from the list and press **Ok**.
2. Define the Variation *Name*.
3. Optionally set a Variation *Colour*. This is used to highlight modifications introduced by the variation in the diagrams.

4. On the *Advanced* tab of the *Basic Data* page, optionally select *Restricted Validity Period* for the variation. If a restricted validity period is defined, the variation will effectively be ignored outside this time range. This option is not normally used, as the same effect can be achieved by having an expansion stage which changes the network and a second which restores the original state.

The “starting” and “completed” *Activation Time* are set automatically according to the Expansion Stages stored inside the Variation. The “starting” time is the activation time of the earliest Expansion Stage, and the “completed” time is the activation time of the latest Expansion Stage. If no Expansion Stages are defined, the activation time is set by default to 01.01.1970.

To activate a previously defined Variation, in the Data Manager, right-click on the Variation and from the context menu select *Activate*. The Variation and associated Expansion Stages will be activated based on their activation times and the current study case time.

In the Variation dialog, the **Contents** button can be used to list the Expansion Stages stored within the Variation.

## 17.3 Expansion Stages

To define a new Expansion Stage (*IntSstage*):

1. First, either:
  - Right-click on the target Variation and select *New* →  *Expansion Stage*....
  - Select the target Variation and click on the *New Object* button  in the Data Manager’s toolbar. Select *Expansion Stage* from the list and press **OK**.
2. Define the *Name*.
3. Set the *Activation Time*.
4. Optionally select to *Exclude from Activation* to put the Expansion Stage out of service.
5. Optionally enter Economical Data on the *Economical Data* page (see Section 44.2 (Techno-Economical Calculation) for details).
6. Press **OK**.
7. Select whether or not to set the current Study Time to the Activation Time of the defined Expansion Stage. See Section 17.5 for details.

From the Expansion Stage dialog, the following buttons are available:

- Press **Contents** to view changes introduced by the Expansion Stage.
- Press **Split** to assign changes from the recording Expansion Stage to a target (see Section 17.9.3).
- Press **Apply** to apply the changes of an Expansion Stage (only available if the parent Variation is inactive). Changes are applied to the *Network Model*, or to the recording Expansion Stage (see Section 17.9.1).

## 17.4 The Study Time

The study case Study Time determines which Expansion Stages are active. If the Study Time is equal to or exceeds the activation time of an Expansion Stage, it will be active (provided that the parent Variation is active, and provided that “Exclude from Activation” is not selected in the Expansion Stage or an active Variation Scheduler). The Study Time can be accessed from:

- The *Date/Time of Study Case* icon .

- Clicking on the lower right corner of the *PowerFactory* window, where the time of the active Study Case is displayed.
- The *Main Menu* under *Edit* → *Project Data* → *Date/Time of Study Case...* and then use the drop-down arrow to open up a standard calendar dialog.
- The Data Manager in the active Study Case folder, object “Study Time”.

## 17.5 The Recording Expansion Stage

When a Variation is activated for a study case, the active Expansion Stage with the latest activation time is automatically set to the recording Expansion Stage. If there are multiple Expansion Stages with this same activation time, the stage that previously set to the recording stage will remain as the recording Expansion Stage. Changes made to the network data by the user are saved to this stage.

As changes are made, an information box will appear for 10 seconds. This tells the user where (in which Variation and Expansion Stage) the changes are being recorded.

As discussed previously, the Study Time can be changed in order to set the active Expansion Stages, and as a consequence, set the “recording Expansion Stage”. To simplify selection of the recording Expansion Stage, in the Data Manager it is possible right-click an Expansion Stage, and select *Set ‘Recording’ Expansion stage* to quickly modify the Study Time to set a particular Expansion Stage as the recording Expansion Stage.

As noted in 17.1, unless an Operation Scenario is active, changes made to operational data are stored in the recording Expansion Stage.

## 17.6 The Variation Scheduler

As an alternative to setting the activation time of Expansion Stages individually, *Variation Schedulers (IntScheduler)* may be used to manage the activation times and service status of each Expansion Stage stored within a Variation. Multiple Variation Schedulers can be defined within a particular Variation, but only one may be active at a time. If there is no active Variation Scheduler, the Expansion Stage activation times will revert to the times specified within each individual Expansion Stage.

To define a Variation Scheduler:

1. Open a Data Manager, and navigate to the Variation where the Scheduler is to be defined. Then, either:
  - Right-click on the Variation and select *New* → *Variation Scheduler...*.
  - Click on the *New Object* button  and select *Variation Scheduler*.
2. Press the **Contents** button to open a data browser listing the included stages with their activation times and service statuses, and modify as required.

The activation time and status of Expansion Stages referred to be a Variation Scheduler can only be changed when the Variation is active, and the Variation Scheduler is inactive. Note that Expansion Stage references are automatically updated in the scheduler.

---

**Note:** If the parent Variation is deactivated and reactivated, the Variation Scheduler must be re-activated by the user, if required.

---

## 17.7 Variation and Expansion Stage Example

Figure 17.7.1 shows an example project where there are two Variations, “New Connection” and “New Line”.

With the current study case time, some expansion stages are active and some are not. Active stages are indicated by their icon being coloured blue. The recording stage is also indicated:

- Expansion Stage “Ld1” is shown with a blue icon, so is active. But it also has its name shown in bold, and this, together with the **●** marker against the icon, shows that it is the recording Expansion Stage.
- Expansion Stage “Ld2” has no colouring; it is inactive.
- Expansion Stage “Line and T2” is shown with a blue icon; it is active.

The Variation Scheduler “Scheduler1” within the “New Connection” Variation, shown with a blue icon and bold text, is active.

Therefore, the activation time and service status of each Expansion Stage within the Variation “New Connection” is determined from the activation times specified in this Variation Scheduler. The alternative Variation Scheduler “Scheduler2” is inactive (only one Variation Scheduler can be active at a time).

Also shown in Figure 17.7.1 on the right-side pane are the modifications associated with Expansion Stage “Ld1”. In this stage, a load and an associated switch and cubicle has been added.

Note that graphical changes are not included in the Variation.

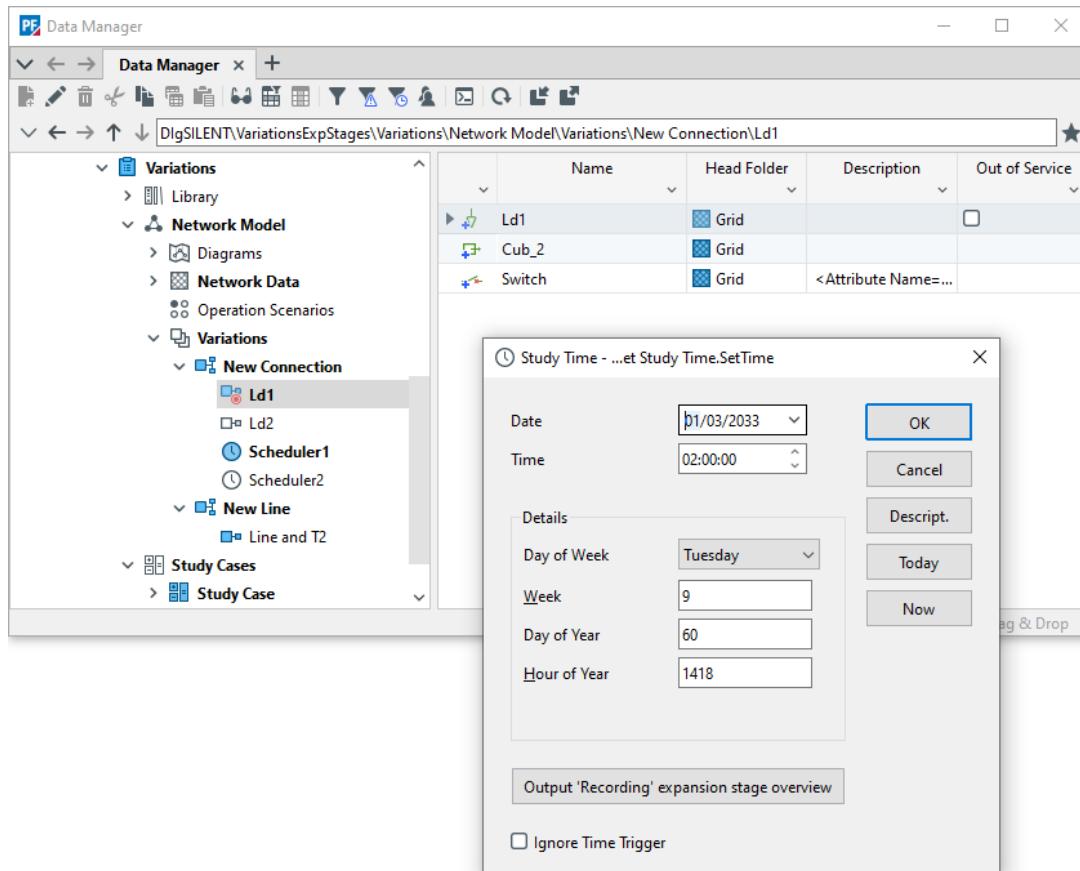


Figure 17.7.1: Example Variations and Expansion Stages - Data Manager

Figure 17.7.2 shows the “unfrozen” diagram (🔒) of the associated network. Since the Expansion Stage “Ld2” is inactive, the Load “Ld2” is shown in yellow.

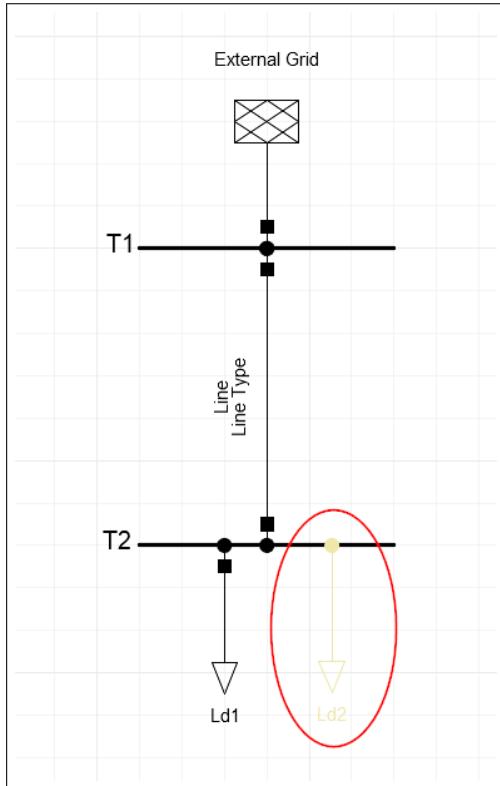


Figure 17.7.2: Example Variations and Expansion Stages - Single Line Graphic

## 17.8 The Variation Manager

The Variation Manager is a tool which enables users to have an overview of their Network Variations and Expansion Stages in a Gantt-chart like format. It can be accessed from the main tool bar, using the icon *Variation Manager* 🔗. In Figure 17.8.1 below, the Variation Manager is used to show several Variations in a project. In this project, the Network Variations have been arranged into subfolders, according to the year. Each Variation is represented by a blue horizontal bar in the chart, and each Expansion Stage by a short vertical bar. The sections following describe the options and features available in the tool, with reference to this example.

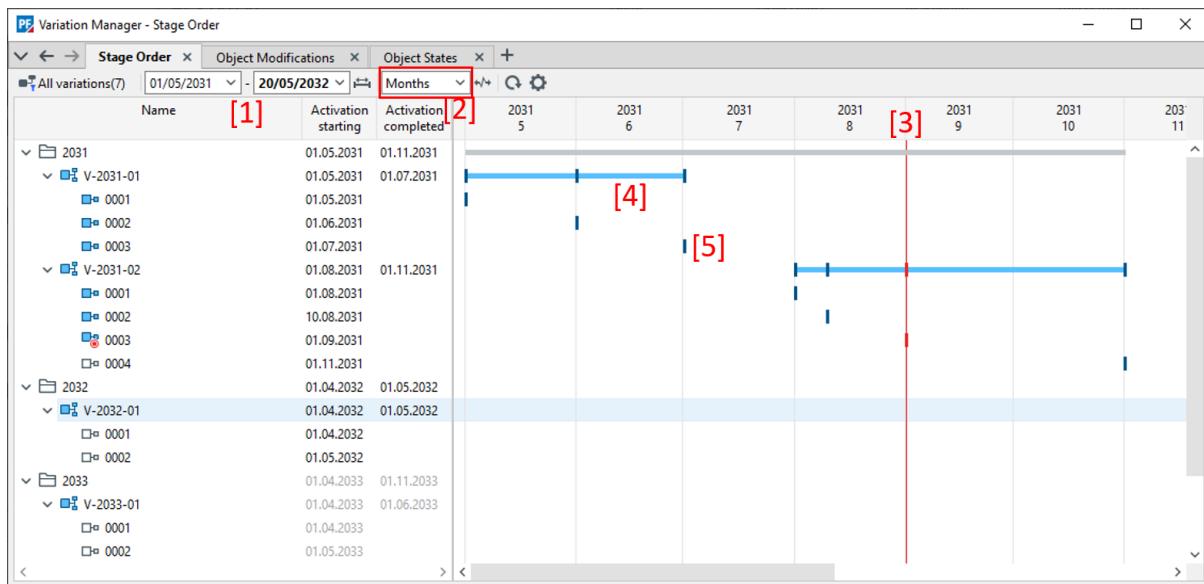


Figure 17.8.1: Variation Manager, Stage Order tab

## 17.8.1 Stage Order

### 17.8.1.1 Selecting the Variations to be shown

By default, all the Variations will be shown in the Variation Manager, but the user can restrict what is shown by clicking on the icon. Then, Variations and/or folders containing Variations can be selected as required. Several options for selecting Variations are offered, including an option to select Variations relating to a particular network element. The Variations are listed on the left-hand side, with two columns ("Activation starting" and "Activation completed") to indicate when each Variation and Expansion Stage is active. If the project contains Variation Schedulers, another column will appear to indicate which Variations include a Variation Scheduler.

### 17.8.1.2 Timeframe and resolution

By default, the period of time covered in the Variation Manager view will be set automatically according to the first and last Expansion Stage activation time of the selected Variations. This time period can be adjusted using the drop-down calendar from the start and end date fields [1]. If the time period is restricted by the user such that not all selected Variations can be displayed in full, this is indicated by the start and/or date being highlighted in bold, and by arrows within the chart indicating that not everything is displayed.

If the time period has been restricted, the full time period is easily restored using the icon.

In addition to managing the time period shown, the user is able to select the time resolution from a drop-down menu [2].

The "Fit time scaling into window" icon will adjust the view so that everything can be seen without the need for a scroll-bar (note that this can result in the resolution being changed). There is also a "Refresh" icon . To further adjust the view, the user can change the column widths manually (in the same way as in a spreadsheet).

### 17.8.1.3 Study case time and recording stage

If the current study case time is within the period shown in the Variation Manager, it will be shown as a red line [3]. By double-clicking on this line, the user can change the study case time, which may in turn affect which Expansion Stage is the recording stage. Note that the current recording stage is indicated on the left-hand side of the view, by the symbol , and highlighted in red in the chart.

### 17.8.1.4 Looking at the Variations in detail

On the right-hand side of the Variation Manager, each blue horizontal bar [4] represents a network Variation, which can be edited by double-clicking on the bar. Likewise, each short vertical bar [5] is linked to an Expansion Stage, and these are also editable in order to see the contents.

The amount of information shown on the left-hand side of the Variation Manager can be controlled by the user, via the Settings  icon, which also offers options for the format of the times and dates shown.

The Variations on the left-hand side can be expanded to show the individual Expansion Stages (as in Figure 17.8.1) or not, as required.

Using right-click on the Variations or Expansion Stages on the left-hand side brings up a context menu. As well as other expected options, there are two options specific to the Variation Manager:

- When only a portion of the chart is visible, **Show in Chart** can be used to scroll the chart automatically so as to see the selected Variation or Expansion Stage.
- **Include in displayed time period** can be used on Variations or Expansion Stages which are not shown because they lie outside the currently-specified time period. The displayed time period will be automatically expanded so that they can be seen.
- Selecting **Show Object Modifications** for a Variation will open up an Object Modifications tab (see Section 17.8.2) for that particular Variation.

## 17.8.2 Object Modifications

The Object Modifications tab provides detail of the modifications recorded in the selected Variation(s), at an object level. Most of the options at the top are the same as described above in Section 17.8.1 for the Stage Order tab.

As well as the button for selecting Variations, there is a button for selecting objects  (the default being “All Objects”). This button will bring up a dialog box for the selection of objects to be viewed. There is a column of checkboxes for selecting objects, and column showing the element class, which can be filtered. Objects can be selected/deselected individually, but also these four options may be used:

- **Select All**
- **Select Visible:** Elements that are visible (depending on the filter applied to the element class column) are added to the selection.
- **Deselect Visible:** Elements that are visible (depending on the filter applied to the element class column) are removed from the selection.
- **Deselect All**

In addition, the  icon allows the user to toggle between displaying the objects in a list or in a hierarchical view.

On the left-hand side, a number of columns provide information about the object modifications listed. Any of these columns can be used to filter the data: for example, filtering on the conflicts column (!) provides an easy way of checking for potential errors in the use of Variations in the project.

- **Number of references (#):** The number of times the object is referenced in the selected Variations
- **Existing in base grid (B):** Shows whether the object already exists in the underlying grid or is only added in a Variation
- **Added (+):** Indicates that the object is added in a Variation
- **Modified (\*):** Indicates that the object is modified in a Variation
- **Deleted (x):** Indicates that the object is deleted in a Variation
- **Scheduler(s):** If Variation Schedulers are present in the project, this column will be visible, to indicate affected objects
- **Conflicts (!):** Possible conflicts are identified. For example, modifying a non-existent object will flag up an error, whereas deleting a non-existent object or adding an already-existing object will flag up a warning. It should be noted that the conflict identification only takes account of the selected Variations.

Right-clicking on any of the objects brings up a context menu, with a number of options for viewing data related to this object. In particular, the option *Show Attribute Modifications* enables the user to examine in detail the changes made to the attributes of the selected object. This option will bring up a new Attribute Modifications tab, as described in the next section.

The Gantt chart on the right-hand side is similar to that for the Variations, described in 17.8.1.4 above. It shows information about the creation, modification and deletion of network elements, and each element can be accessed directly by double-clicking on the blue horizontal bar.

### 17.8.3 Attribute Modifications

The Attribute Modifications tab does not appear by default when the Variation Manager is first opened, but is generated for an object using the *Show Attribute Modifications* context menu in the Object Modifications tab, or from the Data Manager or Network Model Manager using the option *Variations* → *Show Attribute Modifications*.

#### 17.8.3.1 Viewing attribute modifications

This view shows a history for all the attributes which are modified in any of the selected Variations. Each column shows the Variation and Expansion Stage in which a change occurs, the changed values being shown in bold. The activation time of the Expansion Stage is also shown in the column heading, and the tool tip for this activation time also considers and lists schedulers and the times with and without scheduler.

It might be that some attribute changes are of little interest to the user. The Settings dialog, accessed via the Settings icon , includes an *Ignored attributes* button, which allows the user to specify (by element class) attributes which should not be considered relevant by the Variation Manager.

#### 17.8.3.2 Editing attribute modifications

As well as looking at the attributes that have been modified via the Network Variations, the user is also able to make changes directly within the expansion stages, by editing the data in the table. This can be used, for example, to make corrections if an erroneous value is spotted. The edits will directly change the relevant Expansion Stage. It is not possible, however, to change attributes which are not already modified within any of the selected/listed Variation(s).

As well as the option to edit the data in the table directly, right-clicking on a value brings up a context menu for editing, including options to apply the selected value to all the relevant Expansion Stages, or a selection of them.

### 17.8.4 Object States

The Object States tab summarises the number of states where an object was handled. An error is shown if none of the existing delta objects in the Expansion Stages actually modifies the object, i.e. if an object was “touched” but not modified.

Figure 17.8.2 shows an example of such a case. On Figure 17.8.2a, a conflict is shown for the transformer “NW\_Trf\_L”, but when selecting *Show Attribute Modifications* for the context menu of this object, the Attribute Modifications tab, shown in Figure 17.8.2b, shows no attributes modified.

The figure consists of two screenshots of the Variation Manager software interface.

(a) Objects States tab:

Name	#	!	Stage 1	Stage 2	Stage 3
NW_Trf_L	1	✖	+ 1		▲ 1
NW04	1	✓	+ 1		
NW_Trf_PV1	1	✖	▣ 1	▲ 1	
NW_PV1	1	✖	▣ 1	▲ 1	
NW_01 boundary	2	✓	▣ 1		▲ 2
NW_L4	1	✓	+ 1		
NW_04	1	✓	+ 1		

7 object(s) of 326

(b) Attribute Modifications tab:

NW_Trf_L	
P26002 - NW demand...	P26002 - NW demand...
Stage 3 - Transformer...	Stage 4 - Load conn...
24.08.2026 17:00:00	27.08.2026 17:00:00
<b>No attributes modified.</b>	

Figure 17.8.2: Unused Delta Objects in the Variation Manager

The not-required delta objects can be directly deleted from the Variation Manager via the context menu option *Delete Stage Object*, as long as the corresponding Variation is not active.

On the Object States tab it is possible to filter the variations and objects using the icons and as explained for the Stage Order and Object Modifications tabs. To simplify the display of the Expansion Stages, the columns show only Stage 1, Stage 2, Stage 3 and so on, these being the different stages for each of the modified objects. If the mouse is hovered over a cell, the complete name of the corresponding Expansion Stage will be shown. Alternatively, the *Detail Mode* icon can be used to show all the Expansion Stages.

## 17.9 Variation and Expansion Stage Management

### 17.9.1 Applying Changes from Expansion Stages

Changes stored in non-active Expansion Stages can be applied to the *Network Data* folder, or if there is an active recording Expansion Stage, to the recording Expansion Stage. To apply the changes, either:

- In the Data Manager, right-click the Expansion Stage and select *Apply Changes*, or in the Expansion Stage dialog press **Apply** (only available if the Expansion Stage is within a non-active Variation).
- In the Data Manager, select item(s) within an inactive Expansion Stage, right-click and select *Apply Changes*. If required, delete the item(s) from the original Expansion Stage.

### 17.9.2 Consolidating Variations

Changes that are recorded in a projects active Variations can be permanently applied to the *Network Data* folder by means of the *Consolidation* function. After the consolidation process is carried out, the active (consolidated) Expansion Stages are deleted, as well as any empty active Variations.

To consolidate an active Variation(s):

1. Right-click on the active study case and from the context menu select *Consolidate Network Variation*.
  2. A confirmation message listing the Variations to be consolidated is displayed. Press **Yes** to implement the changes.
  3. View the list of consolidated Variations and Expansion Stages in the output window
- 

**Note:** Variations stored within the Operational Library must be consolidated in separate actions. To consolidate a Variation stored in the Operational Library, right-click and from the context menu select *Consolidate*.

---

### 17.9.3 Splitting Expansion Stages

Changes stored in the recording Expansion Stage can be split into different Expansion Stages within the same Variation using the Merge Tool.

To split an Expansion Stage:

1. Open the dialog of the recording Expansion Stage and press **Split**. Alternatively, right-click and from the context menu select *Split*.
2. A data browser listing the other Expansion Stages from the parent Variation is displayed. Double-click on the target Expansion Stage.
3. The Merge Tool window is displayed, listing all the changes from the compared Expansion Stages. Select the changes to be moved to the “Target” stage by double-clicking on the *Assigned from* cell of each row and selecting *Move* or *Ignore*. Alternatively, double-click the icon shown in the “Target” or “Source” cell of each row.
4. Press **Split**. All the changes marked as *Move* will be moved to the target Expansion Stage, and the changes marked as *Ignore* will remain in the original “Base” stage. Once completed, the Variation is automatically deactivated.

### 17.9.4 Comparing Variations and Expansion Stages

Variations and Expansion Stages can be compared, as can any other kind of object in *PowerFactory*, using the Merge Tool. To compare objects using the Merge Tool, a “base object” and an “object to compare” must be selected. The comparison results are presented in a data browser window, which facilitates the visualisation, sorting, and possible merging of the compared objects.

Refer to Chapter 21: Data Management, Section 21.4 (Comparing and Merging Projects) for further details on use of the Merge Tool.

### 17.9.5 Colouring Variations the Single Line Graphic

The single-line graphic colouring function  offers three modes which may be used to identify changes from Variations and Expansion Stages. To set the colouring mode, go to *Diagram Colouring*, and under *Other* select *Variations / System Stages*, and the desired mode from the following:

- *Modifications in Recording Expansion Stage*. Colours can be defined for *Modified*, *Added*, and *Touched but not modified* components.
- *Modifications in Variations / System Stages*. Objects are shown in the colour of the Variation in which the object is last added or modified.
- *Original Locations*. Objects are shown in the colour of the grid or the Variation in which the object is added.

The use of colouring in network diagrams is described in Section 10.3.10.1, and more detail about the use of colours and colour palettes can be found in Section 4.7.8.

### 17.9.6 Variation Conflicts

Active Expansion Stages with the same activation time must be independent. This means that the same object can not be changed (modified, deleted, or added) in active Expansion Stages with the same activation times. If there are dependent Expansion Stages, when the Variation is activated *PowerFactory* will display an error message to the output window and the activation process will be cancelled. Other conflicts that may arise during the activation of a Variation:

- The same object is added by more than one Expansion Stage. In this case the latest addition is applied and a warning message is displayed in the output window.
- A previously deleted object is deleted. In this case the deletion is ignored and a warning message is displayed in the output window.
- An object is changed or deleted in a Expansion Stage but it does not exist. In this case the change is ignored and a warning message is displayed in the output window.
- A deleted object is changed in a Expansion Stage. In this case the change is applied to the deleted target object and a warning message is displayed in the output window.

### 17.9.7 Error Correction Mode

As well as recording the addition and removal of database objects, variations also record changes to database objects. Human error or the emergence of new information can result in a need to update a change. Suppose that at some time after the change has been introduced, the user wishes to update the change. If additional variations have been created since the change was introduced, this will be hard to achieve. The user must first remember in which Expansion Stage the change was introduced, then they must make this Expansion Stage the Recording Stage before finally updating the change or

rectifying the error. The Error Correction mode is intended to simplify this procedure. The following example illustrates use of the Error Correction Mode.

Suppose that a project is planned consisting of a base case and 2 Variations, namely Variation 1 and Variation 2. Suppose that the base case network contains a line object (*ElmLne*) of length 1 km. When Variation 1 is recorded, the length of the line is updated from the base case value to a new value of 10 km. This change is recorded in the Expansion Stage associated with Variation 1. Subsequently, the user creates Variation 2 and records a new set of changes in the Expansion Stage of Variation 2. The user makes no changes to the line object in Variation 2, but suddenly realises that the length of the line is incorrect. The length should be 15 km not 10 km. If the user makes a change to the line length while Variation 2 is recording this change will be recorded and applied while Variation 2 is activated. However, as soon as Variation 2 is deactivated, providing Variation 1 is activated, the line length will return to the 10 km value. This is incorrect and the error is therefore still present in the project.

Instead of recording the change in the recording Expansion Stage of Variation 2, the user should turn on the Error Correction Mode. This can be achieved by first ensuring that the Project Overview Window is visible. (If not, select *Window → Project Overview*). Then, by right-clicking in the Project Overview on the title line of the Network Variations section. A context menu as illustrated in Figure 17.9.1 will appear. The option Error Correction Mode should be selected from the context menu.

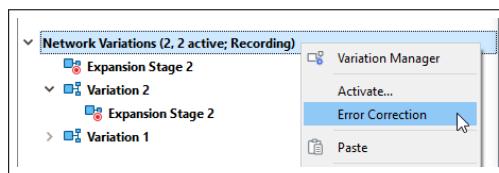


Figure 17.9.1: Activating Error Correction Mode

Once the Error Correction Mode has been switched on, any changes introduced will now, not automatically be stored in the Recording Expansion Stage. Instead, they will be stored in the Expansion Stage containing the record of the last change to the object in question. For the example described, this will be in the Expansion Stage associated with Variation 1, where the length was updated from 1 km to 10 km. The 10 km value will be updated to 15 km. If the Error Correction Mode is now switched off, again by right-clicking in the Project Overview Window, the user can proceed knowing that the error has been eliminated from the project.

Please note, if any change to the line had been recorded during Variation 2 prior to the application of the Error Correction Mode, not necessarily a change to the length of the line, but a change to any *ElmLne* parameter, then with Error Correction Mode active, the change would be recorded in the Recording Expansion Stage of Variation 2. This is because the Expansion Stage containing the record of the last change to the object in question would in fact be the one in Variation 2. In this case, the error would still be present in the project.

# Chapter 18

## Parameter Characteristics, Load States, and Tariffs

### 18.1 Introduction

This chapter provides details on how to define and use characteristics, load states, load distribution states, and tariffs.

It is useful to be aware that when element parameters have characteristics applied to them, they appear differently coloured in both the element dialogs and in a network model manager. By default, the colouring for characteristics is lilac (pale purple). Note that both the colour and its priority can be changed in the User Settings (see Section 7.8).

### 18.2 Parameter Characteristics

In *PowerFactory* any parameter may be assigned a range of values (known as a “characteristic”) that is then selectable by date and time, or by a user-defined trigger. The range of values may be in the form of a scaling factor, a one-dimensional vector or a two-dimensional matrix, such as where:

- Load demand varies based on the minute, day, season, or year of the study case.
- Generator operating point varies based on the study being conducted.
- Line/transformer ratings, generator maximum power output, etc. vary with ambient temperature.
- Wind farm output varies with wind speed, or solar farm output varies with irradiance.

The assignment of a characteristic may be made either individually to a parameter or to a number of parameters. New characteristics are normally defined in either:

- The *Characteristics* folder of the *Operational Library* within the project.
- A global *Characteristics* folder directly in the *Database → Library* (see Section 14.3: [Custom Global Library](#)).

Studies which utilise characteristics are known as ‘parametric studies’.

### 18.2.1 Scales and Triggers

The value of the characteristic is defined by the value of the scale. New scales are normally defined in the *Scales* folder within the *Characteristics* folder in the *Operational Library*.

When a scale is created, a means to 'set' the scale, and thereby to set the parameter to the corresponding value, is required. This is called a trigger (*SetTrigger*, ). When a new scale is defined, a trigger is automatically created in the *Triggers* folder of the active study case (see also Chapter 13, Section 13.12: Triggers). If the folder does not already exist, it will be created. When a trigger is edited and a 'current' value is set the scale is set and the parameter value is changed. When a different study case is activated, or a new study case is created, and a load-flow is performed, all relevant triggers are copied into the study case folder and may be used in the new study case. Triggers for characteristics may be created at any time in the Data Manager within the *Library* → *Operational Library* → *Characteristics* → *Scale* folder, or at the time the Characteristic is created. Triggers for characteristic can generally be accessed from either:

- The Date/Time of Study Case icon ().
- The Trigger of Study Case icon ().

Figure 18.2.1 illustrates an application of scales and triggers, where the study case time is used to set the output of a load based on the hour of the day.

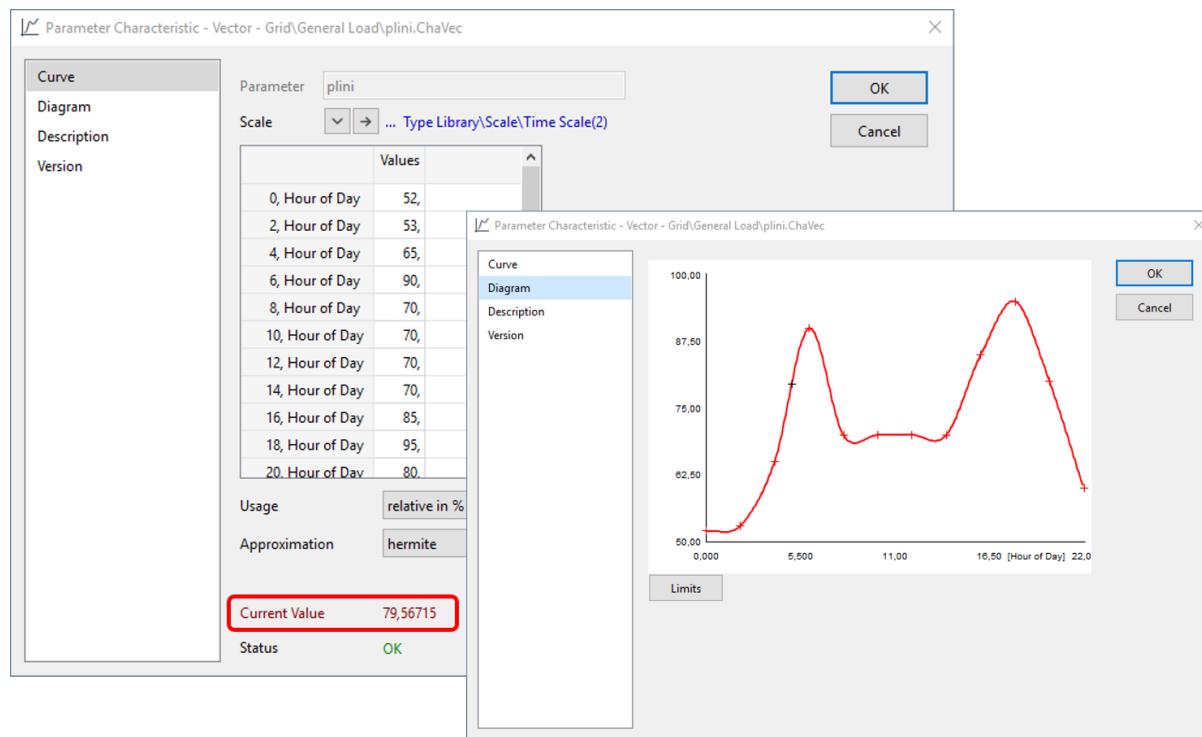


Figure 18.2.1: Illustration of Scales and Triggers

### 18.2.2 Available Characteristics

Table 18.2.1 shows a summary of the Parameter Characteristics available in *PowerFactory*. Note: Click on Characteristic description to link to the relevant section.

Characteristic	Description of Application
18.2.6: Time Characteristics	Parameter(s) are to be modified based on the day, week, or month set in the Study Time. Parameter states may be interpolated between entered values.
18.2.7: Profile Characteristics	Parameter(s) are to be modified according to seasonal variation and the day, week and month set in the Study Time.
18.2.10: Vector Characteristics with Discrete Scales	Discrete parameter states are to be selectable.
18.2.10: Vector Characteristics with Continuous Scales	Parameter states may be interpolated between entered values.
18.2.10: Vector Characteristics with Frequency Scales	Parameter(s) are to be modified with frequency.
18.2.10: Vector Characteristics with Time Scales	Parameter(s) are to be modified based on a user-defined scale referencing the Study Time.
18.2.11: Matrix Parameter Characteristics	Parameter states are based on two variables, and may be interpolated between entered values.
18.2.12: Parameter Characteristics from Files	Parameter states and the trigger (optional) is to be read from a file.
18.2.9: Linear Function	Parameter varies according to a mathematical relationship.
18.2.13: Characteristic References	Reference link between a parameter and a Characteristic.

Table 18.2.1: Summary of Parameter Characteristics

### 18.2.3 Usage

The “Usage” field at the bottom of the characteristic dialog can be used to specify how “Values” are applied to the parameter that the characteristic is associated with:

- Relative in % will multiply the parameter by the percentage value.
- Relative will multiply the parameter by the value.
- Absolute will replace the current parameter with the absolute value entered. (not available for the Scaling Factor characteristic)

When the *Absolute* option is selected, an additional field for the *Unit* is available. In this field there are two options:

- *Default unit*: the unit defined in the project settings will be used.
- *User defined*: an exponent for the unit can be specified. Then *PowerFactory* will automatically convert the entered value to the unit specified in the project settings.

### 18.2.4 Characteristic Curves

For continuous characteristics, various approximation methods are available to interpolate and extrapolate from the entered Values:

- Constant: holds the Y-value in between X-values.
- Linear: uses a linear interpolation.
- Polynomial: uses a polynomial function with a user defined degree.

- Spline: uses a spline function.
- Hermite: uses Hermite interpolation.

The approximation curve will be shown in the diagram page of the Characteristic dialog. The interpolated Y-value may vary considerably depending on the entered data and the approximation function applied.

Figure 18.2.2 highlights the difference between interpolation methods for an example characteristic with a continuous scale (shown on the horizontal axis from -20 to +45). For instance, at a trigger value of 25, linear interpolation will give an output value of 60, whereas constant interpolation will give an output value of 40.

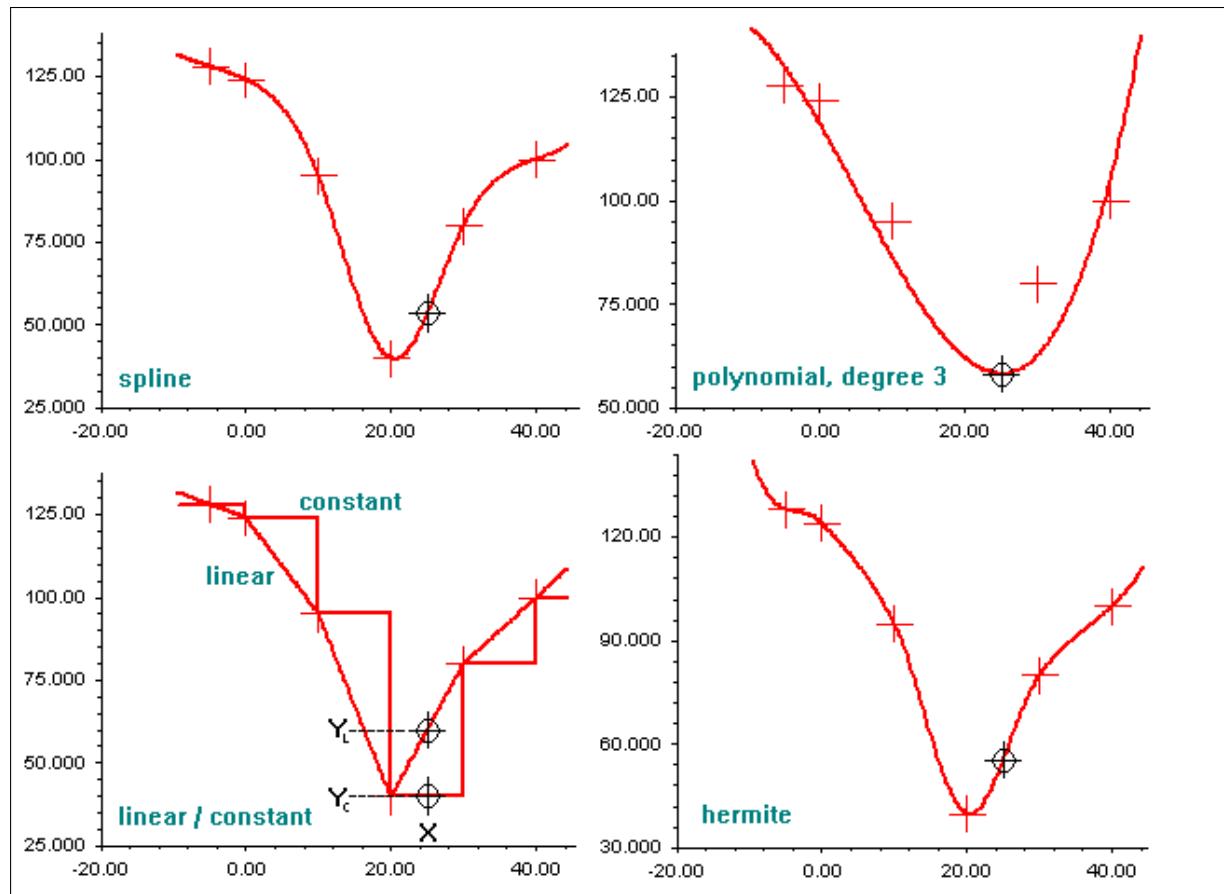


Figure 18.2.2: Approximated characteristics

Note that Approximation methods are not available for discrete characteristics.

### 18.2.5 Creating a Characteristic

To create a Characteristic, right-click on the desired parameter (e.g. 'Active Power'), right-click and select *Add project characteristic* or *Add Global Characteristic* and create the desired characteristic. It is also possible to edit the existing characteristic by selecting the option *Edit Characteristic*. Details of how to create the different types of characteristics are provided in the following sub-sections, including an example application of characteristics.

## 18.2.6 Time Characteristics

General background on characteristics and their properties is provided in Section 18.2. The time characteristic determines the value of the parameter according to the *Study time (SetTime)*.

### 18.2.6.1 Recurrence

When using the time characteristic (*ChaTime*), the user has the option to set a *Recurrence* period for the characteristic values. The available options are:

- Daily
- Weekly
- Monthly
- Yearly
- None

If the values are to be supplied locally via a table (see below for the descriptions of different data sources), the user will also see a field called *Resolution*. The resolution that can be selected depends upon the recurrence period.

A special case is the selection of yearly recurrence with a resolution of “Seasons”. If this option is selected, a new *Seasons* page name appears on the left-hand side of the dialog. On this page, the user is able to configure the required seasons. Once this is done, the values can be entered on the *Curve* page.

There are four options for defining the data source of values used in a time characteristic: *Table*, *File*, *Result File* and *Database*. The *Table* data is stored internally within *PowerFactory*. The *File* data is stored externally to *PowerFactory* in a *Comma Separated Values (\*.csv)* file or *User Defined Text File*. For the *Result File* option, the characteristic is created from data in an existing results file, and the *Database* option enables the data to be taken from a database outside the *PowerFactory* database, for which access information has to be provided by the user, as described below.

### 18.2.6.2 Time characteristic using internal table

To define a project time characteristic for a parameter using a table:

- In the edit dialog of the target network component right-click on the desired parameter.
- Select *Add project characteristic* → *Time Characteristic ...*
- Click the *New Object* button 
- The edit dialog of the Time Characteristic will be displayed. Define the parameter name and select ‘Data Source’ *Table*.
- Select the desired ‘Recurrence’ and the ‘Resolution’.
- Define the ‘Usage’ and ‘Approximation’ and enter the characteristic values in the table.
- Press **OK**.

### 18.2.6.3 Time characteristic using an external file

To define a project time characteristic for a parameter using an external file:

- In the edit dialog of the target network component right-click on the desired parameter.
- Select *Add project characteristic* → *Time Characteristic* ...
- Click the *New Object* button 
- The edit dialog of the Time Characteristic will be displayed. Define the parameter name and select 'Data Source' *File*.
- Select the desired 'Filename' and file 'Format'.
- Define the file configuration including:
  - The 'Time Column' and 'Data Column' and 'Column separator' and 'Decimal separator'.
  - The 'Unit' of time if is not a time stamped data file and the recurrence is *Daily*.
  - If the file has 'Time Stamped Data' format, select 'Timestamps are given in UTC' if this is the case.
- Define the 'Usage' and 'Approximation'.
- Press **OK**.

#### 18.2.6.4 Time characteristic using a Result File

To define a project time characteristic for a parameter using a result file:

- In the edit dialog of the target network component right-click on the parameter for which the characteristic is to be defined.
- Select *Add project characteristic* → *Time Characteristic* ...
- Click the *New Object* button 
- The edit dialog of the Time Characteristic will be displayed. Define the parameter name and select 'Data Source' *Result File*.
- Use the drop-down arrow to select the Result File.
- Select the element whose results are to be used.
- Select the relevant parameter.
- Define the 'Usage' and 'Approximation'.
- Press **OK**.

#### 18.2.6.5 Time characteristic using a Database

When defining characteristics which use data from an external database, the user has to set up an ODBC Database Configuration object, or select an existing Database Configuration. These database configurations can exist within the project, in a folder, or in a configuration area, for example.

If defining a new configuration, this can be done from the *ChaTime* dialog, using the drop-down arrow next to *Database*. This is described below, as part of the process of setting up the characteristic.

### Compatible database data types

It is important to ensure that the data types used in the database are suitable for use with the time characteristic. These are the supported types:

- Supported types for the Timestamp column:
  - DATETIME
  - TIMESTAMP, TIMESTAMP WITH TIMEZONE
  - BIGINT or INTEGER or INT (number of seconds that have elapsed since 00:00:00 UTC on 1 January 1970)
- Supported types for ID columns:
  - VARCHAR
- Supported types for Value columns:
  - REAL
  - DECIMAL(precision, scale) e.g. DECIMAL(14,3)
  - NUMERIC(precision, scale) e.g. NUMERIC(14,3)
  - DOUBLE

### Setting up the time characteristic

To define a project time characteristic for a parameter using an external database:

- In the edit dialog of the target network component right-click on the parameter for which the characteristic is to be created.
- Select *Add project characteristic* → *Time Characteristic* . . .
- Click the *New Object* button 
- The edit dialog of the Time Characteristic will be displayed.
- Define the parameter name and select “Data Source” *Database*.
- Using the drop-down arrow next to *Database* in the Database panel of the dialog, the database configuration details are either selected or a new configuration created like this:
  - Navigate to the chosen location for the new object.
  - Click the *New Object* button 
  - Select the database system and the ODBC driver. Oracle, PostgreSQL and SQL Server are offered explicitly as database system options; a fourth option *Generic* enables the user to select any available ODBC driver, but in this case the user must determine which of the remaining fields needs to be populated.
  - Enter the server name.
  - Enter the access details (Username and password) for the database system.
  - Enter the database name.
  - Enter database schema (optional): some database systems support several schemas in one database (e.g. SQL Server or PostgreSQL). If a schema is set, *PowerFactory* searches the table only in that schema.
- Four *Table modes* are offered. These allow for different arrangements of characteristics data with the database, as illustrated in detail below - see the section entitled **Worked examples for each table mode**.
  - “Value column” is used if each element has its own individual table in the database, or individual columns within a single table.
  - “ID column” is used if there is a common table where data for more than one characteristic is stored, and IDs are used within the table to identify the data for each element, rather than having separate columns for each element.

- “Day-based without ID column” is used if each row in the database table for an element contains data for one day, with a separate column for each point in time.
- “Day-based with ID column” is used if each row in the database table contains data for one day, with a separate column for each point in time, but the table contains data for multiple elements, which are identified by IDs.
- *Default value*: It can often be the case that values are missing in database data, and these values can be covered within the day by using extrapolation of the existing data. However, for days where no data at all exists, these null values will default to zero. This *Default value* field can be used to specify a default value to be used instead of zero for replacing these missing values.
- The *Time offset*: The values in the time column are normally assumed to be UTC times, which are therefore independent of the local time zone or daylight saving regime. If the user wishes to enter local times instead, the *Time offset* is used to make the necessary adjustment to these values to convert them to UTC times. For example, if the values in the time column are in Central European Time (CET i.e. UTC+01:00), the *Time offset* should be “-1.0”.
- Select the relevant parameter.
- Define the ‘Usage’ and ‘Approximation’.
- Press **OK**.

---

**Note:** When setting up the database, users should be aware that the read performance for large tables can be vastly improved by ensuring that there are appropriate table indexes. The *value column-based* table should have an index on the *Timestamp* column (e.g. with “CREATE INDEX IndexName ON Table (Timestamp)"); the *ID-based* table should have a combined index on both the *Timestamp* and the *ID* column (e.g. with “CREATE INDEX IndexName ON Table(Timestamp, ID)”).

---

**Note:** When defining characteristics with time stamped data, be aware of **daylight savings**.

- In summer when the clocks are put forward 1 hour, the time stamps from 02:00 AM to 02:59 AM do not occur this day. *PowerFactory* expects that these time stamps do not occur in the time characteristic either. Time stamps within this time range will be ignored.
  - In winter when the clocks are put back 1 hour, the time stamps from 02:00 AM to 02:59 AM occur twice this day. *PowerFactory* expects that these time stamps only occur once in the time characteristic. Multiple definitions for the same time stamp will be ignored.
- 

### Worked examples for each table mode

This section shows how the settings in the *ChaTime* dialog should be configured, according to how the data is stored in the database that is serving as the data source. It is assumed that characteristics are being created for active and reactive power for two load objects, Load01 and Load02.

Figure 18.2.3 shows some of the load data for the two loads.

Load01			Load02		
Timestamp	P	Q	Timestamp	P	Q
2025-01-01 00:00:00	31.28	11.85	2025-01-01 00:00:00	53.46	21.32
2025-01-01 00:15:00	31.14	11.60	2025-01-01 00:15:00	52.89	20.75
2025-01-01 00:30:00	30.65	11.03	2025-01-01 00:30:00	52.21	20.17

Figure 18.2.3: Sample data for two loads

For each table mode, the illustrations below show expected database table arrangements and the corresponding settings in the *ChaTime* dialogs. Note that the modes can accommodate different options for data storage within the database; the *ChaTime* parameters just need to be set accordingly.

The diagram illustrates two separate tables, **Table LOAD01** and **Table LOAD02**, each containing timestamped data for two elements, P and Q. To the right of each table are two *ChaTime* configurations:

- ChaTime for Load01 P**: Table = LOAD01, Time Column = Timestamp, Value Column = P
- ChaTime for Load01 Q**: Table = LOAD01, Time Column = Timestamp, Value Column = Q
- ChaTime for Load02 P**: Table = LOAD02, Time Column = Timestamp, Value Column = P
- ChaTime for Load02 Q**: Table = LOAD02, Time Column = Timestamp, Value Column = Q

Table LOAD01		
Timestamp	P	Q
2025-01-01 00:00:00	31.28	11.85
2025-01-01 00:15:00	31.14	11.60
2025-01-01 00:30:00	30.65	11.03
...	...	...

Table LOAD02		
Timestamp	P	Q
2025-01-01 00:00:00	53.46	21.32
2025-01-01 00:15:00	52.89	20.75
2025-01-01 00:30:00	52.21	20.17
...	...	...

Figure 18.2.4: “Value column” mode; Example A - separate tables for each element

The diagram illustrates a single central table, **Table LOADS**, which contains timestamped data for four elements: Load01P, Load01Q, Load02P, and Load02Q. To the right of the table are four *ChaTime* configurations:

- ChaTime for Load01 P**: Table = LOADS, Time Column = Timestamp, Value Column = Load01P
- ChaTime for Load01 Q**: Table = LOADS, Time Column = Timestamp, Value Column = Load01Q
- ChaTime for Load02 P**: Table = LOADS, Time Column = Timestamp, Value Column = Load02P
- ChaTime for Load02 Q**: Table = LOADS, Time Column = Timestamp, Value Column = Load02Q

Table LOADS				
Timestamp	Load01P	Load01Q	Load02P	Load02Q
2025-01-01 00:00:00	31.28	11.85	53.46	21.32
2025-01-01 00:15:00	31.14	11.60	52.89	20.75
2025-01-01 00:30:00	30.65	11.03	52.21	20.17
...	...	...	...	...

Figure 18.2.5: “Value column” mode; Example B - one table for multiple elements

Table LOADS			
Timestamp	ID	P	Q
2025-01-01 00:00:00	Load01	31.28	11.85
2025-01-01 00:15:00	Load01	31.14	11.60
2025-01-01 00:30:00	Load01	30.65	11.03
...	...	...	...
2025-01-01 00:00:00	Load02	53.46	21.32
2025-01-01 00:15:00	Load02	52.89	20.75
2025-01-01 00:30:00	Load02	52.21	20.17
...	...	...	...

**ChaTime for Load01 P**  
 Table = LOADS  
 Time Column = Timestamp  
 ID Column = ID  
 ID Value = Load01  
 Value Column = P

**ChaTime for Load01 Q**  
 Table = LOADS  
 Time Column = Timestamp  
 ID Column = ID  
 ID Value = Load01  
 Value Column = Q

**ChaTime for Load02 P**  
 Table = LOADS  
 Time Column = Timestamp  
 ID Column = ID  
 ID Value = Load02  
 Value Column = P

**ChaTime for Load02 Q**  
 Table = LOADS  
 Time Column = Timestamp  
 ID Column = ID  
 ID Value = Load02  
 Value Column = Q

Figure 18.2.6: “ID column” mode; Example A - the ID used to identify the *element*

Table LOADS			ChaTime for Load01 P
Timestamp	ID	Value	Table = LOADS Time Column = Timestamp ID Column = ID ID Value = Load01.P Value Column = Value
2025-01-01 00:00:00	Load01.P	31.28	
2025-01-01 00:15:00	Load01.P	31.14	
2025-01-01 00:30:00	Load01.P	30.65	
...	...	...	
2025-01-01 00:00:00	Load01.Q	11.85	ChaTime for Load01 Q
2025-01-01 00:15:00	Load01.Q	11.60	Table = LOADS Time Column = Timestamp ID Column = ID ID Value = Load01.Q Value Column = Value
2025-01-01 00:30:00	Load01.Q	11.03	
...	...	...	
2025-01-01 00:00:00	Load02.P	53.46	ChaTime for Load02 P
2025-01-01 00:15:00	Load02.P	52.89	Table = LOADS Time Column = Timestamp ID Column = ID ID Value = Load02.P Value Column = Value
2025-01-01 00:30:00	Load02.P	52.21	
...	...	...	
2025-01-01 00:00:00	Load02.Q	21.32	ChaTime for Load02 Q
2025-01-01 00:15:00	Load02.Q	20.75	Table = LOADS Time Column = Timestamp ID Column = ID ID Value = Load02.Q Value Column = Value
2025-01-01 00:30:00	Load02.Q	20.17	
...	...	...	

Figure 18.2.7: “ID column” mode; Example B - the ID used to identify *element:attribute*

Table LOAD01P					ChaTime for Load01 P
Timestamp	C0000	C0015	C0030	...	Table = LOAD01P Time Column = Timestamp Value Column = C0000,C0015,C0030...
2025-01-01 00:00:00	31.28	31.14	30.65	...	
2025-01-02 00:00:00	...	...	...	...	
Table LOAD01Q					ChaTime for Load01 Q
Timestamp	C0000	C0015	C0030	...	Table = LOAD01Q Time Column = Timestamp Value Column = C0000,C0015,C0030...
2025-01-01 00:00:00	11.85	11.60	11.03	...	
2025-01-02 00:00:00	...	...	...	...	
Table LOAD02P					ChaTime for Load02 P
Timestamp	C0000	C0015	C0030	...	Table = LOAD02P Time Column = Timestamp Value Column = C0000,C0015,C0030...
2025-01-01 00:00:00	53.46	52.89	52.21	...	
2025-01-02 00:00:00	...	...	...	...	
Table LOAD02Q					ChaTime for Load02 Q
Timestamp	C0000	C0015	C0030	...	Table = LOAD02Q Time Column = Timestamp Value Column = C0000,C0015,C0030...
2025-01-01 00:00:00	21.32	20.75	20.17	...	
2025-01-02 00:00:00	...	...	...	...	

Figure 18.2.8: “Day-based without ID column” mode

Table LOADS					
Timestamp	ID	C0000	C0015	C0030	...
2025-01-01 00:00:00	Load01.P	31.28	31.14	30.65	...
2025-01-02 00:00:00	Load01.P	...	...	...	...
...	...	...	...	...	...
2025-01-01 00:00:00	Load01.Q	11.85	11.60	11.03	...
2025-01-02 00:00:00	Load01.Q	...	...	...	...
...	...	...	...	...	...
2025-01-01 00:00:00	Load02.P	53.46	52.89	52.21	...
2025-01-02 00:00:00	Load02.P	...	...	...	...
...	...	...	...	...	...
2025-01-01 00:00:00	Load02.Q	21.32	20.75	20.17	...
2025-01-02 00:00:00	Load02.Q	...	...	...	...

**ChaTime for Load01 P**  
 Table = LOADS  
 Time Column = Timestamp  
 ID Column = ID  
 ID Value = Load01.P  
 Value Column = C0000,C0015,C0030...

**ChaTime for Load01 Q**  
 Table = LOADS  
 Time Column = Timestamp  
 ID Column = ID  
 ID Value = Load01.Q  
 Value Column = C0000,C0015,C0030...

**ChaTime for Load02 P**  
 Table = LOADS  
 Time Column = Timestamp  
 ID Column = ID  
 ID Value = Load02.P  
 Value Column = C0000,C0015,C0030...

**ChaTime for Load02 Q**  
 Table = LOADS  
 Time Column = Timestamp  
 ID Column = ID  
 ID Value = Load02.Q  
 Value Column = C0000,C0015,C0030...

Figure 18.2.9: “Day-based with ID column” mode

**Note:** For day-based data, the values are assigned according to the number of value columns, e.g. hourly if there are 24 columns, quarter-hourly if there are 96 columns.

### 18.2.6.6 Discrete Time Characteristics

The discrete time characteristic (*ChaDisctime*) is provided for backward compatibility with previous versions of *PowerFactory*. It is more restricted than the time characteristic and hence its use is limited since *PowerFactory* version 15.1. Similar to the time characteristic, the discrete time characteristic uses an internally defined series of time scales that are convenient to use to define the characteristic. The user simply selects a scale (e.g. day of the week) and enters the corresponding values.

### 18.2.7 Profile Characteristics

General background on characteristics and their properties is provided in Section 18.2.

The profile characteristic is used to select a time characteristic (*ChaTime*) corresponding to individual days or group of days and each season. The profile characteristic can also be used to select a time characteristic for certain holiday days.

To define a project profile characteristic for a parameter:

- In the edit dialog of the target network component right-click on the desired parameter.
- Select *Add project characteristic* → *Profile Characteristic ...*
- Click the *New Object* button 

- The edit dialog of the Profile Characteristic will be displayed.
- Select the 'Seasons' page and define one or more seasons with a 'Description', 'Start Day', 'Start Month', 'End Day' and 'End Month'. Note that Seasons can not overlap with each other.
- Select the 'Groups of Days' page and define grouping for each day and holiday.
- Select the 'Holidays' page and define one or more holidays with a 'Description', 'Day', 'Month', if it is 'Yearly' or select a holiday 'Year'.
- Select the 'General' page, right-click and Select *Select Element/Type...* or double-click on each relevant cell and select or create a time characteristics for each group of days, holiday and season.
- Press **OK**.

#### 18.2.7.1 Yearly Growth Characteristic

In addition to seasonal characteristic variation, a yearly growth characteristic can also be defined. A yearly growth characteristic is defined using a time characteristic (*ChaTime*) with a recurrence value of "None", for the specified years.

#### 18.2.8 Scaling Factor

General background on characteristics is provided in Section [18.2](#).

Scaling factors are used when a parameter should be multiplied by a certain value or percentage. For example, a scaling factor could be used to multiply the *Active Power* value of one or more static generators by 0.5. If a parameter is assigned several scaling factors, it will be multiplied by their product.

To define a project scaling factor for a parameter:

- In the edit dialog of the target network component right-click on the desired parameter (e.g. 'Active Power').
- Select *Add project characteristic* → *Scaling Factor...*
- Click the *New Object* button 
- The edit dialog will be displayed. Set the value of the factor. The associated trigger is automatically created in the current study case.
- Define the 'Usage' "relative" or "relative in %".
- Press **OK**.

#### 18.2.9 Linear Functions

General background on characteristics and their properties is provided in Section [18.2](#).

Linear Functions are used when a parameter should vary according to a mathematical relationship, with reference to a scale value "x". For example, a linear function may reference a *Scalar and Trigger* (*TriVal*) with a *Unit* of 'Temperature'. Then, if the temperature is set to, say, 15 deg, the parameter that this characteristic is applied to will thus be multiplied by the value of the linear function  $2 \cdot 15 + 3 = 33$ .

To define a project linear function for a parameter:

- In the edit dialog of the target network component right-click on the desired parameter (e.g. 'Active Power').

- Select *Add project characteristic* → *Linear Function...*
- Click the *New Object* button 
- The edit dialog will be displayed. Click 'Select' from the drop down menu next to 'Scale' and select an existing scale and press **OK**, or create a new scale:
  - Click on the *New Object* button to create a *Scalar and Trigger (TriVal)* and set the desired units of the scale. The associated trigger is automatically created in the current study case.
  - Press **OK**.
- Define the 'Usage' and enter the parameters 'A' and 'b' of the linear function  $A \cdot x + b$ .
- Press **OK**.

## 18.2.10 Vector Characteristics

Vector Characteristics may be defined with reference to *Discrete Scales*, *Continuous Scales*, *Frequency Scales*, and *Time Scales*.

### 18.2.10.1 Vector Characteristics with Discrete Scales (TriDisc)

General background on characteristics and their properties is provided in Section [18.2](#).

A discrete parameter characteristic is used to set the value of a parameter according to discrete cases set by the trigger of a discrete scale. A discrete scale is a list of cases, each defined by a short text description. The current value is shown in the characteristic dialog in red, according to the case that is currently active.

To define a new project discrete parameter characteristic:

- In the edit dialog of the target network component right-click on the desired parameter.
- Select *Add project characteristic* → *One Dimension - Vector...*
- Click the *New Object* button 
- The edit dialog of the one dimension vector characteristic (generic class for one dimensional characteristics) will be displayed. Click 'Select' from the drop down menu next to 'Scale' and select an existing scale and press **OK**, or create a new scale as follows:
  - Make sure that the scales library is selected (*Operational Library* → *Characteristics* → *Scale*).
  - Click on the *New Object* button and select *Discrete Scale and Trigger (TriDisc)*.
  - Write the name of the scale cases (one case per line).
  - Press **OK**.
- Define the *Usage* and *Approximation* and enter the characteristic values.
- Press **OK**.

The diagram page for the discrete characteristic shows a bar graph for the available cases. The bar for the case that is currently active (set by the trigger) is shown in black.

### 18.2.10.2 Vector Characteristics with Continuous Scales (TriCont)

General background on characteristics and their properties is provided in Section [18.2](#).

A continuous parameter characteristic is used to set the value of a parameter ('Y' values) according to the 'X' values set in the continuous scale.

To define a new project continuous parameter characteristic:

- In the edit dialog of the target network component right-click on the desired parameter.
- Select *Add project characteristic* → *One Dimension - Vector...*
- Click the *New Object* button 
- The edit dialog of the one dimension vector characteristic (generic class for one dimensional characteristics) will be displayed. Click 'Select' from the drop down menu next to 'Scale' and select an existing scale and press **OK**, or create a new scale:
  - Click on the *New Object* button and select *Continuous Scale and Trigger (TriCont)*.
  - Enter the unit of the 'X' values.
  - Append the required number of rows (right-click on the first row of the Scale table and select *Append n rows*) and enter the 'X' values.
  - Press **OK**.
- Define the 'Usage', enter the characteristic 'Y' values, and define the 'Approximation' function.
- Press OK.

#### 18.2.10.3 Vector Characteristics with Frequency Scales (TriFreq)

General background on characteristics and their properties is provided in Section [18.2](#).

A frequency characteristic is a continuous characteristic with a scale defined by frequency values in Hz. The definition procedure is similar to that of the continuous characteristics, although the Frequency Scale (*TriFreq*) is selected.

#### 18.2.10.4 Vector Characteristics with Time Scales (TriTime)

General background on characteristics and their properties is provided in Section [18.2](#).

Time parameter characteristics are continuous characteristics using time scales. A time scale is a special kind of continuous scale that uses the global time trigger of the active study case. The unit of the time trigger is always a unit of time but may range from seconds to years. This means that changing the unit from minutes to hours, for instance, will stretch the scale 60-fold. The units seconds, minutes, and hour of the day are respectively, the second, minute and hour of normal daytime. A Time Scale may be used, for example, to enter four equidistant hours in a year (1095, 3285, 5475, and 7665). The Time Scales offer a range of different units such as hour of day, week of the year, month of year, year and additionally, to facilitate the use of time series data that cover days when daylight saving clock adjustments occur, the following options:

- **UTC time:** the Scale Values are specified as UTC timestamps in a prescribed format. These will be converted into local time automatically when they are used.
- **Unix time:** the Scale Values are specified as integers that are Unix times. Unix time is defined as the number of non-leap seconds which have passed since 00:00:00 UTC on Thursday, 1 January 1970.

The definition procedure is similar to that of the continuous characteristics, although the Time Scale (*TriTime*) scale is selected.

#### 18.2.11 Matrix Parameter Characteristics

General background on characteristics and their properties is provided in Section [18.2](#).

When defining a matrix parameter characteristic, two scales must be defined. The first scale, that for columns, must be a discrete scale. The scale for rows may be a discrete or continuous scale.

To define a new project matrix parameter characteristic:

- In the edit dialog of the target network component right-click on the desired parameter.
- Select *Add project characteristic* → *Two Dimension - Matrix...*
- Click the *New Object* button 
- The edit dialog of the matrix characteristic will be displayed. Click 'Select' from the drop down menu next to each 'Scale' and select an existing scale and press **OK**, or create a new scales. Scales can be defined as discussed in previous sections.

A column calculator can be used to calculate the column values, as a function of other columns. This is done by pressing the **Calculate...** button. Once the values have been entered and the triggers have been set, the 'Current Value' field will show the value to be used by the characteristic.

### 18.2.12 Parameter Characteristics from Files

General background on characteristics and their properties is provided in Section [18.2](#).

When a series of data is available in an external file, such as an Excel file, or tab or space separated file this data may be utilised as a characteristic if the “Parameter Characteristic from File” (*ChaVecfile* object) is used. The external file must have the scale column for the data series in column 1.

To define a new parameter characteristic from file:

- In the edit dialog of the target network component right-click on the desired parameter.
- Select *Add project characteristic* → *Characteristic from file...* or *Add global characteristic* → *Characteristic from file...*
- The corresponding library (local or global) will open and there click on the *New Object* button  to create a new *Parameter Characteristic from file* (*ChaVecfile*)
- Complete the input data fields, including:
  - Define (or select) a scale and trigger. Scales can be defined as discussed in previous sections.
  - Generally the 'Column' should be set to the default of '1'. The field is used for specialised purposes.
  - Set the 'Factor A' and 'Factor B' fields to adjust or convert the input data. The data contained in column 2 of the external file will be adjusted by  $y = ax + b$  where "x" is the data in the external file and "y" is what will be loaded into the characteristic.
  - Set the 'Usage' and 'Approximation'.
  - Once the file link has been set, press the Update button to upload the data from the external file to the characteristic.

### 18.2.13 Characteristic References

When a characteristic is defined for an objects parameter, *PowerFactory* automatically creates a characteristic reference (*ChaRef* object). The characteristic reference is stored within the *PowerFactory* database with the object. The characteristic reference acts as a pointer for the parameter to the characteristic. The characteristic reference includes the following parameters:

**Parameter** the name of the object parameter assigned to the characteristic. This field cannot be modified by the user.

**Characteristic** the characteristic which is to be applied to the parameter.

**Inactive** a check-box which can be used to disable a characteristic reference.

The ability to disable the characteristic for individual objects using the object filter and the *Inactivate* option makes data manipulation using characteristics quite flexible.

### 18.2.14 Edit Characteristic Dialog

Once a parameter has a characteristic defined, then an option to *Edit characteristic(s)* becomes visible on the parameters context menu, i.e. select parameter and *right-click* → *Edit characteristic(s)*. Once selected, the *Edit characteristics* dialog appears which lists all the characteristics referenced by the parameter. The *Edit characteristics* dialog provides a graphical representation of the characteristic and allows characteristics to be inserted, appended and deleted. The *Edit characteristics* dialog also allows modification of individual characteristics values, triggers and characteristic activation and deactivation.

**Note:** By default the value of the first active characteristic is assigned to the parameter.

### 18.2.15 Characteristics Tab in Data Filters

When viewing elements in a Data Manager or Network Model Manager, parameter characteristics information can be seen by selecting the Characteristics tab. For this tab to be visible, it must be enabled in the User Settings, on the “Functions” page. An example of a Network Model Manager showing the Characteristics tab is shown in Figure 18.2.10 (remember that the browser must be in “detail” mode to see these tabs). Note also that the data colouring indicates that characteristics are applied.

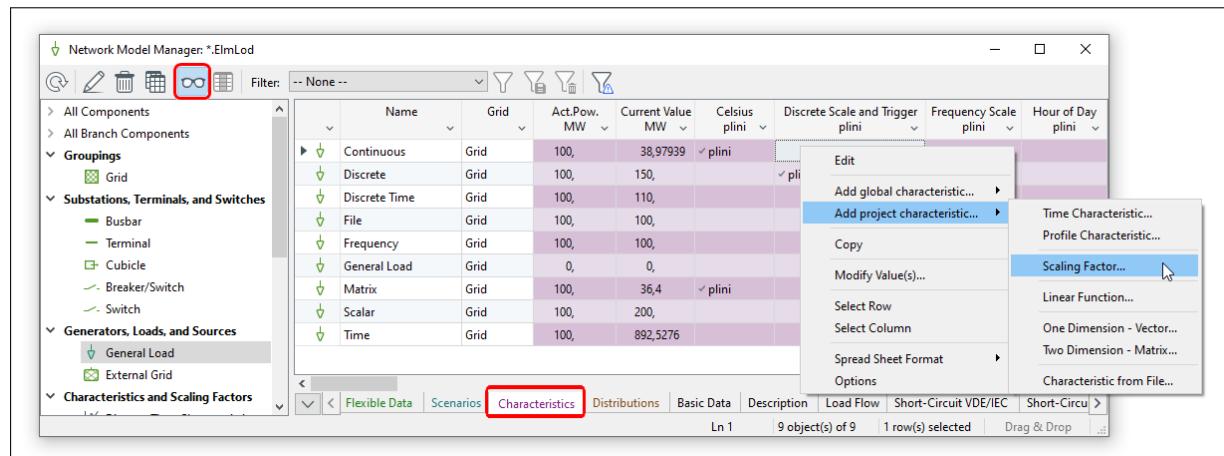


Figure 18.2.10: Network Model Manager Characteristics tab

The Characteristics tab shows all characteristics defined for the displayed objects, together with the original value and the current value as determined by the characteristic. In the example, various scales are applied to modify the active power from 100 MW to the “Current Value”. The current values will be used in all calculations. New characteristics for individual or multiple elements can be defined by selecting the relevant fields and doing *right-click* → *Characteristic/Distribution*→ *Add project characteristic...* . . .

The Characteristics tab will only show a particular characteristic column when at least one of the objects has that characteristic defined for a parameter. It is thus necessary to define a characteristic for one object prior to using the browser, when the user would like to assign characteristics, for the same

parameter, for a range of other objects. To define a Project “High-Low” loading characteristic for all loads, for instance, can thus be done by performing the following steps.

- Create a discrete scale in the grid folder.
- Create a vector characteristic using this scale in the grid folder.
- Edit one of the loads, right-click the active power field and assign the vector characteristic to the relevant parameter.
- Open a browser with all loads, activate the “detail” mode and select the Characteristics tab.
- Select the characteristic column (*right-click → Select Column*) and then right-click the selected column.
- Use the *Select Project Characteristic...* option and select the vector characteristic.

### 18.2.16 Example Application of Characteristics

Consider the following example, where the operating point of a generator should be easily modified by the user to predefined values within the capability limits of the machine.

Firstly, the *Active Power* of the synchronous generator is set to the maximum capability of 150 MW. Then, a vector characteristic is added to the *Active Power* parameter. To create a new *Project Vector Characteristic*, right-click on the *Active Power* parameter (pgini) and select *Add project characteristic → One Dimension - Vector...*. Click on the *New Object* icon and define a characteristic called “Active Power” in the *ChaVec* dialog.

A new discrete scale is required. To create the scale, click on the arrow next to *Scale* and select *Select....*. Click on the *New Object* icon and create a new *Discrete Scale and Trigger (TriDisc)*. The *Discrete Scale and Trigger* is named “Output Level”, with three cases as shown in Figure 18.2.11.

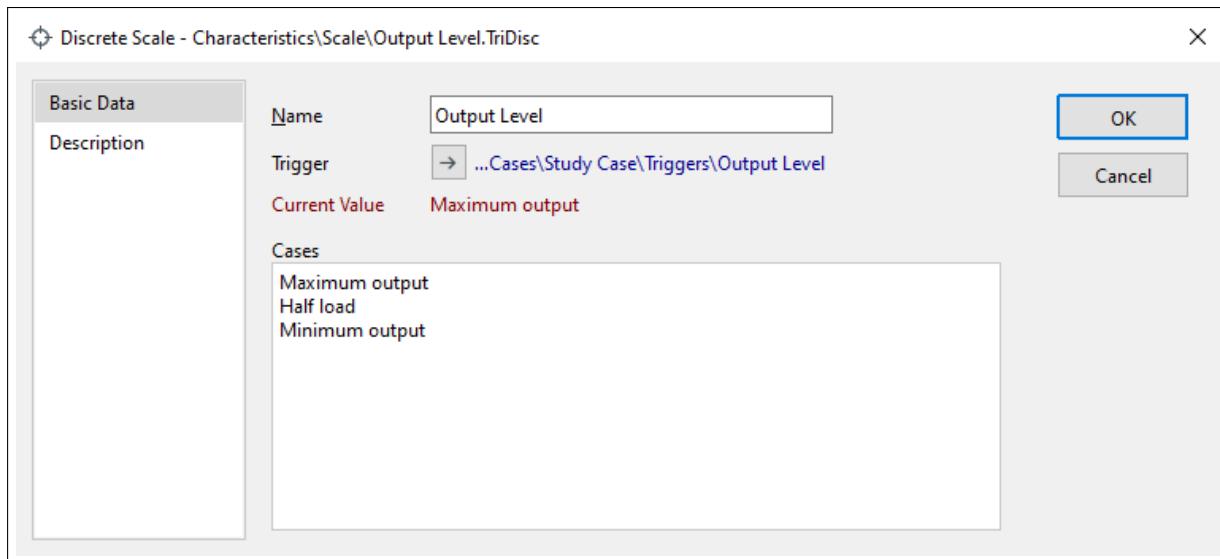


Figure 18.2.11: Active Power Discrete Scale and Trigger

Click on **OK** to return to the *Vector Characteristic*. Define the values for the different loading scenarios. Values are entered in %, and thus *Usage* is set to ‘relative in %’. Figure 18.2.12 shows the resultant vector characteristic, including a reference to the Scale “Output Level” and the current parameter value.

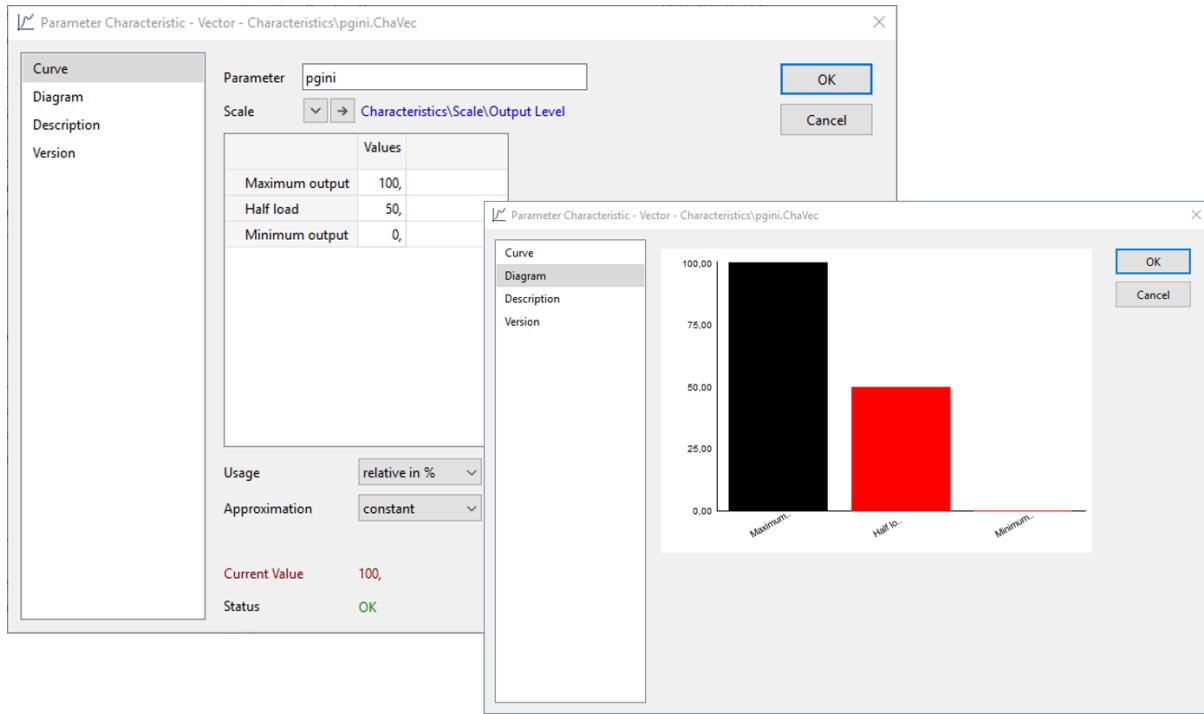


Figure 18.2.12: Active Power Parameter Characteristic

Next, a matrix characteristic is added to the *Reactive Power* parameter of the generator in a similar fashion to the *Active Power* characteristic. A new discrete scale named “Operating Region” is created (for the Columns) and the three operating regions “Underexcited”, “Unity PF” and “Overexcited” are defined.

The scale “Operating Region” is linked to the *Scale for Columns*, and the previously defined scale “Output Level” is selected for the *Scale for Rows*. Absolute Mvar values are entered in the Matrix Characteristic as shown in Figure 18.2.13.

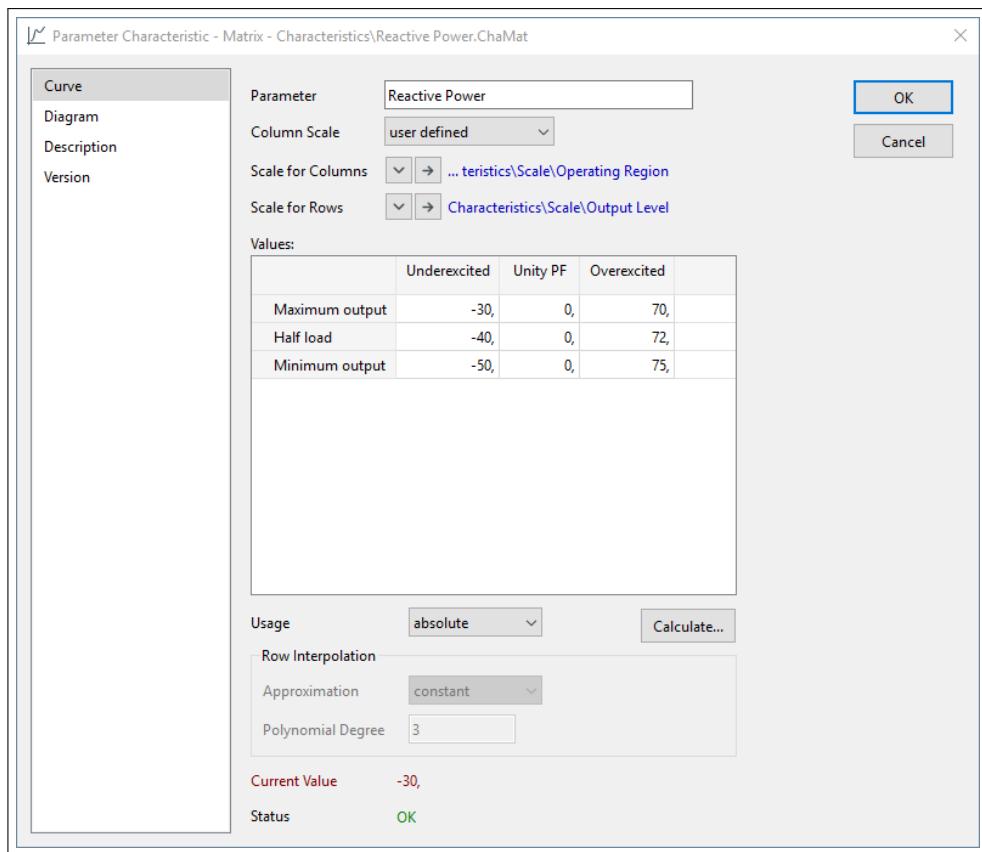


Figure 18.2.13: Reactive Power Matrix Characteristic

Now that the characteristics and triggers are defined, the “Operating Region” and “Output Level” triggers can be used to quickly modify the operating point of the generator (see Figure 18.2.14).

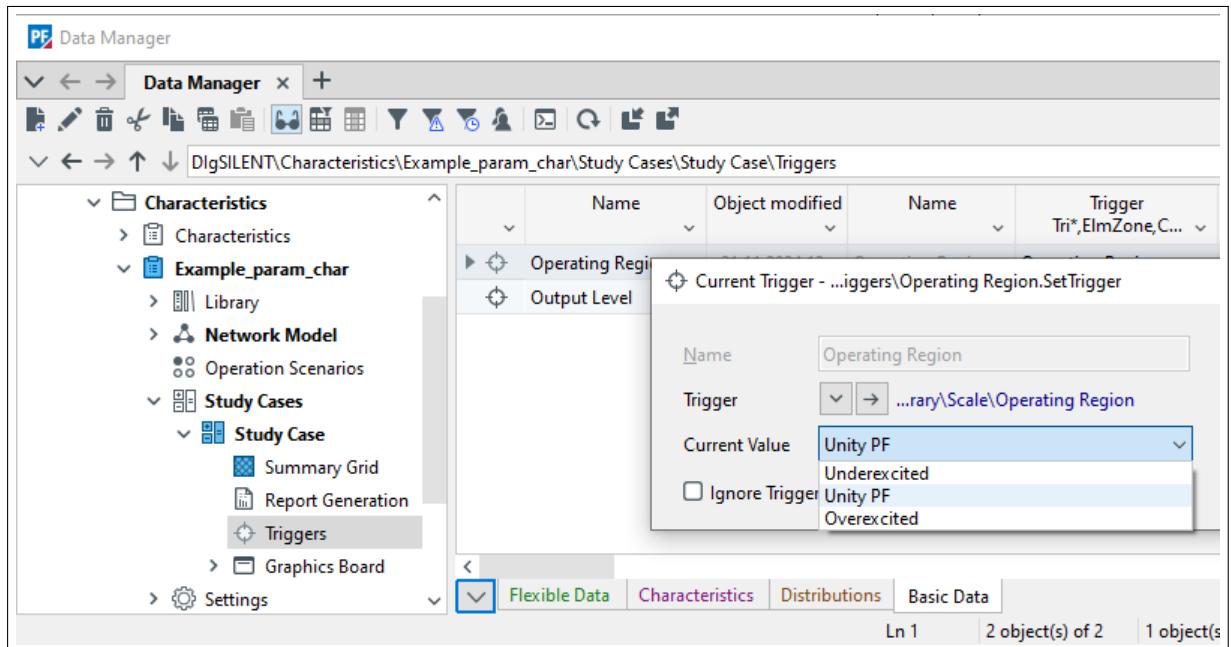


Figure 18.2.14: Setting of Discrete Triggers

## 18.3 Load States

This section describes Load States, as used in Reliability and Optimal Capacitor Placement calculations.

### 18.3.1 Creating Load States

Pre-requisites:

Prior to creating load states, a time-based parameter characteristics must be defined for at least one load in the network model. See Time Characteristics (*ChaTime*) in Section 18.2.6 and Vector Characteristics with Time Scales (*TriTime*) in Section 18.2.10 for more information on parameter characteristics, as well as the example later in this section.

Follow these steps to create the load states:

1. For calculation of load states:
  - (Reliability) click the 'Create Load States' icon ( ) on the reliability toolbar and select 'Load States'. Optionally inspect or alter the settings of the Reliability Calculation and Load Flow commands.  
**Note:** Optionally choose the option to "Consider Generators" to consider time varying power feed-in of generators. If selected, the generator power state will additionally be contained within the clusters. Please note, that the Load and Generator States can only be applied for Reliability assessment.
  - (Optimal Capacitor Placement) Click on 'Load Characteristics' page of the Optimal Capacitor Placement command and select 'Create Load States'.
2. Enter the time period for calculation of load states:
  - (Reliability) Enter the year.
  - (Optimal Capacitor Placement) Enter Start Time and End Time. The time period is inclusive of the start time but exclusive of the end time.
3. Enter the Accuracy. The lower accuracy percentage, the more load states are generated.
4. Optional: Limit the number of load states to a user-defined value. If the total number of calculated load states exceeds this parameter then either the time period of the sweep or the accuracy should be reduced.
5. Optional: Change the threshold for ignoring load states with a low probability by altering the 'Minimum Probability'. If selected, states with a probability less than this parameter are excluded from the discretisation algorithm.
6. Click Execute to generate the load states.

### 18.3.2 Viewing Existing Load States

After you have generated the load states as described above, or if you want to inspect previously generated load states follow these steps:

1. Using the Data Manager, select the 'Reliability Assessment' or 'Optimal Capacitor Placement' command within the active Study Case.
2. Use the filter ( ) (in the Data Manager window) to select the 'load states' object. There should now be created load states visible in the right panel of the Data Manager.
3. Locate the 'load states' object and double-click to view the load states.

### 18.3.3 Load State Object Properties

The load states object properties are as follows:

#### 18.3.3.1 Basic Data

- **Year:** The year used to create the load states.
- **Number of loads:** Number of loads and generators considered in the load cluster object.
- **Number of states:** This equals the number of columns in the “Clusters” table.
- **Loads:** Table containing each load considered by the load states creation algorithm and their peak demand.
- **Clusters:** Table containing all load clusters. The first row in the table contains the probability of the corresponding cluster. The remaining rows contain the power values of the loads. Every column in the table contains a complete cluster of loads with the corresponding power.

#### 18.3.3.2 Diagram Page

- **Displayed Load:** Use the selection control to change the load displayed on the plot.

The plot shows the cluster values (P and Q) for the selected load where the width of each bar represents the probability of occurrence for that cluster in the given year.

### 18.3.4 Example Load States

The example below shows how load states can be generated for a network model with four Loads (Ld1, Ld2, Ld3, and Ld4).

1. The Vector Characteristic shown in Figure 18.3.1 is applied to both Active Power and Reactive Power of load Ld4 only, with the associated Time Scale shown in Figure 18.3.2 Ld4 is initially set to 3.1 MW, 0.02 Mvar.

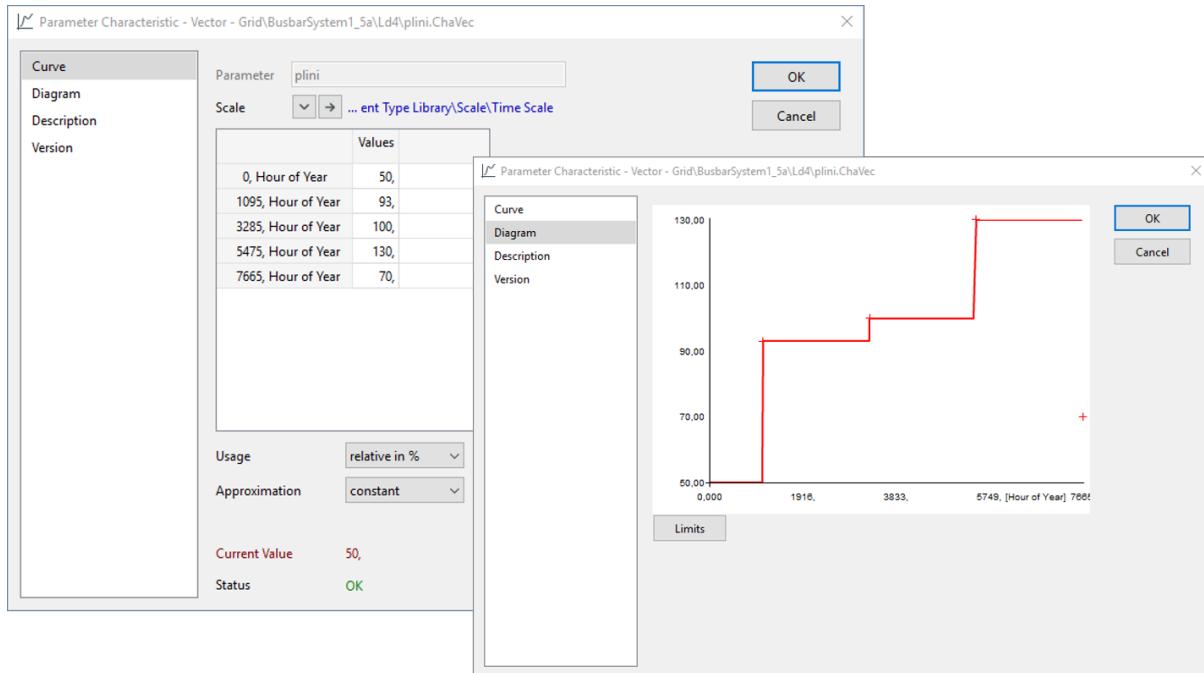


Figure 18.3.1: Load State Vector Characteristic

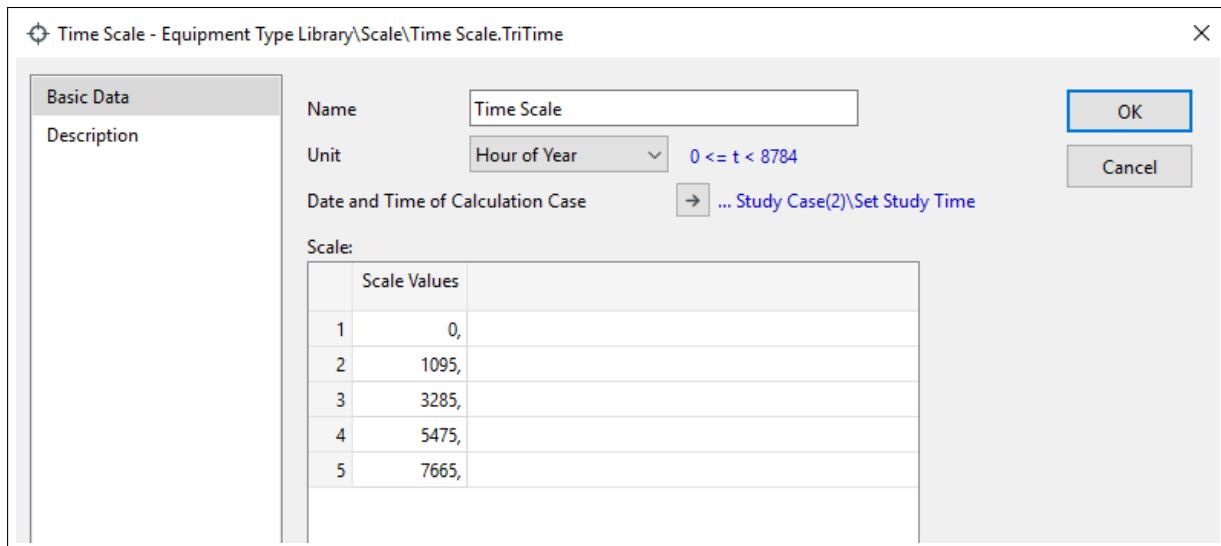


Figure 18.3.2: Time Scale for Load State Characteristic

2. Load States are generated by clicking *Create Load States* (as discussed in the preceding section).
3. *PowerFactory* calculates the resultant Load States:
  - The maximum value of each load  $L_p$  is determined for the time interval considered. In the example, Ld4 has a peak load of 4.03 MW.
  - The 'load interval size' ( $Int$ ) is determined for each load, where  $Int = L_p \cdot Acc$  and 'Acc' is the accuracy parameter entered by the user. For the example above using an accuracy of 10 %, the interval size for Active Power is 0.403 MW.
  - For each hour of the time sweep and for each load determine the Load Interval:  $LInt = Ceil(\frac{L_i}{Int})$  where  $L_i$  is the load value at hour 'i'.
  - Identify common intervals and group these as independent states.
  - Calculate the probability of each state based on its frequency of occurrence.

The independent states and their probabilities are shown in Figure 18.3.3. Load states for Ld4 vary according to the characteristic parameters, where the states from characteristic values of 93 % and 100 % have been combined due to the selection of 10 % accuracy in the calculation. Load states for Ld1, Ld2, and Ld3 do not vary, since characteristics were not entered for these loads.

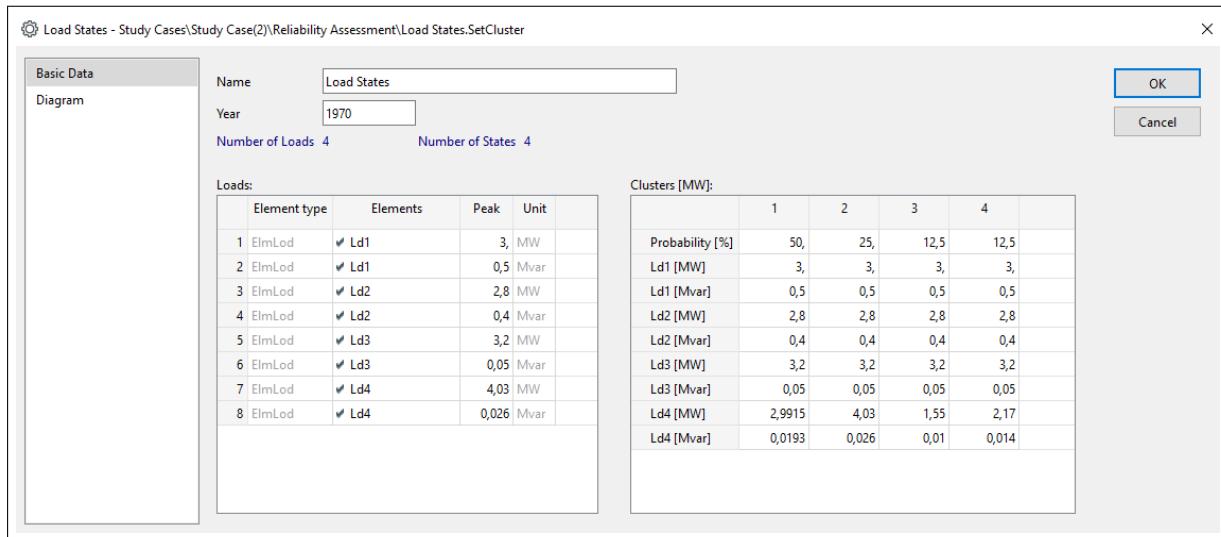


Figure 18.3.3: Load States (SetCluster) dialog box

## 18.4 Load Distribution States

This section describes how to create load distribution states, as used by the Reliability calculation.

### 18.4.1 Creating Load Distribution States

Pre-requisites:

Prior to creating load distribution states a substation/s must have been defined within the model. A distribution curve must have also been defined (accessed from the reliability page of the substation/s).

Follow these steps to create the load distribution states:

1. Click the 'Create Load States' button ( on the reliability toolbar. The load states creation dialog will appear.
2. Optional: Use the Reliability Assessment selection control to inspect or alter the settings of the Reliability Calculation command. This selection control points to the default reliability command within the active Study Case.
3. Optional: Use the Load Flow selection button to inspect and alter the settings of the load flow command. This selection control points to the default load-flow command within the active Study Case.
4. Enter the Minimum Time Step in hours (suggested to be the minimum step size on the load distribution curve).
5. Enter the Maximum Power Step (0.05 p.u. by default).
6. Optional: Force Load State at S = 1.0 p.u. so that a state is created at P = 1.0 p.u., irrespective of the load distribution curve data and step sizes entered.
7. Click Execute to generate the load distribution states.

### 18.4.2 Viewing Existing Load Distribution States

After you have generated the load states as described above, or if you want to inspect previously generated load states follow these steps:

1. Using the Data Manager, select the 'Reliability Assessment' Command within the Active Study Case.
2. Optional: Use the filter (grid icon) (in the Data Manager window) to select the 'load distribution states' object. There should now be created load distribution states visible in the right panel of the Data Manager.
3. Locate the 'load distribution states' object and double-click to view the load states.

### 18.4.3 Load Distribution State Object Properties

The distribution load states object properties are as follows:

#### 18.4.3.1 Basic Data

- **Year:** The year used to create the load states.
- **Clusters:** Table containing all substation clusters. The first row in the table contains the probability of the corresponding cluster. The remaining rows contain the power values of the substations. Every column in the table contains a complete cluster of substations with the corresponding power.
- **Number of substations:** Number of substations considered in the Distribution State object.
- **Number of states:** This equals the number of columns in the Distribution State table.

#### 18.4.3.2 Diagram Page

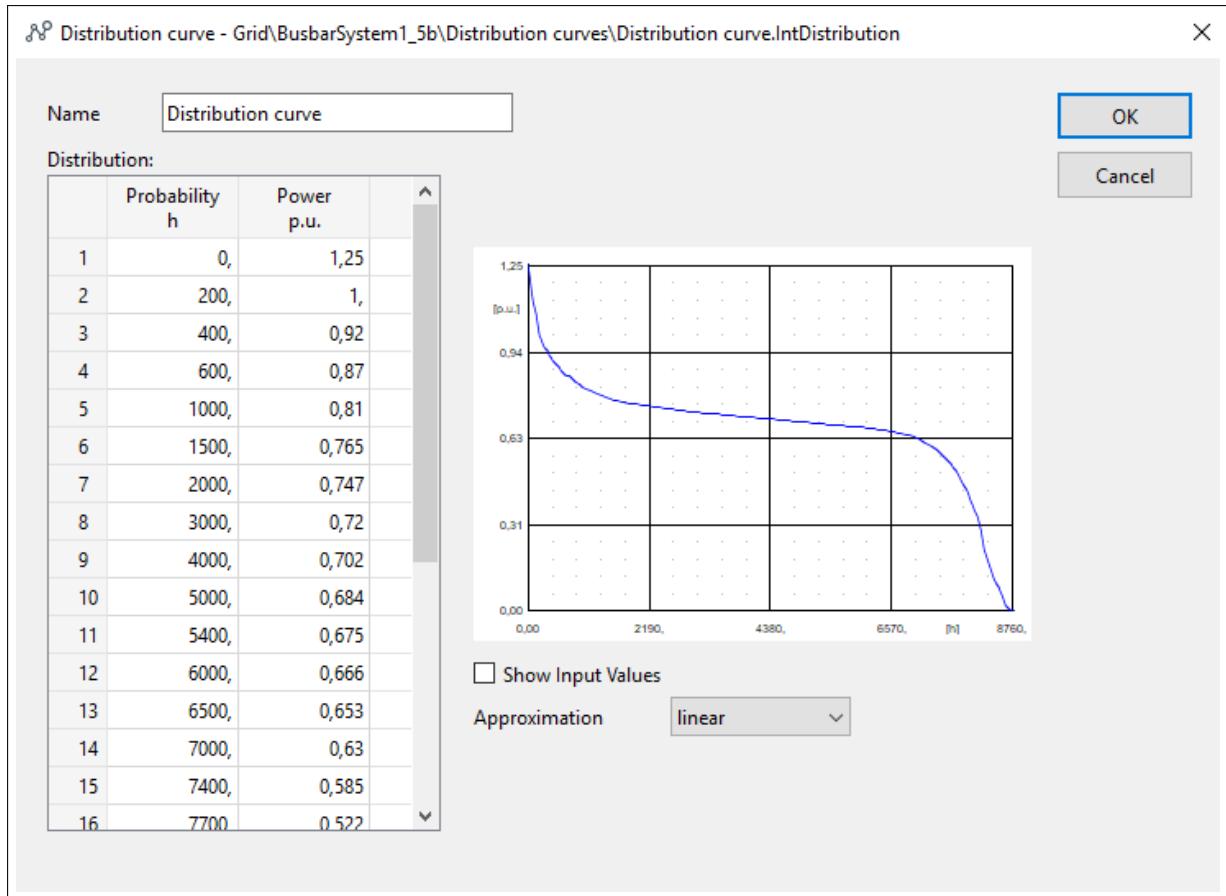
**Displayed Station:** Use the selection control to change the load displayed on the plot

The plot shows the cluster values (Apparent power in p.u. with reference to the substation load) for the selected substation where the width of each bar represents the probability of occurrence for that cluster.

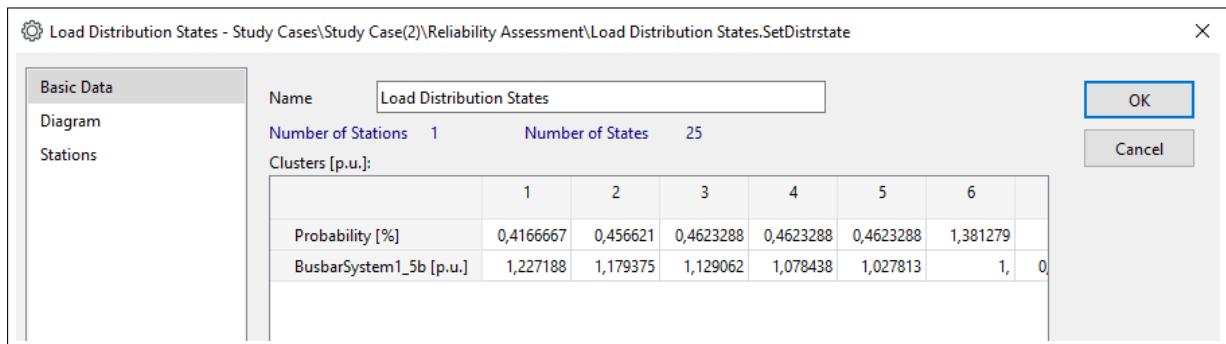
### 18.4.4 Example Load Distribution States

In this example, a Load Distribution Curve is entered for a substation.

1. The Load Distribution Curve shown in Figure 18.4.1 is entered for the substation (Apparent power in p.u. of substation load).

Figure 18.4.1: Substation Load Distribution Curve (*IntDistribution*)

2. Load States are generated by clicking *Create Load Distribution States* (as discussed in the preceding section).
3. The resultant Load Distribution States are shown in Figure 18.4.2. 'Force Load State at S = 1.0 p.u.' has not been selected in this instance.

Figure 18.4.2: Load Distribution States (*SetDistrstate*)

## 18.5 Tariffs

This section describes the definition of Time Tariffs (as used in Reliability calculations), and Energy Tariffs (as used in Reliability calculations and Optimal RCS Placement calculations, and Techno-Economical calculations).

### 18.5.1 Defining Time Tariffs

A time tariff characteristic can be defined by taking the following steps:

1. Choose the *Select* option from the 'Tariff' selection control on the *Reliability* page of the load element. A Data Manager browser will appear with the *Operational Library* → *Tariff* folder selected.

**Note:** Optional: If you have previously defined a 'Tariff' characteristic and want to re-use it, you can select it now. Press **OK** to return to the load element to reliability page.

2. Create a time tariff object by pressing the *New Object* button from the data browser toolbar.
3. Select *Time Tariff* and press **OK**. A 'Time Tariff' dialog box will appear.
4. Select the unit of the interruption cost function by choosing from the following options:  
**\$/kW** Cost per interrupted power in kW, OR  
**\$/customer** Cost per interrupted customer, OR  
**\$** Absolute cost.
5. Enter values for the Time Tariff (right-click and 'Append rows' as required).
6. Press **OK** to return to the load element reliability page.

#### 18.5.1.1 Example Time Tariff

An example Time Tariff characteristic is shown in Figure 18.5.1. In this example, 'Approximation' is set to 'constant', i.e. no interpolation between data points, and 'Unit' is set to \$. An interruption to a load for a duration of 200 minutes would lead to a cost of \$20, irrespective of the active power consumption.

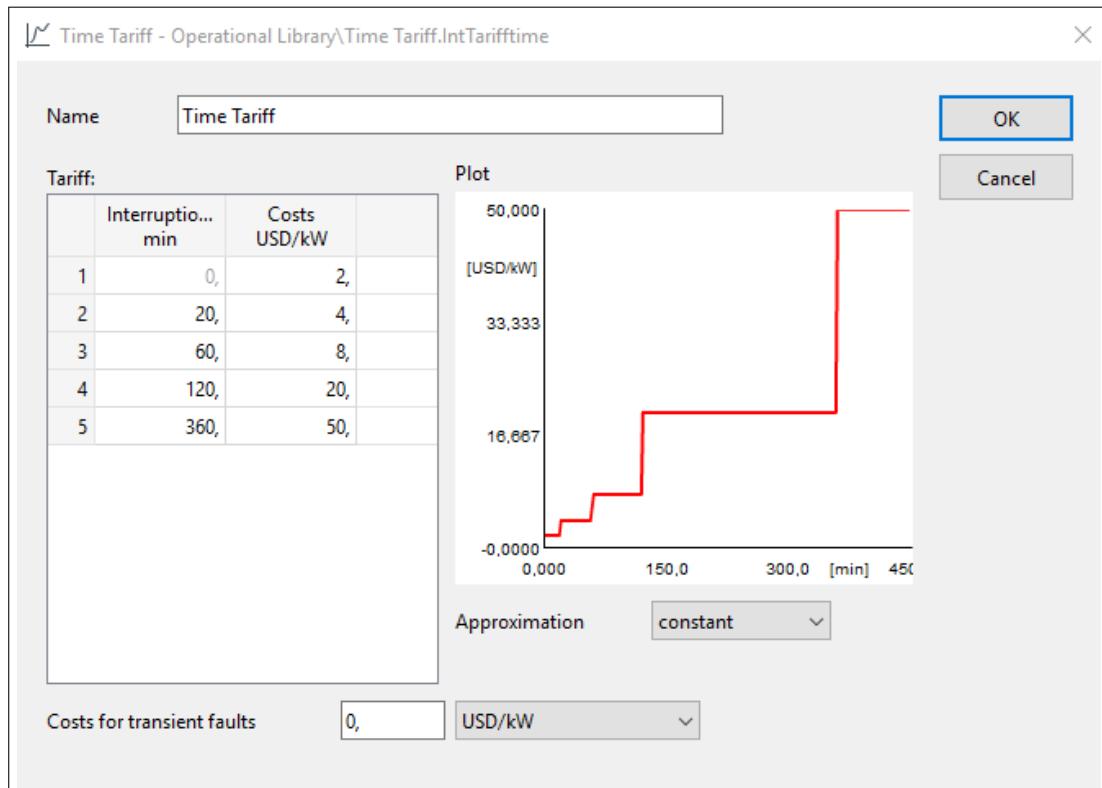


Figure 18.5.1: Example Time Tariff

## 18.5.2 Defining Energy Tariffs

An energy tariff characteristic can be defined by taking the following steps:

1. Choose the 'Select' option from the 'Tariff' selection control on the reliability page of the load element. A Data Manager browser will appear with the *Operational Library* → *Tariff* folder selected.
2. Optional: If you have previously defined a 'Tariff' characteristic and want to re-use it, you can select it now. Press **OK** to return to the load element to reliability page.
3. Create an energy tariff object by pressing the *New Object* button  from the data browser toolbar.
4. Select 'Energy Tariff' and press OK. An 'Energy Tariff' dialog box will appear.
5. Enter Energy and Costs values for the Energy Tariff (right-click and 'Append rows' as required).
6. Press **OK** to return to the load element reliability page.

### 18.5.2.1 Example Energy Tariff

An example Energy Tariff characteristic is shown in Figure 18.5.2. In this example, 'Approximation' is set to 'constant', i.e. no interpolation between data points. A fault which leads to energy not supplied of 2.50 MWh would result in a cost of

$$\$9,20 \cdot 2,50 \cdot 1000 = \$23000 \quad (18.1)$$

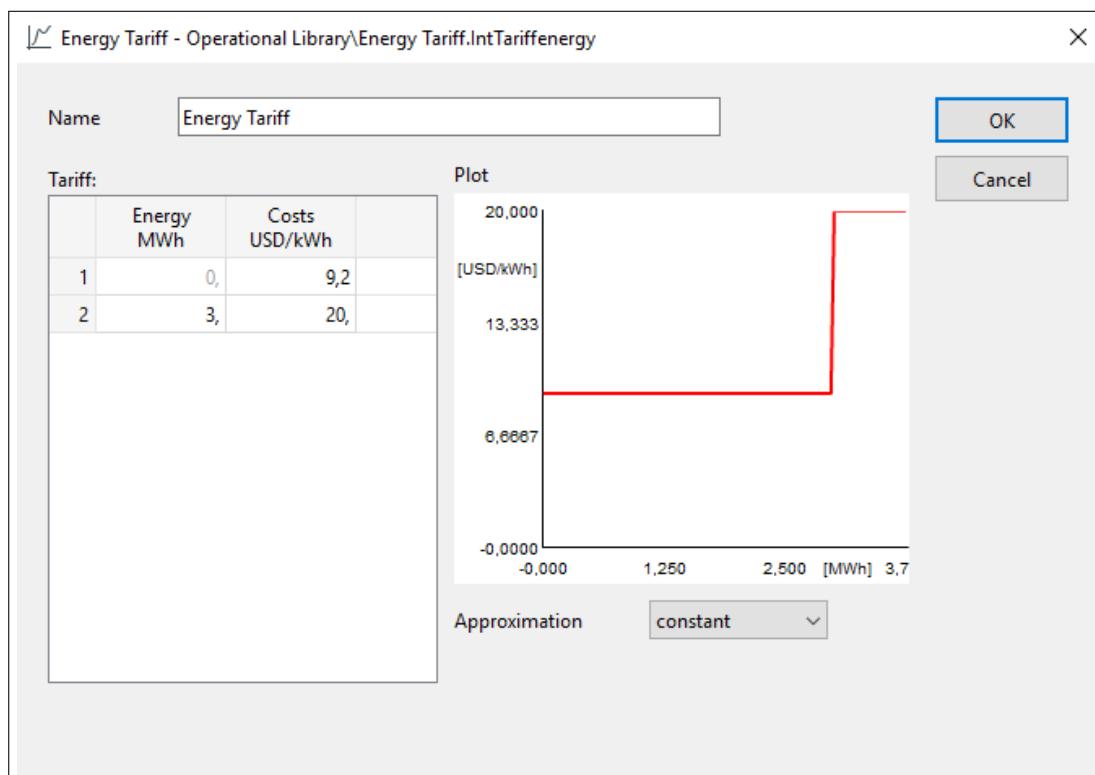


Figure 18.5.2: Example Energy Tariff

# Chapter 19

# Reporting and Visualising Results

## 19.1 Introduction

This chapter introduces the tools and options in *PowerFactory* for presenting the calculation results. Key concepts in this topic are Result Boxes, Output Reports, Results Objects, Variable Selection and Plots. The structure of this chapter is as follows:

- Section 19.2 provides the instructions for customising the result boxes displayed in the single-line, overview and detailed diagrams. Instructions about selecting the predefined formats are given in chapter Network Graphics, Section 10.5.
- Section 19.3 describes the *Variable Selection* object, which is used to define the variables to be presented, either in the Result Boxes, *Flexible Data* page or Results Files.
- Section 19.5 describes *Report Generation* command, used for generating inbuilt or customised reports.
- Section 19.6 describes the option to compare steady state calculations results.
- Section 19.7 describes the *Results File* object to store results or selected variables.
- Section 19.8 lists and describes all the plot types available in *PowerFactory* and the tools used to modify/customise them.

## 19.2 Result Boxes

Results are displayed with help of result boxes in the single line diagrams. Several predefined formats can be selected, as described in Chapter 10, Section 10.5 (Result Boxes, Text Boxes and Labels).

The result box itself is actually a small output report, based on a form definition. This form definition is used to display a wide range of calculated values and object parameters, and can be also be used to specify colouring or user defined text.

### 19.2.1 Editing Result Boxes

To edit result boxes the so-called “Format” dialog is used. In this dialog, text reports can be defined, from very small result boxes to more complex and comprehensive reports within DlgSILENT *PowerFactory*.

The Format object (*IntForm*), shown in Figure 19.2.1, will be used in most cases to change the contents of the result boxes in the single line graphic; the Format dialog is accessed by right-clicking on a result box and selecting the option *Edit format for...*

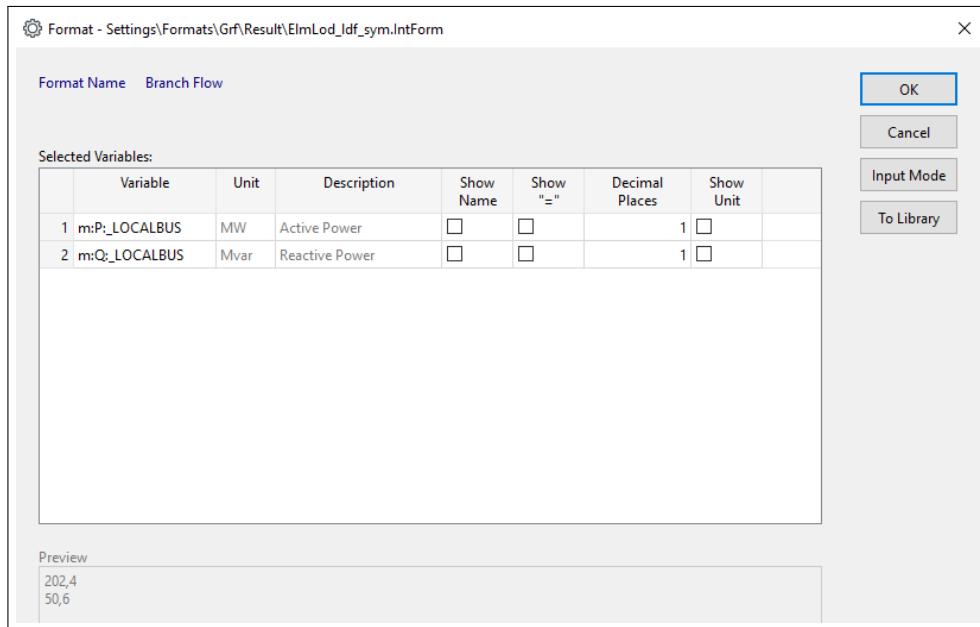


Figure 19.2.1: The Format dialog

The format defined in this dialog can be saved for later use by clicking on the button **To Library** and defining a user-specific name for it.

Such saved formats are stored in the user's own settings folder, and can therefore be selected for use in any of the user's projects.

This Format dialog has a page to change the format by selecting variables and a page to manually define a format. What is displayed on this page depends on the input mode; that can be changed using the button **Input Mode**. Both options are described in the following sections.

### 19.2.1.1 Input Mode - User Selection

When using this input method it is possible to select any number of parameters out of all available parameters for the selected object or class of objects. This includes model parameters as well as calculated values.

Different variables can be added by appending new rows. By double clicking on the corresponding row in the column *Variable*, a Variable Selection showing the list of all available variables will appear. More information about Variable Selection is available in Section 19.3.

It is also possible to define how the variable will be shown by selecting the columns *Show Name*, *Show "="*, *Decimal Places* and *Show Unit*. A preview of the result box is shown in the *Preview* field.

### 19.2.1.2 Input Mode - Format Editor

This is the most flexible, but also the most difficult mode. In this mode, any text and any available variable, in any colour, can be entered in the Form. The highly flexible *DlgSILENT* output language allows for complex automatic reports. The **User defined** button acts like the input mode *User Selection*.

with one important difference: where the *User Selection* mode is used to redefine the complete form text, the **User defined** button appends a line for each set of variables to the existing form text.

For example if the active and reactive power of an element have been selected using the input mode *User Selection*, when switching to *Format Editor* the variables will be shown in the *DlgSILENT* output language code like this:

```
#.## $N,@:m:P:_LOCALBUS  
#.## $N,@:m:Q:_LOCALBUS
```

This example shows the basic syntax of the *DlgSILENT* output language:

- The '#' sign is a placeholder for generated text. In the example, each line has a placeholder for a number with two digits after the decimal point ('#.##'). The first '#' -sign stands for any whole number, not necessarily smaller than 10.
- The '\$N' marks the end of a line. A line normally contains one or more placeholders, separated by non-'#' signs, but may also contain normal text or macro commands.
- After the '\$N', the list of variable names that are used to fill in the placeholders have to be added. Variable names must be separated by commas. Special formatting characters, like the '@': -sign, are used to select what is printed (i.e. the name of the variable or its value) and how.

The Format Editor offers options for the unit or name of the selected variable. If the *Unit-show* option is enabled, a second placeholder for the unit is added:

```
#.## # $N,@:m:P:_LOCALBUS,@:[m:P:_LOCALBUS  
#.## # $N,@:m:Q:_LOCALBUS,@:[m:Q:_LOCALBUS
```

The '[' -sign encodes for the unit of the variables, instead of the value.

The same goes for the variable name, which is added as

```
# #.## $N,@:~m:P:_LOCALBUS,@:m:P:_LOCALBUS  
# #.## $N,@:~m:Q:_LOCALBUS,@:m:Q:_LOCALBUS
```

where the “~” -sign encodes for the variable name. With both options on, the resulting format line

```
# #.## # $N,@:~m:P:_LOCALBUS,@:m:P:_LOCALBUS,@:[m:P:_LOCALBUS
```

will lead to the following text in the result box:

P -199,79 MW

Other often-used format characters are '%', which encodes the full variable description, and '&', which encodes the short description, if available.

For a detailed technical description of the report generating language, see Appendix 55 (The *DlgSILENT* Output Language).

## 19.3 Variable Selection

Variable Selection (*IntMon*) objects are used to select and monitor variables associated with objects in the data model. The variable selection object can be used to select the variables to be recorded during a calculation (e.g. RMS/EMT simulation, quasi-dynamic simulation, harmonic analysis) and to define the variables to be displayed in the result boxes and in the *Flexible Data* page (see Section [Flexible Data Page](#)).

An example of a variable selection dialog is shown in Figure 19.3.1.

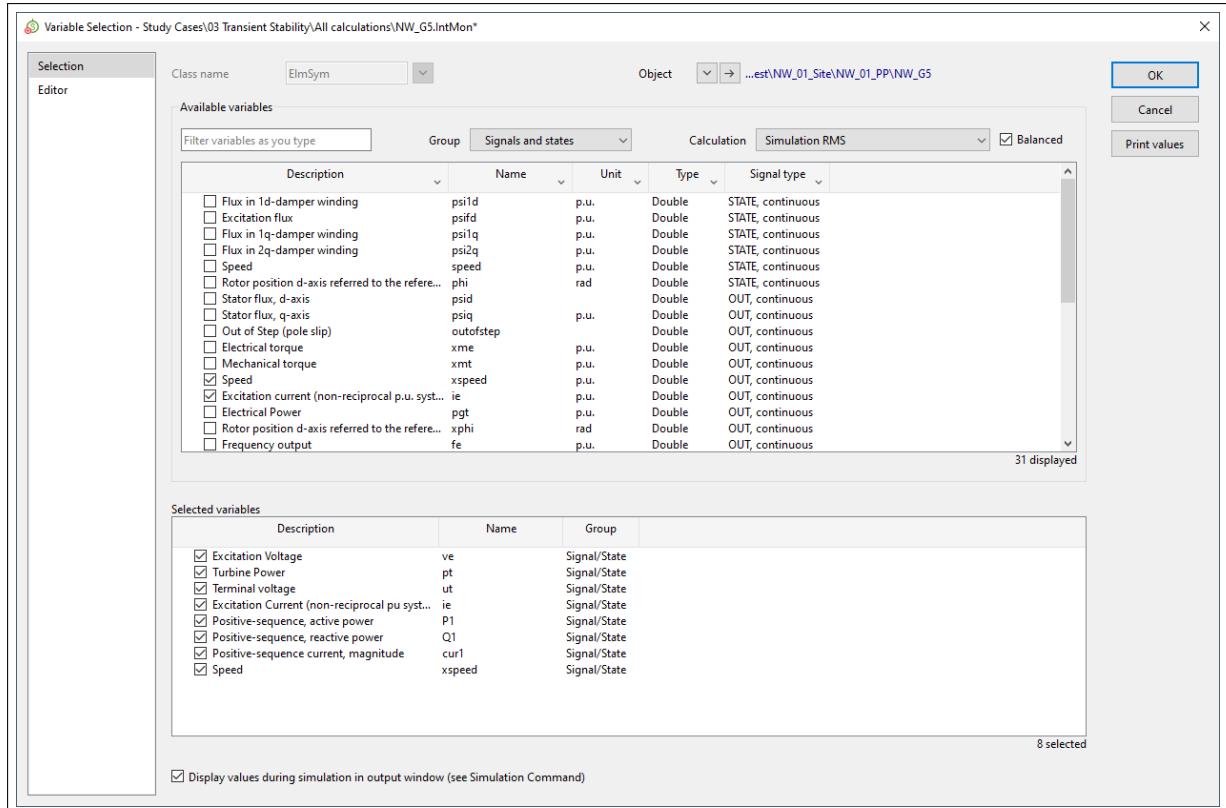


Figure 19.3.1: Example of a variable selection dialog

### 19.3.1 Selection Page

The *Selection* page contains three main areas: object, available variables and selected variables.

#### 19.3.1.1 Object or Class Name

The object field has a reference to the object whose variables are monitored and the class of the selected object. If no object has been selected, the *Class name* field becomes active. For some calculations, for example quasi-dynamic simulation and contingency analysis, the class name is used instead of the object.

These fields are usually automatically assigned by the way the dialog is opened.

#### 19.3.1.2 Available Variables

In the *Available variables* field, the variables to be monitored can be selected. There are three filters to help to find the variables: the *Group* and *Dialog page / Calculation* determines which variables are displayed in the *Available variables* window and the *Filter variables as you type* can be used to filter the displayed variables, either by description or by name.

The displayed variables can be filtered with the *Group* and *Dialog page / Calculation* filters as follows:

- **Group All:** all the variables of the object/class are displayed.
- **Group Input data:** input parameters that belong directly to the selected object, which can be additionally filtered using the *Dialog page* filter, where one of the pages of the edit dialog of the object can be selected. The options also include *All* and *Not displayed in dialog*.

- **Group Results:** previously known as *Calculation Parameters*, are variables derived from the primary calculations (i.e. currents, loading, power, losses, etc.), from input data (i.e. the absolute impedance of a line, derived from *impedance/km \* linelength*), or that have been transformed from input data to a format useful for calculation (actual to per unit), or that are required for such transformation (e.g. rated power). The parameters that actually are available depend on the object type.

Results can be filtered using the *Calculation* filter, where one of the calculations or the options *All* and *Independent of calculation* can be selected.

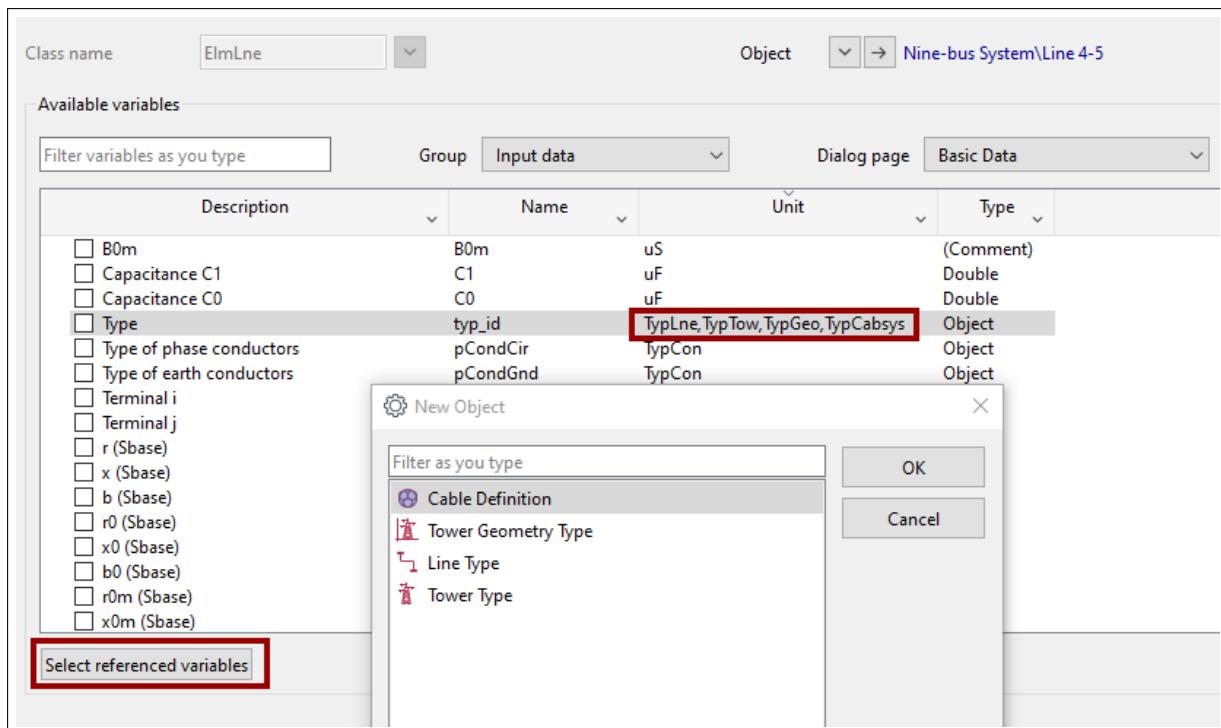
- **Group Results per connection:** Monitored or measured variables. These are the outputs as calculated by a calculation function selected via the *Calculation* filter.
- **Group Results per bus:** variables that belong to the bus/es where the element is connected. Can be filtered using the *Calculation* filter, where one of the calculations or the options *all* and *independent of calculation* can be selected.
- **Group Signals and states:** variables that can be used as interface between user-defined and/or *PowerFactory* models (inputs and outputs). These variables are accessible whilst an iteration is being calculated, whereas the other variables sets are calculated following an iteration. The signals and states can be:

- Input signal (IN)
- Output signal (OUT)
- State variable (STATE)
- State derivative (d/dt)

The *Balanced* checkbox can be used, depending on the type of calculation to be monitored, to toggle between balanced and unbalanced variable selections. When not checked, i.e. unbalanced variables are required, an additional column *Phase*, will be shown with an indication of which variables are phase-wise.

The columns of the *Available variables* area can be sorted by clicking on them or further filtered using the drop-down arrow on each column header.

For variables that are linked to an object, i.e. variables whose unit is an object (e.g. Int\*, Elm\*, Typ\*, Sta\*), an additional button **Select referenced variables** is displayed when clicking on the variable row. If the linked object can only be one class, a new variable selection dialog will open to select the variables of the referenced object. Otherwise, i.e. if the object can be more than one class, a dialog will open to select first the corresponding class as shown in Figure 19.3.2 for the variable *Type* of a line.

Figure 19.3.2: Referenced variables of a *TypLne*

User-defined variables are shown when the dialog page *Data Extensions* is selected. Additional information about Data Extensions is available in Chapter 20.

Clicking on the checkbox on the left side of the variable will add that particular available variable to a *Selected variables* list. To multi-select variables, select them first and then click on one of the checkboxes.

### 19.3.1.3 Selected Variables

The selected variables are displayed on the *Selected variables* window. Depending on the variable type, additional columns will be displayed.

- Bus:** for the variables of the *Results per connection* and *Results per bus* groups, of elements having more than one connection, an additional column called *Bus* will be displayed. Here the different buses can be selected.
- Phase:** when selecting unbalanced variables (*Balanced* checkbox unchecked), the corresponding phase/s can be selected in the column *Phase*.
- Function:** for complex variables, the column *Function* should be used to define which variable should be displayed, as shown in Figure 19.3.3.

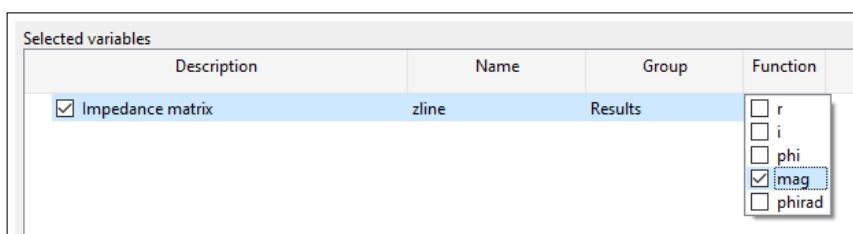


Figure 19.3.3: Complex variables selection

- **Statistic:** for double statistic variables, typical result of an optimisation, the column *Statistic* can be used to select the statistical values of the control variables.
- **Subindex:** for arrays, vectors, matrices and multi-line strings variables, the column *subindex* should be used to define the index to be displayed.

---

**Note:** The indexing of the variable depends on the object:

- Stored results (e.g. *Results* objects) use 0-based indexing
- Vector variables use 0-based indexing
- *Modelica Models* internally use 1-based indexing

For more information, refer to the Technical Reference documentation of a specific model or to the Modelica dynamic models described in Section [30.8](#).

---

#### 19.3.1.4 Print Values

The **Print values** button on the right side of the dialog will display the current values of all the selected variables in the output window.

If the variable definition is being done for a time domain simulation (RMS/EMT), the option *Display Values during simulation in output window* will be visible. By checking this box and selecting the option *Display results variables in output window* in the simulation command, the values calculated for the selected variables during a simulation will be displayed in the output window.

#### 19.3.2 Editor Page

On the *Editor* page, the variables can be manually input. Or copied to be used in, for example a script. All the variables' names are preceded by a letter depending on the group and type of variable; these can be:

- **e:** for the group *Input data*
- **m:** for the group *Results per connection*
- **n:** for the group *Results per bus*
- **c:** for the group *Results*
- **s:** for the group *Signals and states*
- **t:** for the parameters of the referenced object "type"
- **r:** for the parameters of a referenced object

#### 19.3.3 Format/Header Page

The *Format/Header* page is only visible when selecting the variables of the flexible data page (Figure [19.3.4](#)) and can be used to change column headers and the format of the displayed variables.

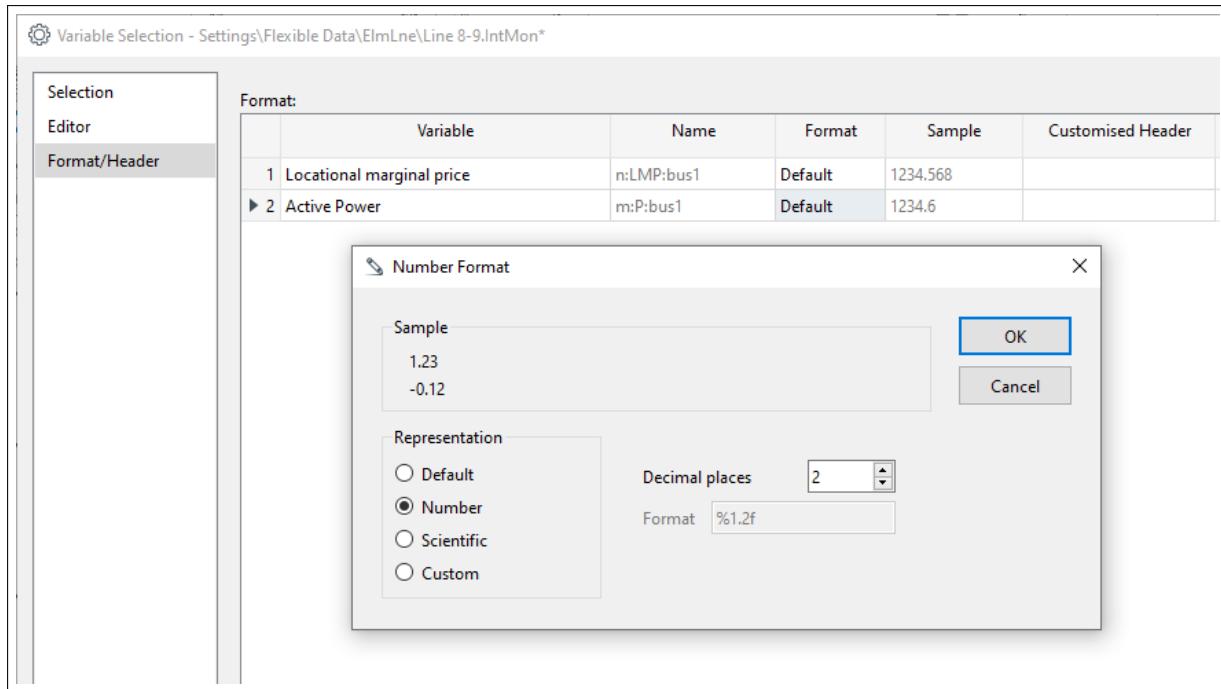


Figure 19.3.4: Customising flexible data page formats and column headers

More options for customising the flexible data page are given in section [Customising the Flexible Data page](#).

## 19.4 Documentation of Device Data

The *Output of Device Data* command (*ComDocu*), also referred to as *Documentation of Device Data*, is a way of listing the data associated with the *PowerFactory* network elements, types etc. in the project. It can be accessed by clicking on the icon  on the main tool menu. When the command is executed, the information is written to the output window.

### 19.4.1 Documentation of Device Data - Settings

#### The Short Listing

The “Short Listing” reports only the most important device data, using one line for each single object, resulting in concise output. Like the “Output of Results”, the “Short Listing” report uses a form to generate the output. This form can be modified by the user. When the report form is changed, it is stored in the “Settings” object of the active project, so does not influence the reports of other projects. The output of objects without a defined short listing will produce warnings like:

Short Listing report for StoCommon is not defined.

#### The Detailed Report

The detailed report outputs all device data of the elements selected for output. In addition, type data can be included (“Print Type Data in Element”). Device Data is split into the different calculation functions like “Load-Flow” or “Short-Circuit”. The “Basic Data” is needed in all the different calculations. “Selected Functions” shows a list of the functions whose data will be output. To report the device data for all

functions, simply move all functions from left to right. If “Selected Functions” is empty no device data will be output.

### Device Data

- **Use Selection:** the set of reported elements depends on the *Use Selection* setting. If *Use Selection* is checked one element or a set object must be chosen for output. If *Use Selection* is not checked, the *Filter/Annex* page specifies the set of elements for the report.
- **Annex:** each class uses its own annex. There is either the default annex or the individual annex. To use the default annex check *Use default Annex*. Changes of the annex are stored in the *Settings* of the active project. The local annex is stored in the *Documentation of Device Data* command. To modify the local annex press the **Change Annex** button.
- **Title:** most reports display a title on top of each page. The reference *Title* defines the contents of the header.

#### 19.4.2 Documentation of Device Data - Filter/Annex

If one wants to report elements without defining a set of objects, *Use Selection* on the *Device Data* page must not be checked. The objects in the list *Selected Objects* will be filtered out of the active projects/grids and reported. *Available Objects* shows a list of elements which can be added to the *Selected Objects* list. The list in *Available Objects* depends on the *Elements* radio button. Elements in the left list are moved to the right by double-clicking them. The text in the *Annex* input field will be set as default annex for the selected class.

### The Annex for Documentation

The *Annex for Documentation* stores the annex for the documentation of results. The annex number and the page number for the first page are unique for each class.

- **Objects:** this column shows the different classes with their title.
- **Annex:** this column stores the annex number shown in the Annex field of the report.
- **First Page:** this column defines the start page for the class in the report. The first page number depends on the class of the first element output in your report. The page number of its class is the page number of the first page.

## 19.5 Output Reports

The results of calculations are presented to users in a number of different ways, depending on the particular calculation function. Many functions have their own dedicated tabular reports, but *PowerFactory* also has a flexible reporting system, which is able to generate reports in PDF and many other formats. This reporting capability is provided by the Report Generation (*ComReport*) command, accessed via the  icon, found on the main toolbar. The use of the command and different options available are described in Section 19.5.1.

For certain calculations, reports were previously generated just as ASCII text in the output window, but now the user can instead easily create individual or combined reports, which make use of inbuilt report templates (see Section 19.5.4).

Reports may be exported or simply viewed as PDFs in *PowerFactory* using the PDF Viewer, described in Section 19.5.2

Although the creation of standard reports is very easy, users can also create and use new report templates, either completely from scratch or based on existing templates. This is done using the *Report Designer*; see Section 19.5.5 for more information.

## 19.5.1 Report Generation Command

The Report Generation command is used once a calculation has been executed, and by default will offer to the user all reports that are relevant for these latest calculation results. The user can select one or more reports and decide whether these should be generated separately or combined into one. There are options for filtering what should be reported and options regarding report format and exporting.

By default, reports are viewed using the inbuilt PDF viewer (see Section 19.5.2). They appear as tabs in the graphic window, and are stored as objects in the desktop of the active study case. The alternative is to generate reports to be exported from *PowerFactory*.

There are two reports that are offered as standard, whatever calculation has just been executed:

- **General Settings**

This report shows user settings and selected configuration details, together with the settings of the currently active project.

- **Project Overview**

This report provides an overview of the contents of the currently active project, including details of the active study case.

### 19.5.1.1 Basic Options

#### Available Reports

This list is divided into four sections:

- **Project Library:** this will only be shown if the project contains new or customised report templates in the project library.
- **Custom Library:** this will only be shown if users have created a custom library (see Section 4.5.2) and stored new or customised report templates in it.
- **DIGSILENT library:** standard report templates are listed, appropriate to the calculation just executed.
- **DIGSILENT library - Legacy:** Any applicable “Legacy” ASCII or tabular reports are listed here. Other settings within the dialog do not affect these reports. Note that this list does not include the main tabular reports that are generated using function-specific report commands.

Reports are selected using the check boxes, and these reports then appear in the *Selected Reports* panel to the right.

Other options are available via a context menu (right-click), for reports other than the “legacy” reports. Actions such as copying reports into other libraries, or deleting them, are only available if the user has write-access to the target library.

- Open the edit dialog of the selected report
- Copy the selected report to another library
- Duplicate the report
- Delete a report

Underneath the list is a **Create new report...** button, which allows the user to create a new report template. Section 19.5.5 describes this process.

## Selected Reports

This is the list of reports to be generated. It can be modified by clicking on a report and using buttons below to move it around in the list, or remove it. Drag and drop can also be used to change the order of the reports.

## Parameters

Some reports require input parameters to be set. If any of the selected reports has parameters, a blue message "Parameters are required for some selected reports, see page Parameters" will appear. The Parameters page is described below (Section 19.5.1.4).

## Document

In this panel, the user chooses whether to generate separate reports or merge them into one document, in which case a name can be chosen.

## Manage

The **Manage** button takes the user to the *Report Manager*, described in Section 19.5.3. For normal report generation using the inbuilt templates, there is no need to use this.

### 19.5.1.2 Page Layout

This page of the report dialog allow the user some control over the appearance of the document.

## Page format

This part of the dialog allows the user to customise the format of the report if required, although the default is to use the format already built in to the selected reports.

A drop-down menu offers a wide range of standard document formats, and a separate drop-down menu gives the choice between Landscape and Portrait.

For customised sizing, a **New...** button allows the user to create a new *SetFormat* object with the required dimensions. Such user-defined page format objects are held in the Settings folder of the project, and will be added to the drop-down menu list. A selected format can also be edited, using the **Edit...** button.

## Title Page

This refers to the design of the (optional) title page and contents.

The inbuilt options are:

- **PowerFactory Title Page Design:** (Default) A front title page and a contents page
- **PowerFactory Title Page Basic:** Just a contents page

The right-arrow gives access to the currently-selected title page template; the down-arrow brings up further options:

- **Select:** A dialog is presented, showing the available title page templates, both those in the *DlgSILENT* library and any user-defined templates. At the bottom of the dialog, a **Create new...** button can be used to create a custom title page report template. See Section 19.5.5.
- **Copy:** To copy the currently-selected template.
- **Paste:** To paste in a previously-copied template.

- **Reset:** Can be used to clear the selection, so that there will be no title page at all.

### Header/Footer

In a similar way to the title page, the header and footer can be selected. Note that the header contains a date, which is automatically generated as the current date.

### Title information

The fields in this panel allow the user to provide a title to the report, with the term “Project” intended in a general sense, rather than being necessarily the *PowerFactory* project name. These fields are only editable if the selected Title Page or Header/Footer templates make use of them.

### Title page parameters

These parameters allow additional information to be supplied to the title page template. For example, the default *PowerFactory Title Page Design (DlgsILENT Library)* template has three parameters; these can be used to provide three extra lines of information on the front page of the report.

### Header/Footer parameters

These parameters allow additional information to be supplied to the header and footer templates. For example, the default *PowerFactory Header (DlgsILENT Library)* template has a parameter for showing or not showing the *DlgsILENT* logo.

### Use custom style

By default, reports are generated using a default *DlgsILENT* style, which contains colour, font and format settings for components such as report titles, chapter headers, data cells, and so on. All the reports in the *DlgsILENT* library uses the *DlgsILENT* default style and users, too, can apply the *DlgsILENT* default style to their own reports.

In order to customise the look of user-defined or *DlgsILENT* reports, users can change the settings of the default style and create a custom style. To find out how to create customised styles, see Section [19.5.5.2](#).

#### 19.5.1.3 Filters

On this page, the user is able to control the amount of reported output by applying filters. These filters limit the network elements that are “collected” for reporting. All the filters are multiplicative, for example if the voltages are restricted to the range 110-220 kV and a grouping filter is applied for Grid 1, the only terminals reported will be those in Grid 1 that *also* lie within the required voltage range.

### Restrict voltage levels

Minimum and maximum voltage levels may be specified here.

### Grouping filter

Groupings refer to the grouping objects Grids, Feeders, Operators, Owners, Zones, Areas and Boundaries, which can be used to group network elements. If this option is selected, only elements within the specified grouping objects will be considered. The *Operation* is also selected as follows:

- **Union:** an element is considered if it is contained in any of the selected groupings.
- **Intersection:** an element is only considered if it is contained in all of the selected groupings.

### Custom filter

Custom filters are defined by the user and can be used to select elements according to other rules. General information about setting up filters can be found in Section [11.3.3](#).

### Selection filter

This option is provided so that output can be restricted, if required, to a specific set of network elements. These network elements can be selected as a set. To define a set, the user can select the required elements then do right-click, *Selections → Set - General*.

#### 19.5.1.4 Parameters

Aside from the filters described above, some reports have parameters as input, enabling the user to determine what should be reported. For example, a report of voltage violations will have parameters specifying the limits of allowed voltage. On this page, it is possible to view and modify any parameters. A drop-down menu allows the selection of each relevant report.

#### 19.5.1.5 Export

By default, reports are generated as PDFs and viewed using the PDF Viewer built into *PowerFactory*. The alternative is to export reports, for which PDF or one of many other file formats can be selected. If reports are exported as PDF *and* automatically opened (see below) the PDF viewer used will be determined by the setting on the External Applications page of the *PowerFactory Configuration* dialog (see Section [5.2.4](#)).

##### Export generated documents

If this option is selected, a target folder for the exported documents must be defined.

##### Export File format

The default, and recommended, format for exporting files is PDF, but a number of other formats are also available.

##### Open documents after export:

- **Not selected:** the reports are simply created in the specified location.
- **Selected:** the reports are automatically opened after being created.

##### Document Settings

Depending on the file format selected, additional options such as *Image quality* are offered. Specifically,

- **Enable smooth upscaling (PDF):** Used to improve the appearance of the report if low resolution images are used.
- **PDF/A Compliance (PDF):** PDF/A is a standard that identifies a “profile” for electronic documents that ensures that the documents can be reproduced exactly the same way in future years. This option can be used to ensure compliance with the selected standard.
- **Embed fonts (PDF):** By default this option is not selected. If the PDF document makes use of proprietary fonts on the user’s computer, embedding such fonts could risk breach of terms and conditions when the document is shared. The option can of course be selected if there are no such concerns.

- **Author** (PDF): An author's name can be assigned; this will be seen in the document properties when the document is opened.
- **Export report pages to separate Excel sheets** (Excel)
- **Export page breaks** (Excel)
- **Continuous page** (HTML 5 (ZIP archive))

### 19.5.2 PDF Viewer

When a PDF report is viewed inside *PowerFactory*, the inbuilt report PDF Viewer is used. This presents the reports in a separate panel of the graphic window, in one or more tabs. These can be managed as with other graphic and reporting tabs, for example moved to form a floating window.

When a report is generated to be viewed inside *PowerFactory*, it is saved in the desktop of the active study case. The report objects are persistent, meaning that they are not deleted when the tab is simply closed (as opposed to actively selecting a context menu option on the tab to delete the report). To look at reports again after they have been closed, the Windows menu in the main tool-bar or the drop-down arrow at the left-hand end of the graphics tool-bar both offer an option *Open Report Document...*. This will bring up a dialog showing the reports available in the study case; it also provides a convenient way to access and delete unwanted reports.

The options available in the viewer toolbar are:

- **Bookmarks:** if this button is active, the “Bookmarks” pane is shown to the left of the document; it acts as a table of contents and can be used to navigate the report.
- **Zoom and pan:** these buttons work in the same way as for network graphics.
- **Zoom to page:** if this button is active, the zoom factor is automatically adapted to display complete pages.
- **Fit to width:** if this button is active, the zoom factor is automatically adapted in order to display the complete (page) width of the document.
- **Zoom factor box:** This can be used to change the zoom factor to one of the pre-set values or a manually-entered value.
- **Search field and Find Previous/Next buttons:** a text string can be typed into the field and the Find Previous/Next buttons used for searching. The search is case-insensitive.
- **Copy:** used to copy currently selected text.
- **Export:** the document can be exported to a selected location.
- **Print:** Opens the standard printing dialog. A selected report can also be printed using *File → Print...* or *Ctrl+P*.

Below the document, page navigation buttons are provided. General information about the document (title, author and creation date) is also shown.

### Hyperlinks

Where appropriate, hyperlinks are generated for network elements mentioned in the report. A left-click on such a hyperlink will open the edit dialog of the element; right-clicking gives access to other options such as mark in graphic.

Using the Report Designer, users may include a variety of links (e.g. to their company's web site) in the reports they have created. For security reasons, these links are not clickable in the PDF Viewer. The links will work, however, when the report is exported and then opened in another PDF viewing application.

### 19.5.3 Report Manager

On the right-hand side of the Report Generation Command dialog, a **Manage** button gives access to the *Report Manager* command. This lists all available report templates (including title page and header/footer templates) from the *DlgSILENT* library, and where applicable the Custom library and the Project library of the currently active project.

Reports may be managed in two ways:

1. By right-clicking on a listed report. This brings up a context menu; the options available depend on whether the template is inside the project library or is elsewhere.
2. Using the two buttons above the list:

#### Create new report...

If this button is used, a new report template is added to the Project library and its Report template (*IntReport*) dialog is opened. Note that if the dialog is cancelled, the newly created report is removed again. Section 19.5.4 below describes the Report template.

#### Copy report to project library..., Copy report to custom library...

The *Copy report to project library...* button allows the user to select a report template and copy it to the project library; it will then open its Report template dialog. Report templates can be selected from any location (e.g. project libraries of other projects or locations outside of any of the three libraries). If a report is currently selected in the report table, that report will be initially selected when clicking on this button.

Likewise, if the user has write-access to a custom library, a *Copy report to custom library...* button will be available. This works in a similar way.

#### Show filter, sorting and column options

If this box is checked, a range of options is presented, that allows the user to manage the list of report templates. The *Name* field can be used to search for templates by name: enter a text string, then click away from the field.

### 19.5.4 Report Templates

If a report template is edited, for example by double-clicking on a report template that is listed in the Report Manager, the report template dialog opens. The following sections explain the options available on each page of the dialog.

#### 19.5.4.1 Basic Options

- **Usage:** The intended use of the report template can be selected, i.e. whether it should be treated as a regular report or used as a Title Page or Header/Footer template.
- **Localised name:** This shows the localised text (i.e. text that depends on the current *PowerFactory* application language) that should be used as the report name. The  icon can be used to edit it. If no localised text is set, the object name (as given in the Name field) is used as report name. User-defined localised texts can be added and edited on the *Localisation* page (see Section 19.5.4.4 below).
- **Restrict to specific calculation models:** Newly created reports are considered calculation-independent, i.e. they can be selected in the Report Generation Command for any active cal-

culation or even if no calculation is active. This button allows the user restrict a report to specific calculation modules.

- **Supported calculation modules:** This panel and the two buttons below are only visible if a restriction is in place. It shows the list of calculation modules for which this report is available.
- **Edit...:** This button can be used to edit the restrictions list.
- **Remove restrictions:** This button can be used to remove restrictions, so that the report is available for all calculation functions.
- **Use settings from command in Report Designer:** If the report template has been opened from within a Report Generation command (*ComReport*) (either by editing an existing template or creating a new one), this option allows the user to transfer some of the options from the Report Generation command into the Report Designer if/when it is used, namely:
  - Any net element filters enabled on the Filters page of the command will be applied to the collected objects that are used to create tables. This is especially useful when editing a report template while working in a large project.
  - Instead of using their initial values, the parameters of the report template are filled with the respective values defined in the command.
  - In the (rare) case that the report template has dependencies (i.e. objects required by built-in extensions), the dependencies set in the command (instead of the ones set on the Dependencies page, see below) are used.

#### 19.5.4.2 Variable Selection

This page allows the addition, editing and removal of the Variable Selections (*IntMon* objects) that will be used to create object tables used by the Report Designer.

---

**Note:** After removing variables from a Variable Selection or deleting a complete Variable Selection, the report might need to be opened in the Report Designer in order to remove any references to the corresponding field or table.

---

The *Show generated tables...* button allows the user to view *all* tables generated by the report template, including:

- *Object tables*, which are based on Variable Selections;
- *Meta tables*, which provide information about the fields of their corresponding object table;
- the *Localisation* and *UserLocalisation* tables (see Localisation page, Section 19.5.4.4), which contain internal and user-defined localised texts, respectively;
- the *Parameters* table (see Parameters page, Section 19.5.4.5), listing all report parameters (filled with the respective initial values);
- tables created by the Extension Script and any built-in extensions of the report (see Extensions page, Section 19.5.4.6).

After selecting a table in the Table list, the name and source of the selected table, together with its data, are shown on the right-hand side of the dialog. In some cases, the source can be edited directly using the *Edit...* button.

#### 19.5.4.3 Images

This page allows the user to add link documents (*IntDocument*) to report templates. The Report Designer can then include these in the report design template.

---

#### 19.5.4.4 Localisation

On this page, it is possible to define localised texts for each supported *PowerFactory* language. In the **Visible columns** panel, the user selects which language columns should be visible in the Localisation table below.

When the report is compiled, each localised text in the **Localisation table** will be added as field (using the identifier as field name) to the *UserLocalisation* table and the currently-set *PowerFactory* application language will determine which column of the localisation table is used to fill the fields (using English as default whenever a cell in the corresponding column is empty).

Figure 19.5.1 below shows an example.

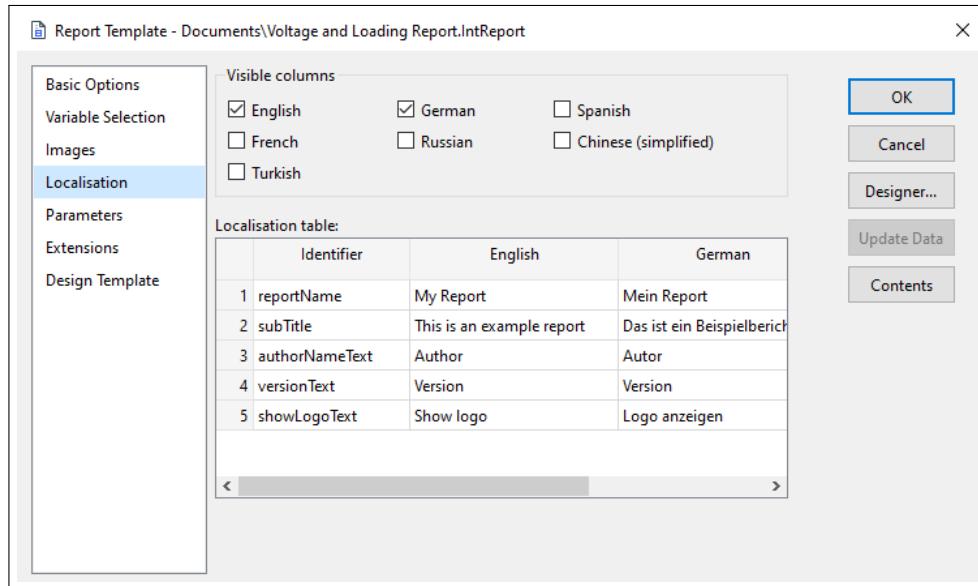


Figure 19.5.1: Defining localised texts in a report template

#### Selecting localised texts

Using the icon next to the Localised name field on the Basic options page (see above) or double-clicking on a cell in the Localised name column of the Parameters table (see next subsection) will open the *Select localised text* dialog. This dialog allows the user to select a localised text from the current report template Localisation table or one of the pre-defined localised texts from the *DlgSILENT* library. In addition, it is possible to remove a previously set localised text.

#### 19.5.4.5 Parameters

Report templates can offer parameters, which are used to dynamically adapt or change the behaviour, content, layout or other characteristics of a report. Each parameter is represented by a field in an automatically created table with name “Parameters”, allowing it to be directly accessed when editing the report template in the Report Designer.

Each parameter is represented as a row in the Parameters table and has the following properties:

- **Identifier:** the name of the parameter and its corresponding field in the Parameters table;
- **Data type:** the data type (string, boolean, integer or floating point value) of the parameter. The data type boolean is internally treated as an integer that can only be set to 0 (=False) or 1 (=True).

- **Initial value:** the default value for the parameter. Users can change this value on the Parameters page of the Report Creation command (*ComReport*) dialog.
- **Localised name:** the localised name of the parameter, used when displaying it on the Parameters page of the Report Creation command (*ComReport*) dialog. If no localised text is set, the identifier is displayed instead.

Whenever the created tables of a report template are viewed or it is opened in the Report Designer, each parameter field is filled with its initial value. When the report template is selected in a Report Creation command (*ComReport*), its parameters use the values provided on the Parameters page (of the Report Generation command) instead. See Section [19.5.1.4](#).

#### 19.5.4.6 Extensions

Extensions allow users to create customised tables and supply additional objects to be collected whenever the report template is compiled (or opened in the Report Designer).

Users can create an extension in the form of an Extension script, which is a DPL (*ComDpl*) or Python (*ComPython*) scripting command that utilises the Report template (*IntReport*) scripting API. The script is created using the *Create extension script...* button. Once an extension script has been created, it can be edited using the arrow next to its name and can be deleted using the *Delete extension script...* button. Please note that a report template can only contain one extension script.

Some reports in the *DlgSILENT* library use inbuilt extensions. These extensions are not based on scripts but rely on inbuilt *PowerFactory* functionality and cannot be edited by users. In such a case, the message “This report uses built-in extensions to create additional tables” will be seen on the Extensions page.

#### 19.5.4.7 Dependencies

This page is only visible if the report template uses inbuilt extensions (see previous subsection) that require specific objects such as result files in order to work properly.

The *Dependencies* table enables these required objects to be temporarily provided so that the report template can be designed and tested with all required data available. Please note that dependencies are not stored persistently in the Report Template and need to be selected on the Basic Options page of the Report Generation command (*ComReport*) dialog before it is executed.

#### 19.5.4.8 Design Template

The design template defines the look and layout of the report and is created and modified whenever the report template is opened in the Report Designer.

In some cases, users might want to abandon a previously created design template and start designing from scratch. This can be achieved by pressing the *Clear* button on this page.

### 19.5.5 Report Designer

Many users will just make use of the standard reports provided within *PowerFactory*, but some users may wish to customise these a little, or create entirely new report templates to their own design.

This is done using the Report Designer. The functionality is provided by *Stimulsoft Reports* software, a comprehensive tool for report design, integrated into *PowerFactory*. A brief overview of the Report

Designer is given in Section [19.5.5.1](#) below, but its use is described in more detail in the Reporting Tutorial.

As well as using the Report Designer to create new and amended report templates (including title page and header and footer templates), it is also used for creating custom styles, which can be selected on the Page Layout page of the Report Generation command. The general process for doing this is described in sections [19.5.5.2](#) and [19.5.5.3](#).

### **19.5.5.1 Report Designer overview**

The Report Designer window is divided into four main components, the page design area, the main menu bar at the top, the toolbox on the left-hand side and the Properties / Report Tree / Dictionary panels on the right-hand side.

#### **Page design area**

This is the central part of the window, which shows the design view of the current page. Much of the design work consists of creating elements in this area and adjusting their position and size. Most other menus and panels refer to elements currently selected in the page design area.

#### **Menu bar**

This menu, at the top of the window, gives access to sub-menus for styling, design and insertion options. It also provides an option for previewing a report as it is being developed.

- **Home:** allows styling of the currently selected element
- **Insert:** components such as pages, bands, labels or charts can be inserted into the document
- **Page:** offers settings for the current page (such as margins and orientation), design grid options and allows showing or hiding of the toolbox and each of the panels
- **Layout:** provides alignment options for the currently selected element(s) and offers the possibility to lock them (preventing any changes to position or size)
- **Preview:** when activated, this sub-menu turns the Page Design Area into a preview of the report document, using the currently provided data

#### **Toolbox**

The Toolbox on the left side of the window offers quick access to the entries of the “Insert” sub menu of the menu bar.

#### **Properties / Report Tree / Dictionary panels**

The area on the right-hand side of the window displays one three different panels. Buttons at the bottom of this area allow the user to select the required panel.

##### *Properties panel*

This panel provides a tabular overview of all properties of the currently selected element. In addition to properties that can be more conveniently edited from the menu bar, the table also contains advanced properties that are not available elsewhere. In particular, it gives access to the events of a component (flash symbol tab), which allow for more advanced scripting and dynamic functionalities.

##### *Report tree panel*

This panel shows the document structure, including all pages, bands and elements. It can be helpful for selecting specific elements directly whenever selection in the page design area proves difficult due

to overlapping elements.

#### *Dictionary panel*

This panel offers access to the main database and other data-related entries. Many entries can be dragged and dropped directly into the Page Design Area to insert a label or table.

#### **19.5.5.2 Custom Styles (general)**

By default, reports are generated using a default *DlgSILENT* style, which contains colour, font and format settings for components such as report titles, chapter headers, data cells, and so on. All the reports in the *DlgSILENT* library uses the *DlgSILENT* default style and users, too, can apply the *DlgSILENT* default style to their own reports.

In order to customise the look of user-defined or *DlgSILENT* reports, users can change the settings of the default style and create a custom style. However, the custom style needs to contain the same categories (report titles, chapter headers, etc.) as the *DlgSILENT* default style in order to work properly.

#### **Creating a custom style**

Typically, the user will make a copy of an existing report template, save it in the project library and customise the style of this copied template. (To make the copy, right click on the standard report in the Available Report list and select “Copy to project library”.)

The report template can then be edited using right-click, Edit.... In the edit dialog for the report template, the **Designer...** button will be active, and can be used to open the Report Designer.

The following steps can then be followed:

- Go to the Home tab.
- To determine the specific component style used by a component of the report, select the component in the Page Design Area, switch to the Properties Panel and look for the entry “Component Style”.
- Then from the main menu at the top, click on (*Style*) → *Style Designer*
- Expand the “DigiStyle” folder and click the relevant category (component style) to see the settings on the right-hand side.
- Settings can be changed as required.
- The Style Designer can be closed using “OK” in order to see the effect of the changes.
- Then the Style Designer can be opened again and under the top menu, *Actions* → *Save As...*, to save the custom style in a selected folder.
- Finally, the Report Designer can be closed. There is no need to save the changed report template if there are no other changes that the user wishes to keep.

#### **Importing the custom style into *PowerFactory***

In the above section, the custom style has been created and saved. In order to use your custom style in a report, you will first need to load the saved style (*sts*) file into a Document (*IntDocument*) in *PowerFactory*:

- In the Data Manager, select a folder, or create a new folder, for storing report styles.
- Use the New Object icon  and type in, and select, “Document”.
- Open the Document object. On the Basic Data page, the ellipse icon (...) is used to select the style (*sts*) file.

- The Document object now has a link to the stored style. The user also has the option to import the style into the database.

### Using a custom style

In the Report Generation command, there is an option on the Page Layout page to “Use custom style”.

The down-arrow next to “Style (general)” can then be used to select previously saved style document (*IntDocument*) object.

With the setting and custom styles selected, every document created by executing this Report Generation command will use that custom style.

#### 19.5.5.3 Custom Styles (table of contents)

The “table of contents” component uses a separate style file, and the process is similar to that for the general style document.

To create a custom style document, open a report in the designer that contains a table of contents. Select the table of contents component, switch to the Properties pane and look for the property “Styles” (in the property group “1 Table Of Contents”). Double click on the value “Styles” of the property to open the Style Designer for the Table of Contents. Adapt the styles, and save the style file.

The style document is then imported in the same way as described above and selected using the down-arrow next to “Style (table of contents)” on the Page Layout page.

## 19.6 Comparisons Between Calculations

At many stages in the development of a power system design, the differences between certain settings or design options become of interest. For a single calculation, the “absolute” results are shown in the single line graphics and in the flexible data page of the elements.

When pressing the *Comparing of Results on/off* button (⊕⊖), the results of the calculation are “frozen”. Subsequent calculations results can then be shown as deviations from the first calculation made. The subsequent calculation results are stored together with the first result. This allows the user to re-arrange the comparisons as desired by pressing the ⊕⊖ icon.

The differences between cases are coloured according to the severity of the deviation, making it possible to recognise the differences between calculation cases very easily.

The set of calculated comparisons may be edited to select the cases which are to be compared to each other or to set the colouring mode. When the ⊕⊖ icon on the main toolbar is pressed, the *Compare* dialog will open.

With the *Compare* dialog, the two cases which are to be compared can be selected. Furthermore, a list of colours may be set which is then used to colour the results displayed in the result boxes, according to certain levels of percentage change.

## 19.7 Results Objects

The results object (*ElmRes*,  ) is used by *PowerFactory* to store tables of results.

By default, the results are stored in a proprietary binary format, generally sparsely populated according

to recording options selected by the user. However, the user can, via the project settings (see Section 9.1.3.3), choose instead to store the results in an open database format (SQLite or ODBC). This allows post-processing based on direct file access to be more easily implemented. Section 19.7.2 gives details of the internal layout of the database-based result files.

The typical use of a results object is in writing specific variables during a transient simulation, or during a data acquisition measurement. Results objects are also used in scripts, contingency analysis, reliability calculations, harmonic analysis, etc.

The results object edit dialog shows the following fields:

- **Name:** the name of the results object
- **File path:** is the path where the results file is saved
- **Last modification:** date when the results file was changed the last time
- **Default for:** the default type of calculation
- **Info:** information about the currently stored data including:
  - the time interval
  - the average time step
  - the number of points in time
  - the number of variables
  - the size of the results file
- **Trigger-Times:** trigger times (in case of a *Triggered* default use)

The **Clear Data** button will clear all result data.

---

**Note:** Clearing the data will delete all calculated or measured data in the results file. It will not be possible to restore the data.

---

The default type settings (*Default for* field) are used for two purposes:

1. Creating a new results object and setting the default type to *Harmonics*, for instance, will cause the harmonics command dialog to use this results object by default.
2. Setting the default type to *Triggered* will cause the calculation module to copy and temporarily store signals in that copied results object, every time a Trigger Event becomes active. The *Triggered* default type enables the trigger time fields.

When the **Output Protocol** is pressed, all events that happened during the simulation, recorded by the results object, will be written again into the output window. So one can check which events took place during the last simulation.

The contents of a results object are determined by one or more monitor Variable Selection (*IntMon*) objects. These monitor objects can be edited by pressing the **Variables** button. This will show the list of monitor sets currently in use by the results object.

Selecting a set of result variables, using monitor objects is necessary because otherwise all available variables would have to be stored, which is practically impossible.

By clicking on the **Variables** button, the list of recorded variables is displayed, if the list is empty a new variable selection can be added by clicking on the *New Object* icon (  ). More information about the definition of variable selections is available in Section 19.3.

## 19.7.1 Exporting Results

The stored results for the monitored result variables can be exported by pressing the **Export** button in the results object. This will activate and open the *Result Export* command (*ComRes*), which enables the definition of the format and the file type used to export the results.

### 19.7.1.1 Results Export - Basic Options

On this page the Results File and its information is displayed, and the type of export to be executed can be defined.

#### Export to

The following options are available:

- Output window
- Windows clipboard
- Measurement file (*ElmFile*)
- ComTrade
- Text file
- PSSPLT Version 2.0
- Comma Separated Values (\*.csv)
- Database

If the last option (Database) is used, there is a requirement to configure the access to the database via an ODBC Database Configuration object (\*.SetDatabase), or select an existing database configuration. These database configurations can exist within the project, in a folder, or in a configuration area, for example. If a new database configuration is to be defined, these are the steps:

- Navigate to the chosen location for the new object.
- Click the *New Object* button 
- Select the database system and the ODBC driver. Oracle, PostgreSQL and SQL Server are supported.
- Enter the access details (Username and password) for the database system.
- Enter the database name.

#### Number format

The format used to export the numbers from the results file can be changed between *Decimal* and *Scientific*. The number of decimal places or significant digits, depending on the format used, can also be specified.

This option is only visible when exporting to:

- Output window
- Windows clipboard
- Measurement file (*ElmFile*)
- Text file
- Comma Separated Values (\*.csv)

### Use system separators

When exporting to a comma separated values (\*.csv) file, it is possible to select the separators for the columns and the decimals. One can either choose the system separators or define specific ones.

### Variable selection

By default, the option *Export all variables* is selected, which mean that all the results for all monitored variables are exported. But also a selection of variables can be made by selecting the option *Export only selected variables*.

#### 19.7.1.2 Results Export - Advanced Options

On this page, additional options such as the individual step size and the columns headers of the results file for the export can be defined.

##### Export

- Values: the results values will be exported
- Variable description only: the description of the recorded variables is exported. This is useful for reviewing the stored data.
- Object header only: also useful for reviewing the recorded data; will only export the columns headers.

##### Interval

A *User-defined interval* for the time/x-scale can be set as the minimum and maximum value of the first recorded variable (in time domain simulations this is of course the time).

##### Shift time

When this box is checked, a *new start time* can be defined. This will “move” the results to the starting time.

##### Column header

Here is possible to customise the column header to be exported not only for the element (e.g. name, path, foreign key), but also for the variable (e.g. parameter name, short or long description)

#### 19.7.2 Results in Database Format

As mentioned above, it is possible for users via a project setting to have results written in a database format, which facilitates post processing based on direct file access. This subsection details the database layout used in the result files.

Table **objectLookup** holds the mapping from id to “orig\_index” and “fullPath”.

<b>id</b>	<b>orig_index</b>	<b>fullPath</b>
0	0	../time
1	1	../obj_1
2	2	../obj_2
3	3	../obj_3

Table 19.7.1: **objectLookup**

Table **dataLookup** holds the information about recorded parameters and the corresponding object, and the tableName and columnName where the recorded parameters are stored.

This table also holds extended information about the parameter such as the displayed name (*varName*), its unit (*varUnit*) and so on.

<b>id</b>	<b>tableName</b>	<b>columnName</b>	<b>objectId</b>	<b>varName</b>	<b>varUnit</b>	<b>varDesc</b>	<b>shortDesc</b>	<b>typeCode</b>	<b>bufferIndex</b>
0	steps	C1	0						
1	T0	C1	1						
2	T1	C1	1						
3	T0	C2	2						
4	T0	C2	2						
5	T0	C3	3						
6	T0	C3	3						

Table 19.7.2: **dataLookup**

Table **steps** holds the recording steps. Typically column[0] and (time:t).

<b>id</b>	<b>C1</b>
0	00:00
1	01:00
2	02:00
3	03:00

Table 19.7.3: **steps**

Data tables **T0, T1...**:

<b>id</b>	<b>C1</b>	<b>C2</b>	<b>C3</b>
0	1.01	2.01	3.01
1	1.02	2.02	3.02
2	1.03	2.03	3.03
3	1.04	2.04	3.04

Table 19.7.4: Data table **T0**

<b>id</b>	<b>C1</b>	<b>C2</b>	<b>C3</b>
0	1.51	2.51	3.51
1	1.52	2.52	3.52
2	1.53	2.53	3.53
3	1.54	2.54	3.54

Table 19.7.5: Data table **T1**

## 19.8 Plots

Plots are used for displaying results graphically. The most common use of a plot is to show the results of a time-domain simulation such an EMT or RMS simulation, but there are various other applications, for example to graphically display voltage profiles, results of a harmonic analysis, results of modal analysis, among others. These could be in the form of a bar graph, a plotted curve, single displayed variables or tables of values.

All signals, parameters, variables or other values from *PowerFactory* can be shown in a plot. The variables are normally floating point numbers, but it is also possible to show discrete variables and binary numbers, for example an *out of service* flag or the switching operation of a circuit-breaker.

The plots are inserted using the *Insert Plot* icon from the main menu () , which will open the insert plot dialog, shown in Figure 19.8.1.

Once a plot has been created, it is held in the desktop of the active study case. If the user closes the plot using the x on the tab, or by right-clicking on the tab and selecting *Close page*, the plot is still retained. It can be re-opened from the *Window* menu of the main toolbar, by selecting the option *Open Plot Page*. If the user wants to delete the plot completely, this is done by right-clicking on the tab and selecting *Delete page*.

There are various designs of plot available. The plots can be filtered by the functions where they are normally used. Some plots are typically used for more than one category (e.g. curve plots) and some are meant to be used for specific functions (e.g. correlation plot, time-overcurrent plot). All the plots are listed under the category (*All*) and the recently used in category (*Recent*).

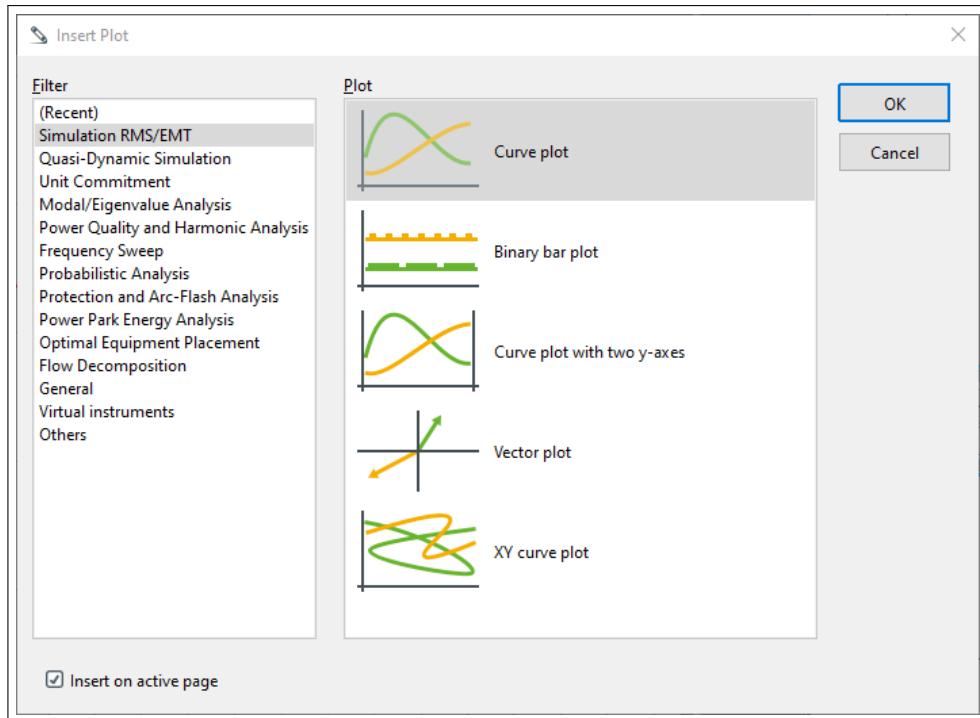


Figure 19.8.1: Insert Plot dialog

All the available plots are described in Section 19.8.9 or in the corresponding chapter (for calculation-specific plots).

The plots have several areas that should be edited separately:

- The curve area (*Data Series*) that can be edited either by right-clicking on the plot and selecting *Edit Shown Data...*, or by double-clicking on the area where the curve is displayed. The functions, common to all the plot types, are described in the Data Series section (19.8.2).
- The plot area, described in Section 19.8.1, that can be edited by double-clicking on an empty area of the plot page (e.g. close to the border).
- The axes and gridlines, accessible by right-clicking on an axis and selecting *Edit Axis...*, or by double-clicking on it. These are described in Section 19.8.4.
- The legend, described in Section 19.8.5 and accessible by right-clicking on the legend and selecting *Edit Legend...*, or by double-clicking on it.

The tools available for modifying plots, such as labels and constants, can be applied equally to most plot types and are described in Section 19.8.6.

The plots can be exported by selecting the option *File → Export → Diagram...* from the main menu or using the *Diagram Export* icon  on the plots toolbar.

The following formats are supported:

- Portable Document Format (\*.pdf)
- Enhanced Windows Metafile (\*.emf)
- Scalable Vector Graphics (\*.svg)
- Portable Network Graphics (\*.png)
- Tag Image File Format (\*.tif, \*.tiff)

- File Interchange Format (\*.jpg, \*.jpeg, \*.jpe, \*.jfif)
- Windows Bitmap (\*.bmp)
- Windows Metafile (\*.wmf)
- Graphics Interchange Format (\*.gif)

## 19.8.1 Plot Area

The plot dialog summarises all the information on the plot, providing the links to all the parts of the plot. In addition, the style and layout of the area outside the curve can be personalised. It can be edited by double-clicking on an empty area of the plot page (e.g. close to the border).

### 19.8.1.1 Basic

On this page, the name of the plot can be modified. In addition, the following links are available:

- **Curves:** a link to the Data Series, described in Section 19.8.2.
- **Title:** clicking on the select arrow opens the edit dialog of the plot title with the following fields:
  - Title: to define visibility and displayed name
  - Positioning and Layout: to define the location of the title of the plot
  - Text Format: to define the font size and colour
  - Border and Background: to define the style of the title
  - Floating position: if the title is moved from the default position, a floating position can be set in this field.
- **Legend:** a link to the plot legend dialog, described in Section 19.8.5.

### 19.8.1.2 Axes

The plot axes often needs to be synchronised for all plots in the Study Case or for all plots on one plot page, for instance to show the same time-scale in all plots.

In this page, the sharing of the axis can be defined. This can also be done by right-clicking on the axis and selecting → *Axis Sharing*.

The sharing options are:

- **Local:** the axis scale, labelling and format are only valid for the local plot.
- **Page:** all the plots on the plot page will use the same axis scale, labelling and format.
- **Desktop:** the axis settings will apply to all the plots in the study case.

The scale, labelling and format options, described in Section 19.8.4, can be accessed by clicking on the → button by the *Used Axis* field.

Additional axes can be added using the button **Create**.

### 19.8.1.3 Style and Layout

On the *Style and Layout* page, the following display options can be selected:

### Border and Background

- **Draw border:** to define outside border and colour of the plot area, i.e. the “non-gridded area”.
- **Fill background:** to define the background colour of the plot area.

### Plot Position on Page (mm)

Defines the position of the plot on the page. Usually this is set using the *Automatic Arrangement Commands* described in Section 19.8.6.4, but it is also possible to define the position on the page using these fields.

### Element Padding (mm)

Defines the size of the plot area outside the curve; this is particular useful when using a coloured background.

### Axis Visibility

In this panel, the user can set the visibility of the axes.

## 19.8.2 Data Series

The Data Series is the base for almost all the plot types. To edit the Data Series, right-click on the plot and select *Edit Shown Data...*, or double-click on the area where the curve is displayed. The pages of the edit dialog are described in the following sections.

### 19.8.2.1 Curves

The data in the curves page is entered in the following panels:

#### Data Source

- **Calculation type:** the calculation type is set automatically when the plot is inserted, based on the category selected on the *Insert Plot* dialog. This setting automatically determines other settings for the plot (e.g. scales).
- **Auto-search results:** this is a reference to the currently active results file (*ElmRes*). With this selection the currently used result file of the last calculation is chosen. More information about results objects is available in Section 19.7
- **Select results individually per curve:** this option gives the user the possibility to choose another result file. The result file should be specified in the *Curves* table.

#### Curves

The definition table for the curves is used to specify the results file (optional), element and variable for each curve as well as its representation. Each line in the *Curves* table is used to define one curve.

- The first column allow the visibility of the curve on the plot to be set.
- If the option *Select results individually per curve* is selected, the next column is for the results object from which the data to plot the curve will be read.
- There then follows a column for the power system element, which is selected from the available elements in the results object.

- The next column is for the actual variable for the curve, selected from the variables in the results object, belonging to the selected element.
- The following columns allow the user to specify the style of the individual curve.
- The last column, named *Label*, can be used to add additional information for the curve legend.

**Note:** As well as the option to edit colours individually by double-clicking on the colour square, there are two further options which are offered in the context menu by right-clicking on the colour box. These are:

- **Apply Colour Palette:** this will apply the selected colour palette and change all curve colours accordingly
- **Auto-Assign Subsequent Colours:** if a different standard colour has been selected for the first curve (for example), the remaining curves will be assigned the subsequent colours from same palette.

Only the elements and variables stored in the results file can be plotted. Additional curves can be added by right-clicking and selecting *Insert Rows* or *Append (n) Rows*. Similarly, to delete a marked curve definition from the list, *Delete Rows* can be selected.

Several elements can be selected and *PowerFactory* will automatically insert the corresponding number of rows. In the same way, several variables of the same element can be added in one step by selecting them together.

The following options are available on the context menu of the curves, i.e. when right-clicking on the curve on the plot:

- **Show in Data Manager:** opens the location (within the project) of the object in the Data Manager.
- **Mark in Graphic:** displays the graphical representation of the element in a diagram. If there is a graphical representation of the element in more than one diagram, a list of the diagrams in which the element is available is displayed.
- **Show Gradient:** dynamically shows the difference in x and y values between the clicked point and another point in the curve, and also the gradient ( $dy/dx$ ) and the frequency.
- **Move Curve:** modifies the position of the curve on the curves table (i.e. front or back).

## Plot Features

In this panel, additional features for the curves, including shape, transformation and stacking of curves are defined.

- **Additional curve shapes:** when this option is checked, two additional columns are shown in the Curves table:
  - On the column *Shape*, one of these options can be selected: Curve, Steps, Filled curve, Filled steps and Bars.
  - If one of the filled shapes is selected in the *Shape* column, the filling style can be changed in the column *Fill Style*
- **Data transformation:** this option allows the transformation of the data on the curve according to the following options:
  - Normalisation: the values of the variable can be normalised to a nominal value ( specified in the column *Nom. Value*).
  - Binarisation: transforms the values of the curve into binary data: if a value is bigger than 0.5, it is set to 1, otherwise is to 0.
  - Functions: probabilistic functions can be used to convert the data. More information can be found in Section 19.8.9.6.

---

**Note:** The options for data transformation might vary depending on the calculation type.

---

- **Curve stacking:** this option can be used if more than one curve is shown and will “pile up” the values of the curves as:
  - Values
  - Absolute values
  - Relative values
  - Percentage

### Export Button

When clicking on the **Export...** button on the right of the plot, the *Result Export* command, described in Section 19.7.1, with a time interval set to the time displayed on the plot can be executed to export the result values of the plotted variables.

### Filter Button

It is possible to add additional filters to the curves presented in the plot; it should be noted that when a filter is defined it is applied to all the curves displayed in the plot. The *Curve Filter* command specifies the type of filter applied to the data read from the results object. The following filter settings are available:

- **Disabled:** no filtering will be performed.
- **Moving average:** the filtered curve is the running average of the last n points. The first n-1 points are omitted.
- **Moving balanced average:** the filtered curve is the running average of the last (n-1)/2 points, the current point and the next (n-1)/2 points. This filter thus looks ahead of time. The first and last (n-1)/2 values are omitted; n must be an odd number.
- **Average:** the filtered curve contains the averages of each block of n values; every n-th value is shown.
- **Subsampling:** the filtered curve only contains every n-th value. All other values are omitted.
- **Deviation from initial value:** the curves are shifted, so that for each curve the values are all relative to the first value.

---

**Note:** A curve filter can only be applied at the end of the simulation or measurement. Points added during a simulation or measurement are not filtered.

---

### Processed and Aggregated Results

The curve plots have the option to define a user defined signal. This option allows calculation of additional results based on the arithmetic manipulation of one or more results calculated by *PowerFactory* and recorded in a results object (*ElmRes* ).

A new user defined signal, can be defined by clicking on the **Create...** button on the edit dialog of the curve plot. An example of the calculated result dialog is shown in Figure 19.8.2.

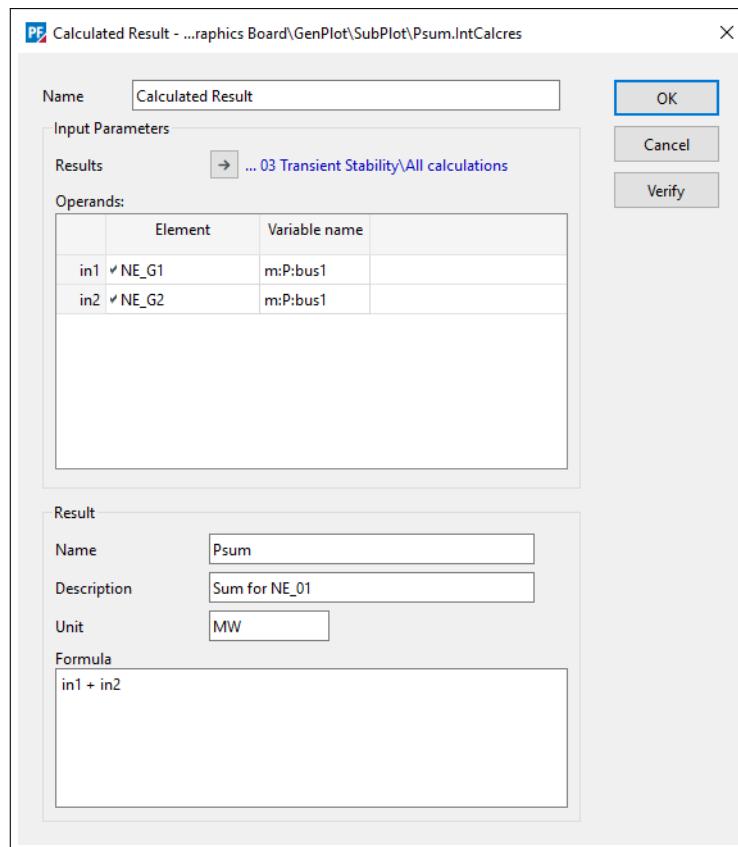


Figure 19.8.2: The calculated results object

The calculated results object dialog includes the following fields:

- **Name:** the name of the calculated results object
- **Input Parameters**
  - Results: defines the results object in which the arithmetic operands are located.
  - Operands: defines the elements and variable names of the operands within the results object. Additional operands can be inserted or appended by *right-click* → *Insert Row(s)* or *Append (n) Row(s)*.
- **Result**
  - Name: defines the name of the user defined curve
  - Description: a free text field for description of the curve
  - Unit: user defined variable unit
  - Formula: DSL expression for arithmetic calculation; operands are defined in accordance with the naming convention in the *Input Parameters* field i.e. in1, in2, in3 etc.

More information about the DSL syntax is available in Section 30.4.

The buttons **Manage...** and **Add to table...** can be used for editing the defined variables and adding them to the *Curves* table.

### 19.8.2.2 Text Format

The label fonts and number formats can be updated using this page as described in Section 19.8.4.3.

### 19.8.2.3 Style and Layout

On the *Style and Layout* page, the following additional display options can be selected:

#### Curve Area Position (mm)

If *Auto-Position Curve Area* is selected, the position of the curve is automatically set to fit the page size. If it is deselected, the border of the plot will appear as shown in Figure 19.8.3 and the user can set the size and position of the plot.

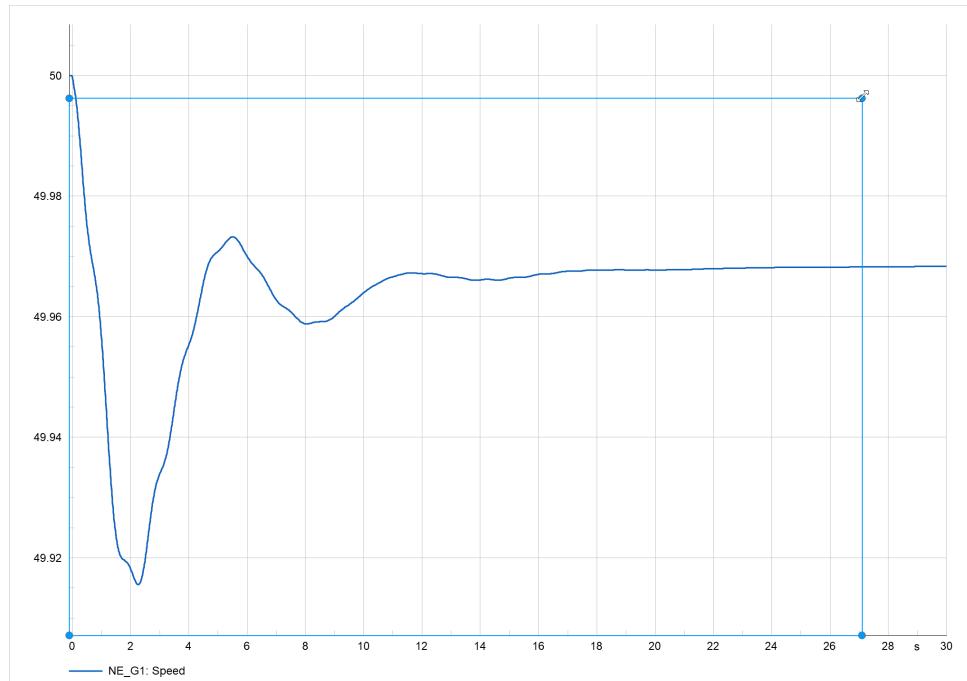


Figure 19.8.3: Resizing a Curve Plot

#### Border and Background

- **Draw border:** to define outside border and colour of the curves area.
- **Fill background:** to define if the curves area should be filled, and select the colour.

#### Colour palette

Link to the colour palette being used. The use of colouring in *PowerFactory* is described in Section 4.7.8. A new colour palette can be selected using the *Select* arrow .

The colour palette can also be changed by right-clicking on the plot area and selecting the option *Apply Colour Palette...* from the context menu or by using the button *Colouring*  in the plots toolbar. When a new colour palette is selected, the colours of the curves will all be changed automatically.

## 19.8.3 Complex Data Definition

The complex data definition is an alternative of the Data Series and is mainly used for the vector plot. In order to edit the Complex Data Definition, right-click on the plot and select *Edit Shown Data...*, or double-click on the area where the curve is displayed. The pages of the edit dialog are described in the following sections.

### 19.8.3.1 Variables

The data in the curves page is entered in the following panels:

#### Data Source

- **Element Variables:** if the plot is inserted by right-clicking on the element, the element and complex variable are automatically defined on the Variables table. Otherwise the element and complex variable can be selected by double clicking on the corresponding fields.
- **Result File:** this option enables the user to add the results file (*ElmRes*). More information about results objects is available in Section 19.7.
  - **Select results individually per curve:** this option gives the user the possibility to choose another result file. The result file should be specified in the *Variables* table.

#### Variables Table

The variables definition table for the curves is used to specify the results file (optional), element and complex variable for each plot as well as its representation. Each line in the *Variables* table is used to define one plot.

- The first column allow the visibility of the curve on the plot to be set.
- If the option *Select results individually per curve* is selected, the next column is for the results object from which the data to plot the curve will be read.
- There then follows a column for the power system element, which is selected from the available elements in the results object.
- The next column is for the complex variable for the plot, belonging to the selected element.
- The following columns allow the user to specify the style of the individual plot.

---

**Note:** The assignation of colours and styles can be done automatically by right-clicking on the colour/style and selecting *Auto Assign Subsequent Colours /Line Styles/Line Widths*.

- 
- The last column, named *Description*, can be used to add additional information for the plot.

Additional curves can be added by right-clicking and selecting *Insert Rows* or *Append (n) Rows*. Similarly, to delete a marked curve definition from the list, *Delete Rows* can be selected.

Several elements can be selected and *PowerFactory* will automatically insert the corresponding number of rows. In the same way, several variables of the same element can be added in one step by selecting them together.

#### Time Point Selection

Once the result file is added to the plot, *Time Point Selection* option can be used to investigate the results at different points in time. Furthermore, the cursor can be added to the plots in order to keep a track of the result variables in both the vector and dynamic simulation plots simultaneously.

#### Plot Features

- **Vector labels:** the vector labels provides the user the possibility to update the labels of the vectors in the plot according to the requirements. If *None* is selected then no labels will be displayed. The *Automatic* option automatically detects the labels based on the complex variables representation. Moreover, the user can explicitly set the labels in polar or cartesian system using *Coordinates*, *Polar* or *Coordinates*, *Cartesian* respectively. The *Phase Only* option can be used to display the labels based on the respective phases of each vectors.

- **Draw arrow heads:** the arrow heads can be drawn on the vectors by enabling this option.
- **Vector transformation:** using this feature, it is possible to transform a complex variable using another variable or a constant. As required, various operations such as addition, subtraction, multiplication and division in addition to shift can be applied to the complex variable. The information about these operations has to be provided in the *Variables* table. Also, the information about the operation being applied to the complex vector is visible in the plot legend. Figure 19.8.4 shows an example of the implementation of vector transformation feature in the edit dialog of the plot.

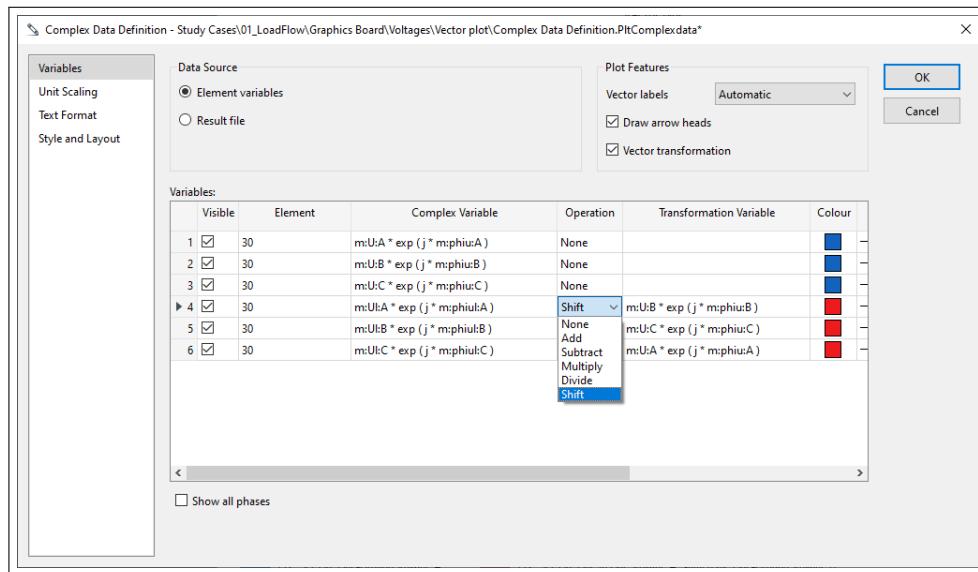


Figure 19.8.4: Different operations for vector transformation

### 19.8.3.2 Unit Scaling

This page allows the user to set the scaling of the units of multiple vectors being displayed in the plot. There is a possibility to determine the unit scales automatically by enabling the corresponding option. And if required, the users can define their own scales for the units to normalise the individual vectors by disabling the *Determine unit scales automatically* option.

### 19.8.3.3 Text Format

The label fonts and number formats can be updated using this page as described in Section 19.8.4.3.

### 19.8.3.4 Style and Layout

On the *Style and Layout* page, the following additional display options can be selected as described in Section 19.8.2.3.

## 19.8.4 Axes and Gridlines

Each of the axes can be edited either by double click or by *right-click* → *Edit Axis*. The plot axis dialog options are described in this section.

#### 19.8.4.1 Scale

The options presented in the *Scale* page depend on whether the edited axis is a x- or a y-axis.

##### Axis mode

This option is only available for the x-axis and is automatically set depending on the plot type. Alternatively, can be selected by the user. The options are:

- Default: depends on the type of simulation and the results object created during the previous simulation.
- Time: the x-axis is set to time and an additional field for the time unit is displayed.
- Date and time: used by default in quasi dynamic simulation plots, the format for the date and hours and the time period can be specified.
- Frequency: the x-axis is set to frequency and an additional field for the frequency unit is displayed.
- Discrete (net elements): used in bar plots, where network elements are shown in the x-axis

##### Scale Type

For the x-axis, for all the modes, except the *Date and time*, the scale can be set to linear or logarithmic. The y-axis can additionally be set to dB.

The option *Reversed* will change the direction of the scale.

##### Range

The *Minimum* and *Maximum* limits of the axes can be set manually.

##### Scale to Contents

The scaling of the axis is done according to the data available, for the x-axis is by default 0 % and for the y-axis 10 %. The option *Limit minimum to origin if possible* will set the minimum to zero. The option *Scale when calculation data changes* will automatically update the scale of the axis after the calculation is executed.

#### 19.8.4.2 Axis Labelling

On this page, the grid lines and styles are defined.

##### Tick Mark Positions

- Step size in data space: *PowerFactory* tries to find the ticks that would best fit with the data. For this, a reference value can be set to force the origin.
- Fixed number of tick marks: the number of ticks a fixed number of subdivisions between the minimum and maximum limits.

For both options, the option *Determine tick positions automatically* can be selected. This option determines the number of ticks depending on what would look best on the plot page. If the option is unchecked, the user can select either the step size (for the step size in data space option) or the number of ticks (for the fixed number of tick marks option).

## Line Style

Options to alter the width, line style and colour of axis, ticks and grid lines.

### 19.8.4.3 Text Format

In the *Font Style* panel, the following options can be set:

- Font: font size, type and effects.
- Label colour and offset: the colour of the axis values and the distance between the axis and the text.
- Show unit: to display the unit of the axis.

In the *Number Format* panel, the following options can be set:

- Format: the user can select between concise, fixed decimals and scientific.
- Digits: if the option *concise* is selected, the user can set the maximum number of digits to be displayed.
- Decimals: if the option *fixed decimals* or *scientific* is selected, the user can set the number of decimal places to be shown.
- Exponent character: the user can select between E and e for the *scientific* option.

## 19.8.5 Plot Legend

The edit dialog of the legend can be accessed either by double-clicking on it or by *right-click* → *Edit Legend*. These are the options available:

### Show legend

Defines the visibility of the legend.

### Positioning and Layout

- **Position:** the position of the legend. Options are:
  - Floating
  - Top left
  - Top centre
  - Top right
  - Bottom left
  - Bottom centre
  - Bottom right
  - Left
  - Right
- **Layout:** the layout of the legend. Options are:
  - Automatic: based on the size of the plot and number of variables
  - Columns: organise in columns; an additional field for the number of columns is displayed
  - Horizontal
- **Margin:** the distance in mm to the axis/curve area.
- **Padding:** defines the size of the box around the legend, particularly useful when defining a background colour for the legend.

- **Include only drawn curves:** only the curves with results will be shown on the legend, otherwise the legend of those curves is shown greyed-out.
- **Reverse item order:** change the order of the legend items.

### Text Format

- **Font:** font size, type and effects.
- **Always show units:** displays the units of the variables.
- **Variable description:** the description of all the variables can be selected. The options are:
  - Long
  - Short
  - Name
- **Line spacing:** defines the spacing of the legend text.

### Border and Background

- **Draw border:** to define outside border of the legend and the colour of it.
- **Fill background:** to define if the legend should be filled and the colour to be used.

### Floating position

If the position of the legend is set to *floating*, the location of the legend in mm can be set in this field.

## 19.8.6 Plots Toolbar

There are numerous tools which help the user interpret and analyse data and calculation results. Most of the tools are accessible directly through plot toolbar, which is displayed when a plot is inserted. Each of the icons of the plot toolbar, shown in Figure 19.8.5 and the additional context menu options are described in the following sections.



Figure 19.8.5: Plots Toolbar

### 19.8.6.1 Insert Plot

The icon inserts a plot in the existing page. Clicking on this button opens the *Insert Plot* dialog, described in Section 19.8.

### 19.8.6.2 Edit Plots on Page

The icon opens the dialog for defining curves of several plots. If the variables of only one plot are to be changed, it is suggested to edit the dialog of the plot itself by double-clicking it. This procedure is more convenient.

This dialog gives a very good overview over the diagrams on the plot page and the variables, axis and curve styles. Figure 19.8.6 shows an example of the dialog.

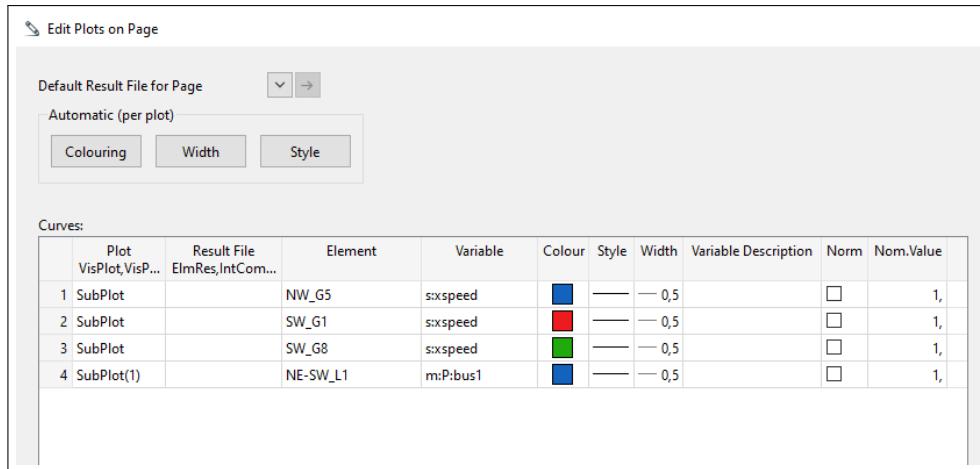


Figure 19.8.6: Editing all plots on the plot page

Each line of the table named *Curves* defines a variable shown on the panel. The variables definition applies to the plot shown in the first column. When the dialog is opened the plots are sorted from left to right and from top to bottom and are numbered accordingly.

All data and settings of each variable are displayed in the table, and the columns are used exactly like the columns in the table of a plot.

The *Default Result File for Page* can be used to set the default result file for all the plots on the page.

The buttons **Colouring**, **Width** and **Style** set those values automatically per plot.

The **Colouring** button will set the colours according to the “*PowerFactory Standard*” palette regardless of which palette the user has selected as a default for plots.

#### 19.8.6.3 View and Select Commands

- **Rebuild:** updates the currently visible page by updating the drawing from the database.
- **Zoom In:** changes the cursor to a magnifying glass. The mouse can then be clicked and dragged to select a rectangular area of the plot to be zoomed.
- **Hand Tool:** if a zoom is applied, can be used to pan the plot.
- **Zoom All:** zooms to the page extends.
- **Zoom Level:** zooms to a custom or pre-defined level.

---

**Note:** Ctrl+ mouse scrolling can be used to zoom in and out

---

#### 19.8.6.4 Automatic Arrangement Commands

A plot's size and position is usually set automatically. There are two different modes for automatically arranging the plots in the plot page:

- *Arrange plots on top of each other*
- *Arrange plots automatically*

The modes can easily be changed by pressing the one or the other button. The relative positions of plots can also easily be changed: mark the plot by clicking it, then 'drag' the plot across another plot.

**Note:** This option of exchanging the plots by dragging is only possible when one of the arrangement buttons are active. If you deactivate both buttons by unselecting them in the toolbar, the plots can freely be moved by dragging them on the panel

#### 19.8.6.5 Scale Buttons

- **Scale x-axes automatically:** scales all the x-axes according to the start and end of the results file.
- **Scale y-axes automatically:** scales all the y-axes according to the maximum and minimum values of the variables in the results file.
- **Zoom x-axis:** zooms in a certain range of the x-axis (if the axis is shared, the zoom is applied to all plots sharing the axis definition).
- **Zoom y-axis:** zooms in a certain range of the y-axis (if the axis is shared, the zoom is applied to all plots sharing the axis definition).
- **Move x-scale:** moves the position of the x-axis (is applied to all plots sharing the axis definition).
- **Stretch/compress x-scale:** modifies the x-axis scales in order to compress or stretch the shown curve (is applied to all plots sharing the axis definition).

**Note:** The scale buttons are inactive if there are no plots shown at all or if the x or y axes cannot be scaled automatically.

#### 19.8.6.6 Add Curve Label

A number of *Curve Labels* tools are available in the Drawing Toolbox, which will be displayed if the user clicks on the *Add Curve Labels* button .

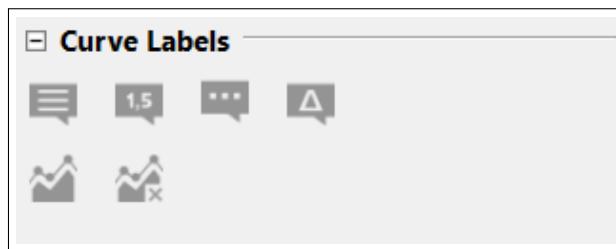


Figure 19.8.7: Curve Labels Tools

There are different styles of labels available for labelling curves and graphics. Setting labels is possible in most of the different plots, although some of the labels are not available in all plot types. Labels are all created in the same way.

Most of the label buttons are only active after clicking on the curve. After selecting the appropriate label from the sub-option of label, a rubber band from the cross to the mouse is shown. A click with the left mouse button sets the label, the right mouse button cancels. The following labels are available:

- **Text Label:** displays user-defined text above and below a line connected to the curve.
- **Label with Curve Value:** displays the x/y coordinates of selected point.
- **Label with Definable Format:** displays the name of the element.
- **Gradient Label:** displays the difference in x and y values (dx and dy) between two points, and also the gradient ( $dy/dx$ ) and the value of  $1/dx$ .
- **Statistic Label:** helps to analyse a curve, by labelling, for example, its extrema.
- **Delete Statistic Label:** delete existing statistic label.

---

**Note:** The font and colour of the text displayed on the label is defined on the *Text Format* page of the Data Series dialog. See Section [19.8.2.2](#).

---

### Text, Curve Value and Gradient labels

The text, value and gradient labels are defined using the same object type: *VisValue*.

Figure [19.8.8](#) shows the three different labels.

The *VisValue* edit dialog contains the following fields:

- **Value:** displays the connected curve position of the label. For curve value labels, this position is displayed automatically as label text. "x" displays the x axis value and "y" the y axis value. If the *Snap to curve* checkbox is active, the references lines and label values will automatically change after any change in the data or recalculation. For plots showing a trajectory, for example an X-Y plot of a time domain simulation, a third value "Time" is displayed on the curve value labels.
- **Show marker:** as shown in Figure [19.8.8](#), a variety of markers are available to mark the point on the curve to which the labels are referenced.
- **Show tolerance:** for curve value labels, a user-defined tolerance and fill style can be added to highlight an area surrounding the point on the curve.
- **Show label:** if selected, the text on top and bottom will be displayed depending on the label type.
- **Text on top and on bottom:** text written above and below the horizontal line. By default, these fields are automatically filled with keywords (placeholders) that correspond to the displayed variables. The following are examples of some of the most commonly used placeholders:
  - DisplayName: {displayname}
  - Value (either cartesian or polar depending on coordinate representation of plot): {value}
  - Value X: {valuex}
  - Value Y: {valuey}
  - Value Magnitude: {valuemag}
  - Value Angle: {valueangle}
  - Timepoint: {timepoint}
  - Gradient: {grad}
- **Alignment:** defines the horizontal alignment of the displayed text.
- **Delete on new calculation:** this option, if activated, will remove the existing label from the plot if a calculation is executed again.

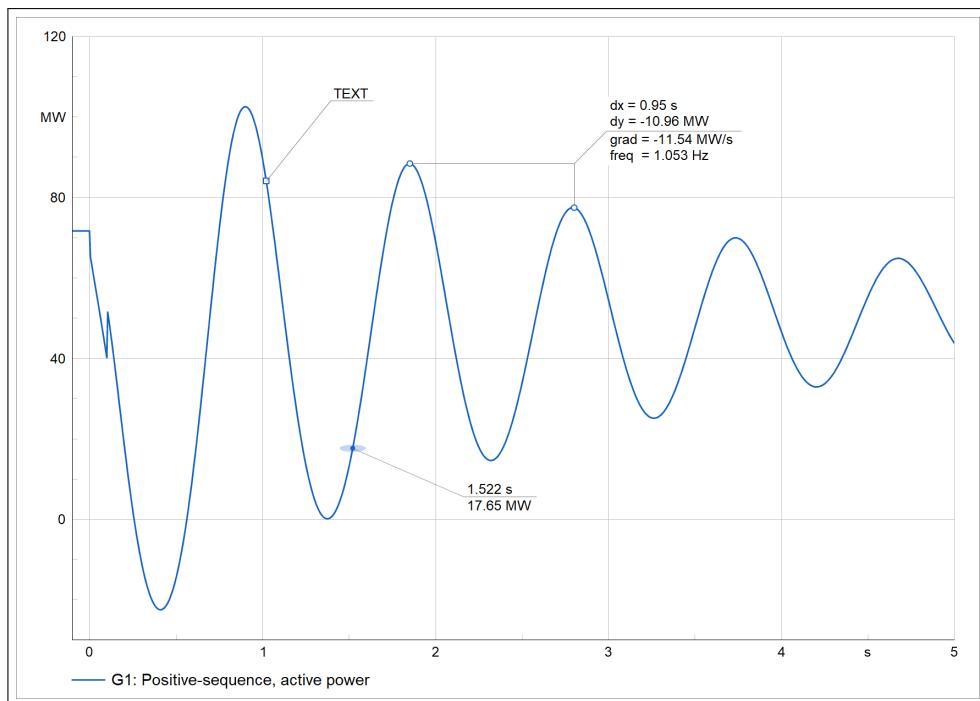


Figure 19.8.8: Text, Value and Gradient labels

### Label with Definable Format

The format-label displays text printed using a form. It is typically used to show the name of the object whose variable is shown in the curve. It is useful when several curves with the same colour are plotted.

The form is different for each type of diagram. It is either defined locally per label or defined for all diagrams of the same type in the activated project. The format-label dialog is shown in the following figure.

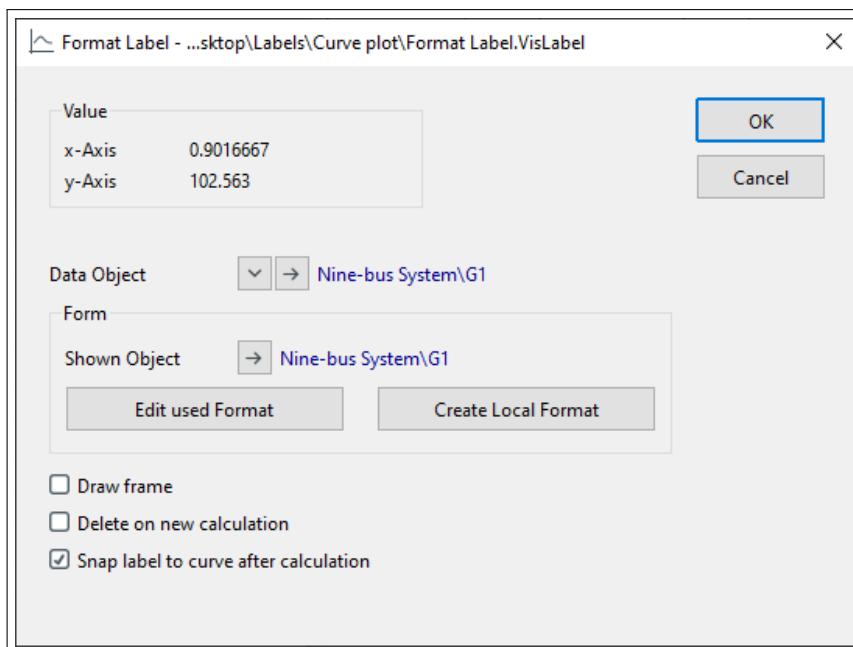


Figure 19.8.9: Edit dialog of the label with definable format

- **Value:** displays the connected curve position of the label. *x-Axis* displays the x axis value and

y-Axis the y axis value.

- **Data Object:** reference to the object of which the plotted curve parameter is derived. If *Data Object* is not set the label itself is taken as the *Shown Object*.
- **Shown Object:** object output by the form, see *Data Object* described above.
- **Edit Used Format:** shows the used *Format Manager*. The used format is either the local format or the one defined for all plots of the same type in the active project.
- **Set Default Format:** sets the used format as the one used for all plots of the same type in the active project.
- **Delete on new calculation:** this option, if activated, will remove the existing label from the plot if a calculation is executed again.
- **Snap label to curve after calculation:** if selected, the references lines will automatically snap to the new curve after any change in the data or recalculation.

### Statistic Labels

The statistic label function provides the possibility to label the following values of the curves:

- Minimum or Maximum in the visible area of the plot
- Global Minima or Maxima
- Local Minima or Maxima
- Global Average
- Average of the visible area of the plot
- Integral of the visible area of the plot
- Energy (Integral calculation available specifically for the load duration plot)
- Global statistical mean value of the probabilistic data
- Global confidence interval for mean value of the probabilistic data

To remove statistic labels from a curve, the button *Delete Statistic Labels*  can be used.

#### 19.8.6.7 Multi-Curve Tracking

Enabling “Multi-Curve Tracking” mode  means that when the mouse is hovered over a point in the plot, a tooltip will appear, giving information about the current value on the x-axis as well as all y-axis values for the curves. The tooltip follows the mouse while it is moved over the plot and therefore makes it very easy to see individual results without zooming in, even if lines are on top of each other, or very close together.

Additionally, once the curve is clicked, even if this option is not selected, a label with the values of the curve at the position of the mouse is shown. The keyboard arrows can be used to move along the curve.

#### 19.8.6.8 Frequency Analysis

When the button  is clicked, the frequency of an area of the curve can be analysed. More information about the frequency analysis is available in Section [29.14](#).

### 19.8.6.9 Cursors

The buttons  and  can be used to add a vertical line in all the plots of the page. These cursor can be used to compare plots instead of defining separate x-constants in every plots.

### 19.8.6.10 Title Block

The icon  shows or hides the title block of the page.

The title can be defined or changed by double-clicking on it.

For details about the settings of the title object refer to Chapter 10: Network Graphics.

### 19.8.6.11 Edit Plot Page

Whenever a plot is inserted, a Plot Page is automatically created, the Plot Page being one of the possible page types in a Desktop. The edit dialog can be opened by clicking on the *Edit Plot Page* () icon and the pages are described below.

#### Basic Data

- *Name*: name of the page, also modifiable by right-clicking on the tab and selecting → *Rename Page*.
- *Page auto layout*: defines the layout of the plots on the page. This option is the same as using the plot arrangement buttons, described in Section 19.8.6.4.
- *Curve area alignment*: aligns the plots on the page, options are:
  - Off
  - Shared axes only
  - All plots on page
- *Colour palette*: link to the colour palette being used. A new colour palette can be selected using the *Select* arrow .
- *Style*: link to the user-defined style used in the plot page. A new style, if already defined, can be selected using the *Select* arrow . Information about how to create user-specific styles is available in Section 19.8.8.
- *Results*: to define the Results File object used for all the plots in the plot page. The result column of the plots need not be set for most calculations: the plot itself will look for the results element to display automatically.

#### Advanced

The settings on this page show the visibility of the page, which tab group it is, and whether it is the currently selected window within that group. Note that although these settings can be changed manually from the dialog, such changes will not take effect until the desktop is reopened, e.g. by reactivation of the study case.

### 19.8.6.12 Page Format

The page format is modified using the  icon. In the Page Format, the drawing size and the page format are defined. The plot page uses the page format set in the desktop.

In addition a local page format can be created for each Plot Page by selecting the option *Create local Page Format* from the context menu.

#### 19.8.6.13 Colouring

The button *Colouring*  can be used to select the used colouring palette on the Plot Page. When a new colour palette is selected, the colours of the curves will all be changed automatically.

The use of colouring in *PowerFactory* is described in Section [4.7.8](#).

#### 19.8.6.14 Export Diagram

The *Export Diagram* icon  opens a dialog offering options for exporting the graphic. A range of file formats is supported.

#### 19.8.6.15 Print

The *Print Diagram* icon  opens the print preview page. The printer and margins to be used can be selected in this dialog.

### 19.8.7 Context Menu Tools

As well as the tools of the plot toolbar, the following additional tools are available via the context menu, displayed by right-clicking on the plot.

#### Copy Plot to New Page

This option will create a new Plot Page containing only the selected plot.

#### Add Constant

The “add constant” is used to display either a x/y point in the plot or a straight line, which can be the y-values for a constant x-quantity, the x-values for a constant y-quantity or a line with a slope in the form of  $y = mx + b$ .

For the x/y value constant, the available options are equivalent to those for the [Curve Value Label](#). For the line constant labels, the appearance can be modified using the following settings:

- **Value:** defines the constant value, either X or Y. The dialog shows if either an X or Y is set. Also the actual position of the cross will be shown as an x- or y-value. It is not possible to change a constant X into a constant Y label other than by removing the old label and creating the new one.
- **Colour:** specifies the colour of the line and the labels/intersections.
- **Show:** changes the representation of the constant label as follows:
  - *Line Only*: displays only the solid line and the related label.
  - *Line with Intersections*: shows a solid line including label and indicates the values when intersections with the curves of the plot.
  - *Short Line Only (Left/Right/Top/Bottom)*: indicates the constant value at the bottom/top respectively at the right/left side of the plot.
  - *Short Line/Intersection (Left/Right/Top/Bottom)*: indicates the constant value at the bottom/top respectively at the right/left side of the plot and the intersections with curves.

- *Intersections Only*: shows only the intersection points with the curves.
- **Linestyle and Width**: specifies the line style and line width for the line shown. Invisible if *Show Values* is set to *Intersections Only*.
- **Label**: either the default value (name of the variable in the axis) or defined by the user.
- **Position**: defines the position of the constant value label as follows:
  - *None*: displays no label at all.
  - *Outside of Diagram*: creates the label between the border of the plot and the diagram area. Labels of constant x values are created above the diagram area, labels of constant y values to the right of the diagram area.
  - *Above Line (right)*: shows a label above the line if y is constant; the label will be on the right hand side.
  - *Below Line (left)*: shows the label below the line on the left hand side.
  - *Left of Line (top)*: shows a label on the left side of the line if x is constant; the label will be on the top end.
  - *Right of Line (bottom)*: shows the label right of the line on the bottom end.
- **Show tolerance**: a user-defined tolerance and fill style can be added to highlight an area around the line constant.
- **m and b**: if the constant line is a *Line Equation*, the m and b values should be defined in these fields.

#### Export Shown Data

This option opens the *Result Export* command, described in Section 19.7.1, with the user-defined interval on the Advanced Options page of the command defined by the time range of the data currently visible in the plot.

If the option is selected when right-clicking on a plot, the variables correspond to the curves displayed on that plot only. Right-clicking on the grey area (plot page) will open the export dialog including all the variables available on the plot page (i.e. variables from all the plots).

#### 19.8.8 User-Defined Styles

The user-defined styles are stored in a folder called *Plot Style Templates* within the *Settings* folder of the active project. A new style is created by right-clicking on the plot or on the plot page and selecting *Style → Save Style as...* .

The style can be changed by right-clicking on the plot or on the plot page and selecting *Change Style*

Inside the user-defined style, a template of the plot page, described in Section 19.8.1, is stored and can be modified.

#### 19.8.9 Plot Types

All the available plots are listed below, grouped by category, and described either in the following sections or in the corresponding chapter (for calculation-specific plots).

##### 1. Simulation RMS/EMT

- Curve plot (Section 19.8.9.1)
- Curve plot with two y-axes (Section 19.8.9.2)
- Vector plot (Section 19.8.9.10)

- XY curve plot (Section 19.8.9.3)

## 2. Quasi-Dynamic Simulation

- Curve plot (Section 19.8.9.1)
- Complete generation (Section 19.8.9.8)
- Plant categories (Section 19.8.9.7)
- Renewable and fossil (Section 19.8.9.9)
- Duration curve (Section 19.8.9.6)
- Curve plot with two y-axes (Section 19.8.9.2)
- XY curve plot (Section 19.8.9.3)

## 3. Unit Commitment

- Curve plot (Section 19.8.9.1)
- Complete generation (Section 19.8.9.8)
- Plant categories (Section 19.8.9.7)
- Renewable and fossil (Section 19.8.9.9)
- Duration curve (Section 19.8.9.6)
- Curve plot with two y-axes (Section 19.8.9.2)
- XY curve plot (Section 19.8.9.3)

## 4. Modal/Eigenvalue Analysis

- Eigenvalue plot (Section 32.4.2.1)
- Mode bar plot (Section 32.4.2.2)
- Mode polar plot (Section 32.4.2.3)

## 5. Power Quality and Harmonic Analysis

- Curve plot (Section 19.8.9.1)
- Harmonic distortion (Section 36.2.5.2)
- Curve plot with two y-axes (Section 19.8.9.2)
- Waveform plot (Section 36.2.5.3)
- XY curve plot (Section 19.8.9.3)

## 6. Frequency Sweep

- Curve plot (Section 19.8.9.1)
- Curve plot with two y-axes (Section 19.8.9.2)
- XY curve plot (Section 19.8.9.3)

## 7. Probabilistic Analysis

- Convergence of statistics (Section 45.3.8.3)
- Distribution estimation (Section 45.3.8.5)
- Distribution fitting (Section 45.3.8.6)
- Correlation plot (Section 45.3.8.4)

## 8. Protection and Arc-Flash Analysis

- Current comparison differential plot (Section 33.10.1)
- Curve-input (Section 19.8.9.11)
- Phase comparison differential plot (Section 33.10.2)
- R-X plot (Section 33.6)
- Relay operational limits plot (P-Q diagram) (Section 33.7)
- Short-circuit sweep plot (Section 33.12)
- Time-distance plot (Section 33.8)

- Time-overcurrent plot (Section [33.4](#))

## 9. Power Park Energy Analysis

- Curve plot (time series)
- Generation curves (basic analysis)
- Losses curves (basic analysis)
- Wind speed cumulative probability (basic analysis)
- Wind speed probability (basic analysis)
- Annual load duration curve

## 10. Optimal Equipment Placement

- Curve plot (Section [19.8.9.1](#))
- Complete generation (Section [19.8.9.8](#))
- Plant categories (Section [19.8.9.7](#))
- Renewable and fossil (Section [19.8.9.9](#))
- Duration curve (Section [19.8.9.6](#))

## 11. Flow Decomposition

- Curve plot (Section [19.8.9.1](#))
- Curve plot with two y-axes (Section [19.8.9.2](#))
- XY curve plot (Section [19.8.9.3](#))

## 12. General

- Curve plot (Section [19.8.9.1](#))
- Bar plot (Section [19.8.9.4](#))
- Binary bar plot (Section [19.8.9.5](#))
- Curve plot with two y-axes (Section [19.8.9.2](#))
- Curve-input (Section [19.8.9.11](#))
- Vector plot (Section [19.8.9.10](#))
- XY curve plot (Section [19.8.9.3](#))

## 13. Virtual Instruments (Section [19.8.9.12](#))

- Digital display
- Horizontal scale
- Measurement instrument
- Vertical scale

## 14. Others

- Button
- Command button
- Network graphic
- Picture
- Schematic path
- Text
- Voltage profile (along feeder) (Section [19.8.9.13](#))
- Voltage sag plot

### 19.8.9.1 Curve Plot

Curve plots are the “basic” diagrams and are typically used to display one or more plotted curves from the results of a simulation (EMT, RMS, Quasi-dynamic). The Data Series of this plot type is described in Section 19.8.2. The following figure shows an example of a curve plot.

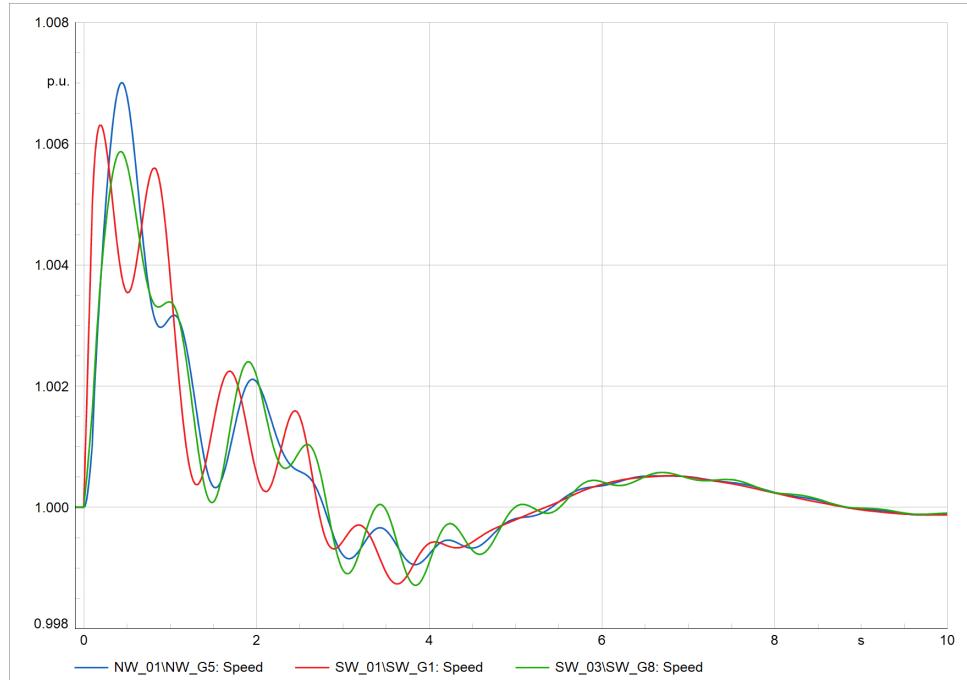


Figure 19.8.10: Curve Plot

### 19.8.9.2 Curve Plot With Two Y-Axes

A curve plot with two y-axes is typically used for displaying together quantities which have very different scales. The Data Series of this plot type is as described in Section 19.8.2, the main difference being the additional y-axis.

The following figure shows an example of a curve plot with two y-axes.

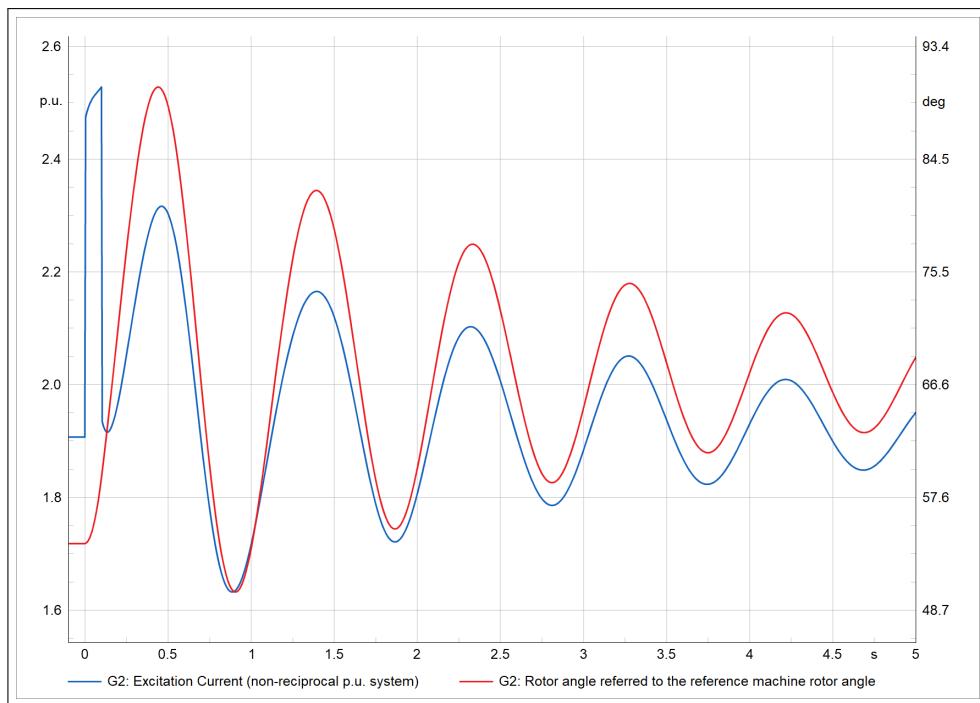


Figure 19.8.11: Curve Plot (two y-axes)

### 19.8.9.3 XY Curve Plot

This plot shows one variable plotted against a second variable. The two variables can be completely independent from each other and do not have to belong to the same element. The differences in the Data Series of this plot type compared with the one described in Section 19.8.2 are:

- Additional columns for the element and variable on the x-axis.
- A specific time range to be displayed can be defined.
- Direction arrows can be drawn on the curve.
- Is it not possible to add processed and aggregated results.

The following figure shows an example of a XY curve plot.

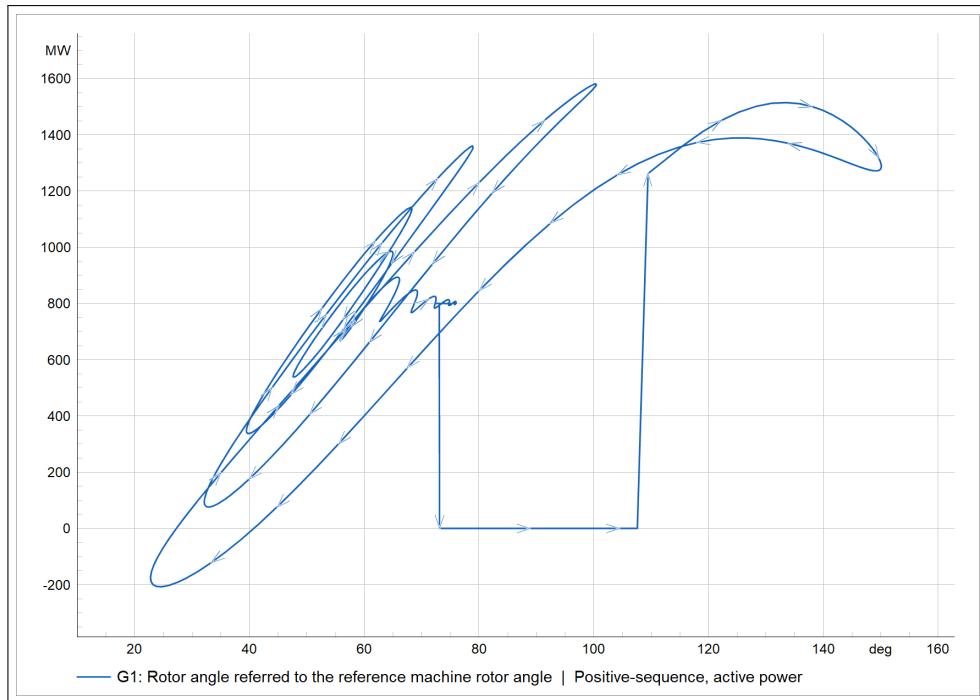


Figure 19.8.12: XY Curve Plot

#### 19.8.9.4 Bar Plot

The bar plot is used to visualise steady-state values such as voltages, currents and power. A bar plot can be inserted after a calculation has been executed using the *Insert Plot* dialog or directly by right-clicking on an element(s) and selecting the option *Plots → Bar-Diagram→ "variable"*. An example of the bar plot is shown in the following figure.

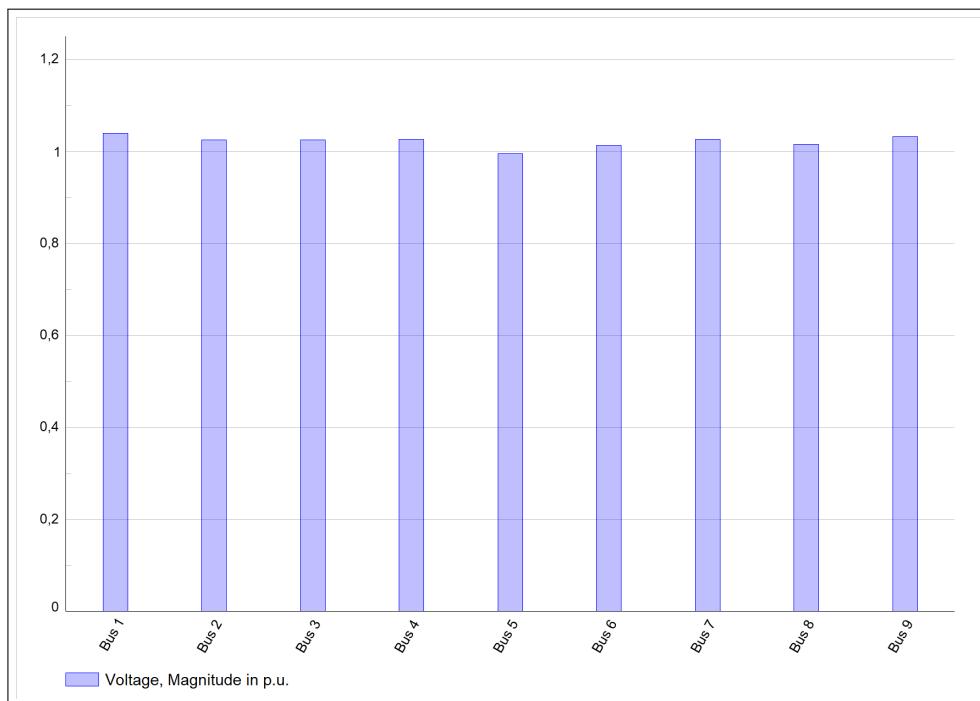


Figure 19.8.13: Bar Plot

The *Style and Layout* page of the edit dialog of the bar plot is the same as that of the Data Series, described in Section 19.8.2.3. The *Curves* page is different, as described below.

### x-Axis Elements

The *x-Axis Elements* table consists of the elements whose variables are shown in the plot. If the plot is inserted by right-clicking on the element, the elements are automatically defined. Otherwise the element can be selected by double clicking on the corresponding field.

### Curves

On the *Curves* table, the user can select the variables to be displayed. The default shape is *Bars* and the fill style can be specified by the user.

#### 19.8.9.5 Binary Bar Plot

The Binary Bar Plot can be used to illustrate digital signals. The Data Series is as the one described in Section 19.8.2 and the following options are set by default:

- Plot Features:
  - Additional curve shapes
  - Data transformation: Binarisation
  - Curve stacking: Values
- Curves table:
  - Shape: Filled steps

An example of the binary bar plot, together with a curve plot, is shown in Figure 19.8.14.

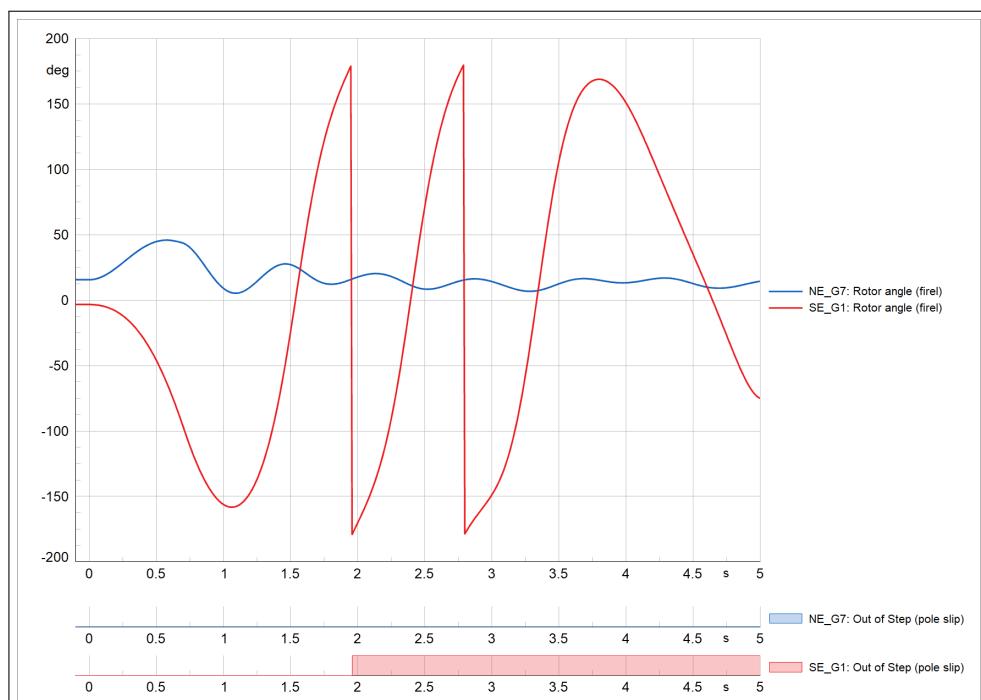


Figure 19.8.14: Binary Bar Plot

### 19.8.9.6 Duration Curve

With the *Duration curve* plot it is possible to evaluate the utilisation of specific elements such as transformers or different power plants over a given time period. After a *Quasi-Dynamic Simulation* is executed, for example, the loading of a transformer can be illustrated.

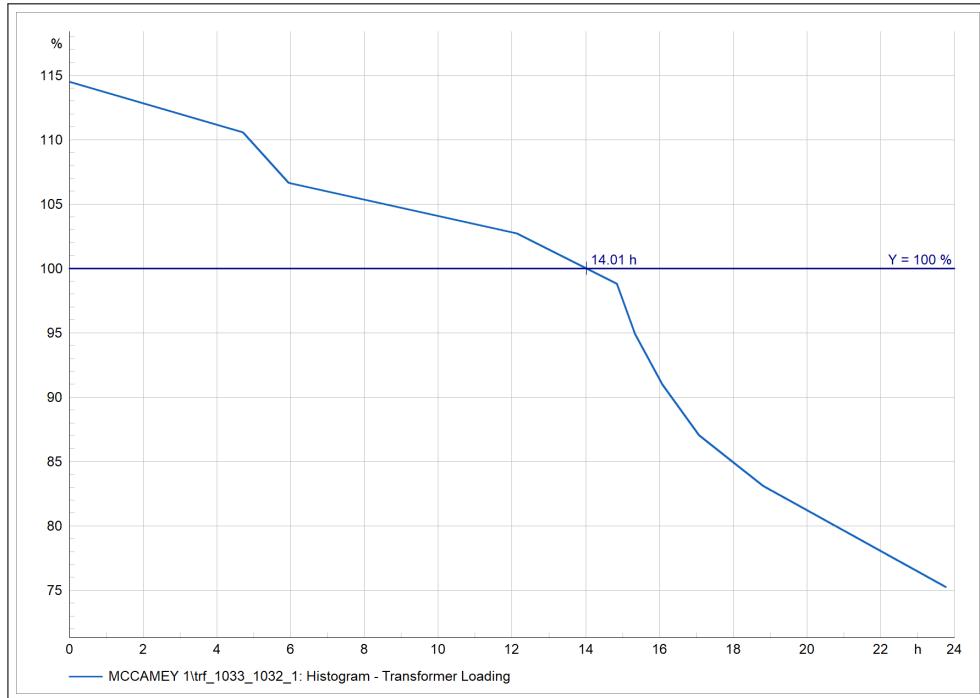


Figure 19.8.15: Duration curve of the loading of a transformer over the duration of a day

The edit dialog of the Data Series of this plot type is the same as the one described in Section 19.8.2. The following options are set by default:

- Plot Features:
  - Additional curve shapes
  - Data transformation: Functions
- Curves table:
  - Shape: Steps
  - Column *Function* set. See description below.

Double clicking on the column *Function* opens the *Distribution estimation* dialog, which contains the following options:

- For the **Method** that is used to determine the duration curve, *Histogram* or *Bootstrapping* can be selected. A detailed description on the methods can be found in Section 45.2.5.
- The **Curve** can be defined through several different functions: *Cumulative distribution function*, *Probability density function*, *Quantile function* and *Mirrored quantile function*. More details can be found in Section 45.2.1.1.
- The different methods that are used for the **Estimation** are *User-defined*, *Automatic parameter deduction*, *Rice Rule*, *Freedman-Diaconis choice* and *Scotts Rule*. Detailed descriptions of the estimation methods can be found in Section 45.2.5.2.

### 19.8.9.7 Plant Categories

The *Plant categories* plot is one of the energy plots and used for displaying the proportion of all defined plant categories from the total generation in a network.

An example of the plot is shown here:

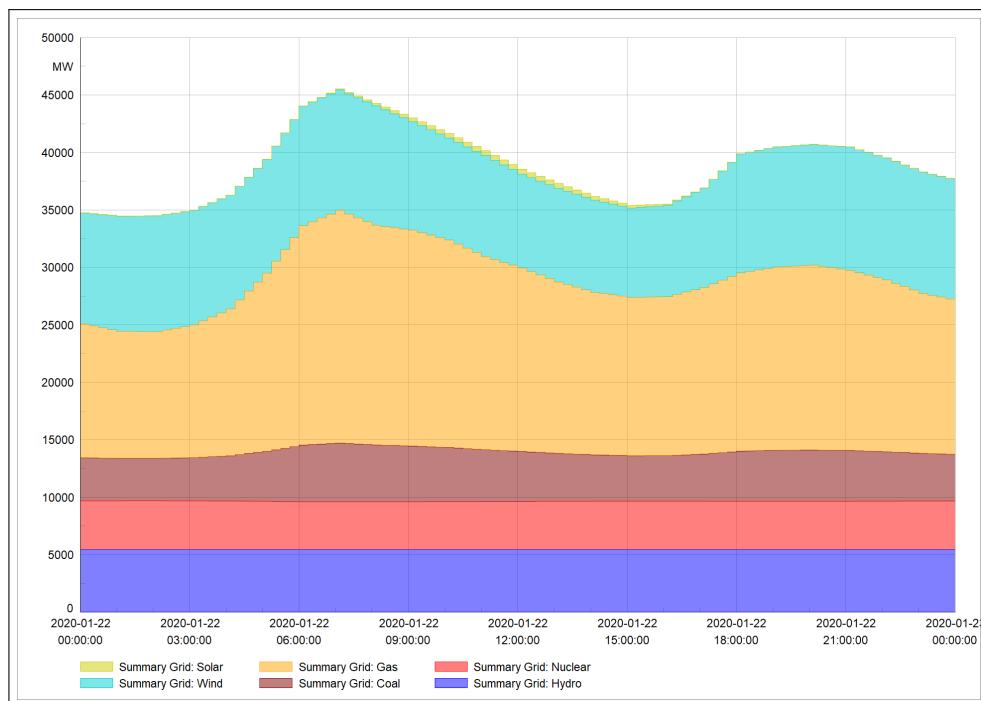


Figure 19.8.16: Total generation in a grid divided into different plant categories

The edit dialog of the Data Series of this plot type is the same as the described in Section 19.8.2. The following options are set by default:

- Plot Features:
  - Additional curve shapes
  - Data transformation: Energy plot
  - Curve stacking: Values
- Curves table:
  - Shape: Filled steps

The plant categories plot can be customised as follows:

- In the *Curves* table all available plant categories are shown by default; however, the user can use the checkbox on the *Visible* column to show only some categories.
- In the column *Group*, the defined grouping object for each category is used. By default the *Summary Grid* is selected to represent all calculated generation in the network. It has to be noted that different grouping objects can be defined for the same category, but no element is allowed to be defined in several grouping objects.
- The order of the stacking of the curves is defined as the first entry is displayed in the plot on the bottom, while the last entry that is defined is displayed on the top. This however, can be modified on the plot by right-clicking on a curve and selecting *Move Curve → Upwards/Downwards*.
- Changing the stacking to *Percentage* or *Relative values* standardises the values to 100 % or 1 to illustrate the share of the shown variables.

The buttons **Filter...** and **Export...**, available for the curve plots, are not available for the energy plots.

### 19.8.9.8 Complete Generation

The *Complete generation* plot is one of the energy plots and displays the total generation in a network. Therefore it can be used to show, for example, the total generation in different areas of the grid, as shown here:

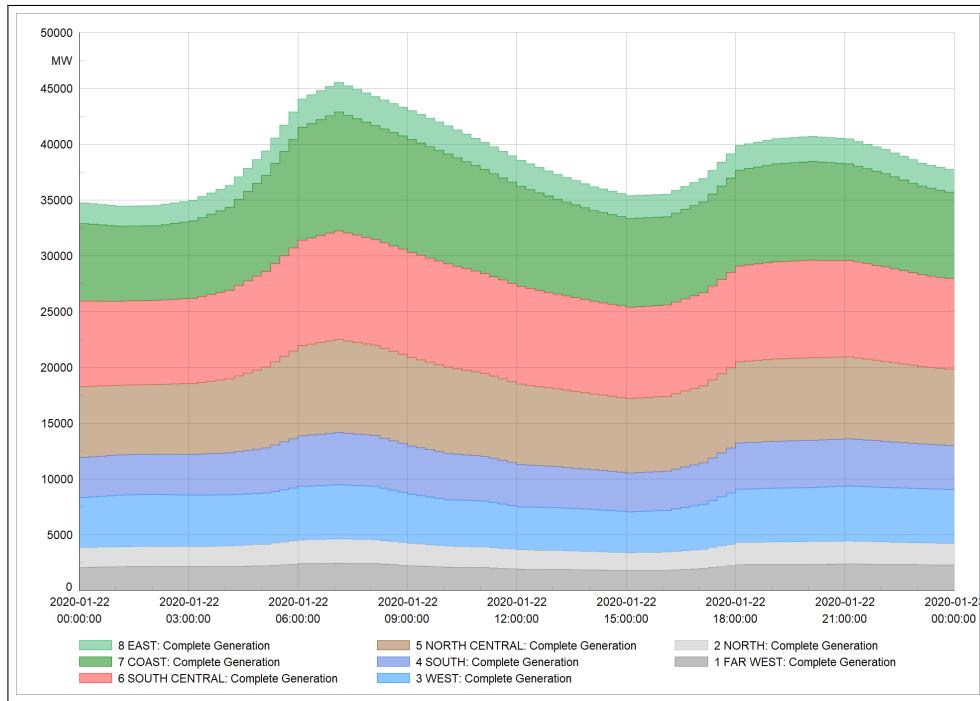


Figure 19.8.17: Total generation in a grid divided into different areas

The pages on the edit dialog of the plot are the same as for the energy plot *Plant categories* described in Section 19.8.9.7.

The only difference is the default entry for the **Curves**, since all plant categories are summed up as *Group Generators* for the whole summary grid.

### 19.8.9.9 Renewable and Fossil

The *Renewable and fossil* plot is one of the energy plots and displays the renewable and fossil shares of the generation in a network.

An example of the plot is shown here:

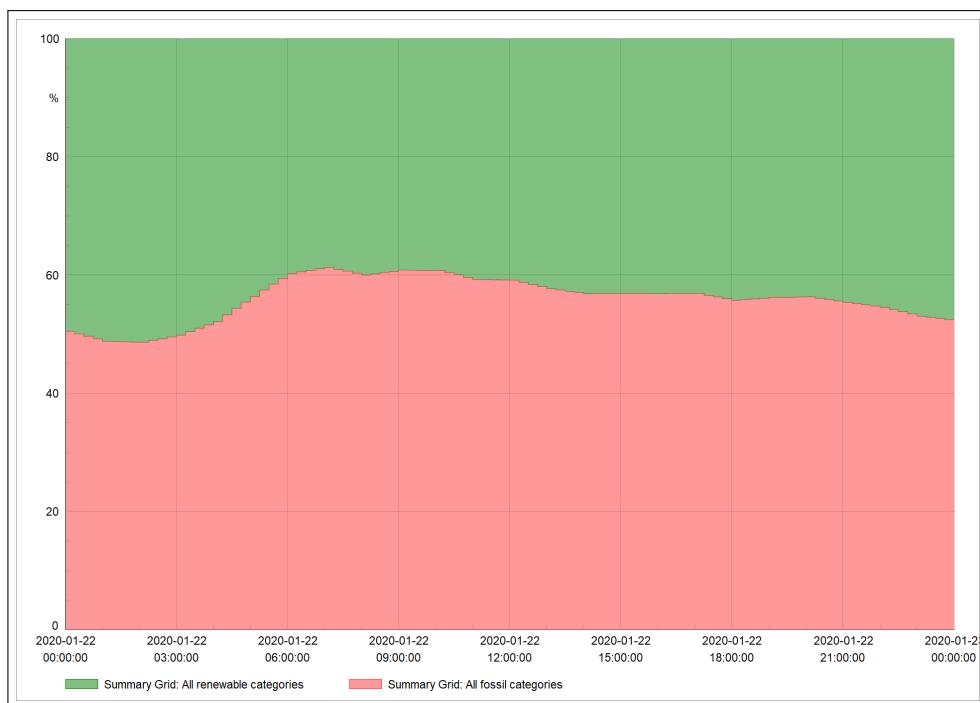


Figure 19.8.18: Total generation in a grid divided into renewable and fossil categories

The pages on the edit dialog of the plot are the same as for the energy plot *Plant categories* described in Section 19.8.9.7.

The only difference is the default entry for the **Curves** since the plant categories in the grid are divided into fossil and renewable categories.

#### 19.8.9.10 Vector Plot

A vector plot is used to visualise complex values such as voltages, currents and apparent power as vectors. A complex variable can be defined and shown in one of two different representations:

- Polar coordinates, e.g. magnitude and phase of the current
- Cartesian coordinates, e.g. active-and reactive power

---

**Note:** A vector plot can be shown after a load flow calculation or before and after a transient simulation.

---

A vector plot can be inserted using the *Insert Plot* dialog or directly by right-clicking on the element and selecting the option *Plots → Insert Vector Plot → “variable”*

Figure 19.8.19 shows an example of a vector plot with vector transformation feature enabled.

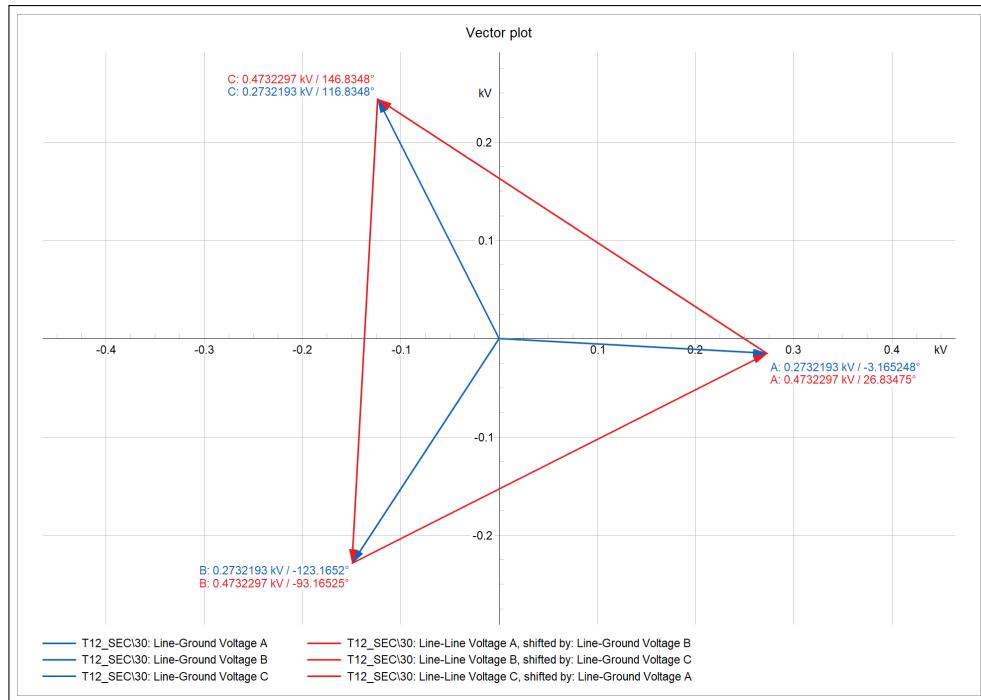


Figure 19.8.19: Vector plot with vector transformation

In this type of plot, instead of the Data Series, a Complex Data Definition (*PltComplexData*) as described in Section 19.8.3 is used.

#### 19.8.9.11 Curve-Input Plot

The curve input command is used for measuring printed curves. The original curves must be available in one of the supported formats and are displayed as a background in the curve input plot as shown in Figure 19.8.20. This plot then allows plot points to be defined by successive mouse clicks.

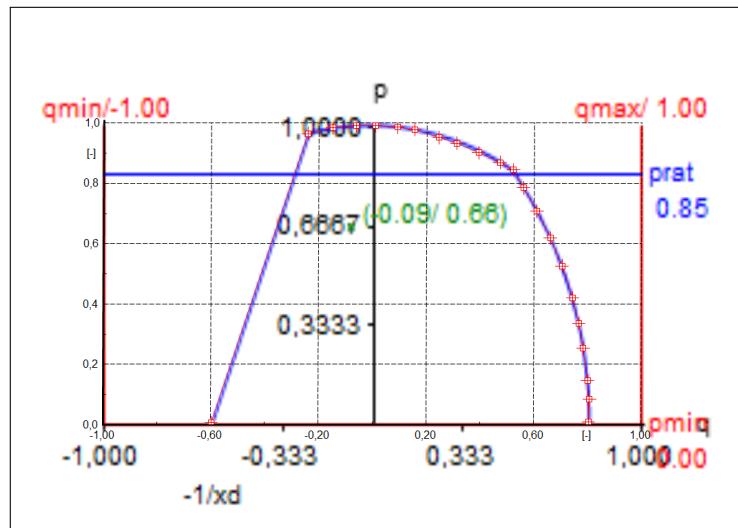


Figure 19.8.20: Curve-Input Plot

The curve input plot allows the measurement and editing of single curves or group of curves at once. The measured curve points will be stored in a Matrix object, which is why, before inserting this plot, it is

necessary to define or select the corresponding matrix.

The matrix object should be created inside the project, for example in the Study Case folder, by opening the Data Manager and once inside the Study Case folder (or selected folder), clicking on the *New Object* icon (  ).

Since several elements can be added to the study case folder, the displayed list is very long. Use the filter to select the object *Matrix (IntMat)* as shown in Figure 19.8.21.

The matrix should have at least two columns and one point (inside the curve) has to be manually defined.

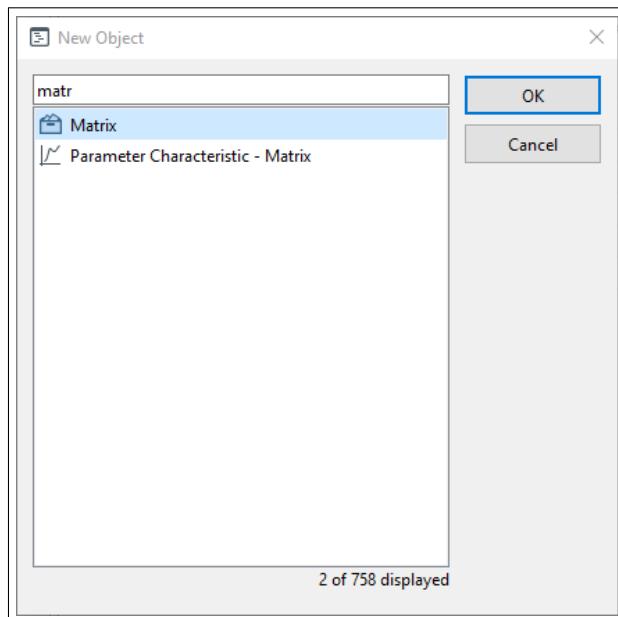


Figure 19.8.21: Defining new matrix

The fields of the curve-input plot edit dialog are:

- **Background:** by clicking on the  button the location of the graphic file to be used as background image can be selected; several formats are supported.
- **Limits:** this is used to set the range of the axes of the curves as they are in the graphics file.
- **Scale:** the options *Linear* and *Log.* (logarithmic) can be selected and should be as per the graphic file.
- **Curves:** two different types of curves can be input:
  - Single: each matrix input defines a single curve. The first column in the matrix holds the x-values, the second one the y values. Other columns are ignored.
  - Set of Curves: only the first matrix is used for input. The first column in the matrix holds the x-values, the other columns hold the y-values of each curve in the group of curves.
- **Interpolation:** the measured curve is drawn between the measured points by interpolation. The available modes of interpolation are:
  - Linear
  - Cub. Spline
  - Polygon
  - Hermite
- **Curves Tables:** the matrix or list of matrices to be used has to be set in this table.

The rest of the setting of the plot are done using the context menu, which is accessed by right-clicking on the plot. The settings are described below in the order they should be executed.

- **Set Axis:** with this option the origin of the axes and the length of the axes can be adjusted according to the figure imported.
  - Origin: sets the origin of the graph to be inserted
  - x-Axis (y=Origin): sets the x-axis dependent on the y-axis origin.
  - x-Axis: sets the x-axis independent of the y-axis.
  - y-Axis (x=Origin): sets the y-axis dependent on the x-axis origin.
  - y-Axis: sets the y-axis independent of the x-axis
- **Active Curve:** sets the curve to modify
- **Input:** specifies the input mode:
  - x/y-Pairs: each left mouse click adds a point to the curve.
  - Drag & Drop: turns on the “edit mode”: all points can be dragged and dropped to change their y-position or left click and delete the point with the **Del** key.
  - Off: switches off the measurement mode
- **Interpolate All:** interpolates undefined y values for all curves for all defined x-values
- **Interpolate N:** interpolates undefined y values of curve N for all defined x-values

#### 19.8.9.12 Virtual Instruments

The virtual instruments are basically measurement instruments that can be used to present steady state values. The variable can be displayed with one of the following instruments:

- Digital display
- Horizontal scale
- Vertical scale
- Measurement instrument

An example of two measurement instruments is shown on the right side of the following figure; the maximum and minimum limits, as well as the element and the variable presented should be defined in the edit dialog of the instrument.

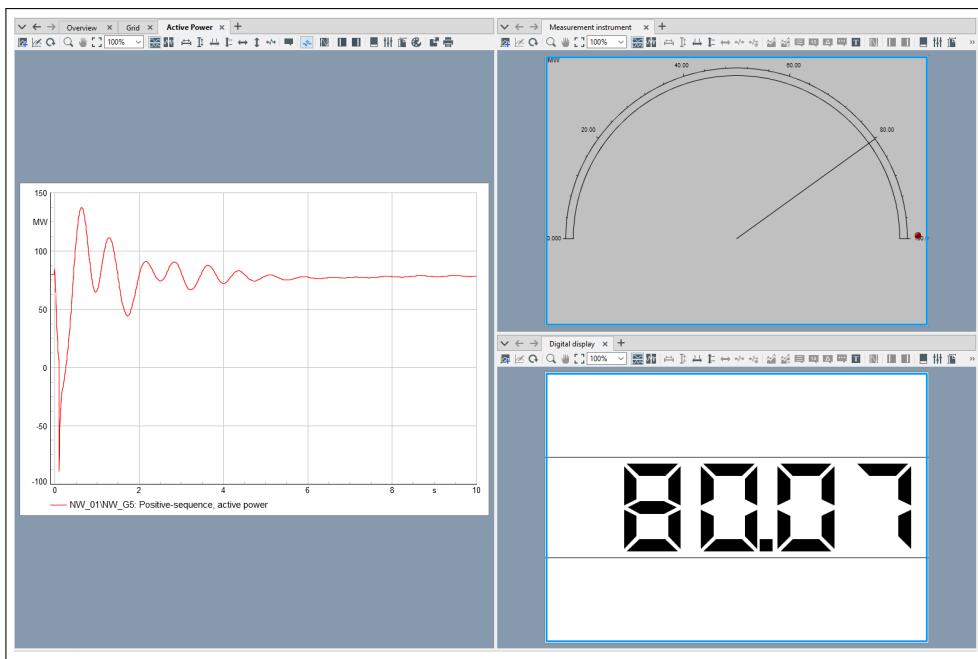


Figure 19.8.22: Measurement Instruments

### 19.8.9.13 Voltage Profile Plot

The *Voltage Profile Plot (along feeder)* shows the voltage profile of a radial network based on the load flow calculation results. The *Voltage Profile Plot* is directly connected to a feeder object defined in the network, so it can only be created for parts of the system where a feeder is assigned.

The *Voltage Profile Plot* requires a successful load flow calculation before it can display any results. The voltage profile plot can be inserted, as all the plots, using the *Insert Plot* dialog, however, since it is linked to one or more feeders, in this case it is recommended to create the plot directly from the context menu of the feeder element, selecting *Show → Voltage Profile*.

An example of a voltage profile plot is shown here:

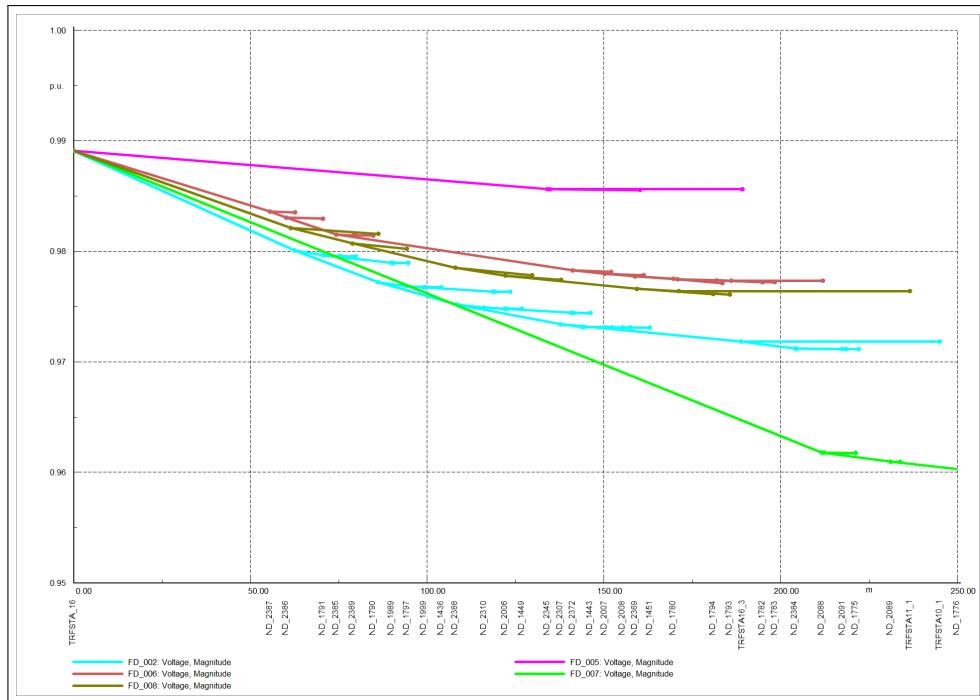


Figure 19.8.23: Voltage profile along feeders

### Customising the Voltage Profile Plot

#### Scales Page

The x-axis variable can be set to one of the following options:

- **Distance:** shows the distance from the beginning of the feeder in km.
- **Bus Index:** each bus is numbered sequentially from the beginning of the feeder and all of the buses are displayed equidistantly on the plot.
- **Other:** this option allows plotting against a user defined variable. Only variables available at all terminals in the feeder can be used.

By default, any branch with a loading greater than 80 % will appear red on the voltage profile plot and any branch loaded less than the *Lower Limit* will be coloured blue. These colours and limits can be adjusted in the *Branch Colouring* field.

The *Parallel Branches* option is required because the voltage profile plot only shows a single connection line between nodes, regardless of how many parallel branches connect the two nodes. If there is a situation where one of these parallel lines is below the *Lower Limit* and another is above the *Upper Limit*, then the parallel branches option determines whether the single line in the voltage profile plot is either the line with the maximum loading or the line with the minimum loading.

#### Curves Page

On the *Curves* page, the colour and style of the displayed feeders can be modified; also, a display filter can be configured to show only the nodes with a nominal values between the specified values and to ignore nodes with voltages below a specified limit.

## Advanced Page

On this page the frame of the plot and the visibility of the legend can be defined. The colour of the busbar (terminal) names on the voltage profile plot can also be modified as follows:

- **Off:** does not display any bus names
- **Black:** shows all names in black
- **Coloured acc. to Feeder:** colours the bus names according to the colour of the different feeders.

The context menu of the plot shows additional functions regarding the voltage profile plot including:

- **Edit Feeder:** opens the *edit* dialog of the feeder related to the plot.
- **Edit Data:** opens the *edit* dialog of the selected line, transformer or other element.
- **Show in Data Manager:** shows the selected element in the Data Manager.
- **Mark in Graphic:** marks the selected element in the single line graphic(s).

# Chapter 20

## Data Extensions

### 20.1 Introduction

The Data Extensions functionality allows users to extend their data models by adding user-defined attributes for elements and other objects in a *PowerFactory* project.

Furthermore, if the users find that defining additional attributes for existing objects is not sufficient to address their needs, it is also possible to define entire new classes of object.

The definitions of the attributes, which are done on an object class basis, include the data type (such as integer, double, string), description, unit and default value. Once defined, the new attributes are treated by *PowerFactory* in the same way as the in-built attributes, available to scripts, DGS interface etc. Data Extensions are specific to the project, but the configuration can be copied from one project to another, where the new Data Extensions will be aggregated with any existing Data Extensions.

### 20.2 Data Extension Configuration

#### 20.2.1 Creating Data Extensions

To create Data Extensions within a project, select from the main toolbar *Tools* → *Data Extensions* → *Configuration*. The Data Extension Configurations are stored in the project Settings folder, and the required \*.SetDataext will be automatically created as required.

When the *Tools* → *Data Extensions* → *Configuration* command is used, a dialog box appears, which allows the user to create new Data Extensions or modify or delete existing ones.

To create a new Data Extension variable definition, follow these steps:

- Use the *New Object* icon ; this will then create a new *IntAddonvars* object, where the attributes are configured.
- Select or type in the name of the class for which the attributes are to be configured. “Wildcards” (e.g. Elm\*) may be used if the attributes are to be made available to multiple classes. Note that if the class name is not recognised as an existing class, it will be assumed that the user wishes to create an entirely new class (see Section 20.2.2 below)
- Give the definition a meaningful name.
- Use right-click to append rows in the table below.

- Populate the rows as required. The Name is the actual attribute name and the Description will appear, for example, as a column heading if the attribute is used in a flexible data page.
- The attribute Type is selected from a drop-down list, and the Unit and Initial Value can also be specified as required.
- By clicking on *More...* an *Additional Options* dialog opens, in which the previous described attributes as well as the following three additional attributes can be defined:
  - With the attributes *Minimum value* and *Maximum value* the permissible range for the values can be defined. Note: These fields are only available for data types “int” and “double”.
  - The attribute *Allowed values* can be used to define which values are to be selectable from a drop-down menu. Multiple values must be separated by a semicolon (e.g. 0;1;3;5). Note: This field is only available for data types “int”, “double” and “string”. Additionally, the fields for the minimum and maximum value will be disabled.
- Click on **OK** to save the new definition.

Note that once Data Extensions have been created, further additions or changes will result in a need to migrate the data within the project, and a version will be created in case there is a need to roll back to the state prior to the change.

Modifications to Data Extensions must be done using the *Tools → Data Extensions→ Configuration* command. Modifications cannot be done by going to the Settings folder of the project and accessing the Data Extensions directly there. It should also be noted that it is currently not possible to change an attribute’s type in an existing Data Extension. The attribute must be deleted, the configuration saved and then the attribute can be redefined with a new type.

### 20.2.2 User Defined Classes

In an enhancement to the above process, the user can choose to define a completely new class of object, then ascribe the attributes to this new object class.

To define a new object class, follow the above steps but instead of using an existing class name enter a new class name. *PowerFactory* will create the new class, prefixing it with “Ext” to indicate that it is a user-defined class.

Once the new class is defined, objects of that class can be created and handled like other objects in the project, for example being populated with data and accessed via DPL or Python scripts.

## 20.3 Using Data Extensions

The new attributes can be treated in much the same way as existing attributes, for example added to flexible data pages or accessed by scripts.

Characteristics can also be added to the new variables. If the reference to the characteristic (*ChaRef* object) is created via a script, it needs to be named “p#attrname”.

To add a new attribute to a flexible data page (see Chapter 11 ([Data Manager and Network Model Manager](#)), Section 11.2.4 for more information about the use of flexible data pages), open up the Variable Definition dialog and select Data Extension on the left-hand side.

They have a prefix p, rather than the e used for the built-in attributes, so for example if Data Extension for synchronous machines includes a new attribute Size, the parameter will be shown as p:Size.

The value of a Data Extension attribute can be modified within the object’s edit dialog (select the Data Extension page) or in a Flexible data page, as with other parameters. Attributes of primitive data types,

i.e. integer and double, can be edited in place. For more complex attributes including strings, double-clicking into the cell opens a dedicated edit value dialog. Although it is not possible to type such values directly in a flexible data page, it is still possible to copy and paste them from one object to another.

As mentioned in the introduction, Data Extensions can be used by scripts, DGS imports etc. Recording in scenarios is also supported; however, this is restricted to attributes of type string, integer, double or object.

It is possible to rename existing data extensions and user-defined classes via DPL or Python, using *RenameParameter* and *RenameClass*.

## 20.4 Sharing Data Extensions

Having created Data Extensions in one project, users may wish to use them in other projects. It is possible to fetch Data Extension configurations from another project using the command *Tools → Data Extensions → Copy settings from project*. This process is additive, i.e. the fetched Data Extensions will be added to any existing Data Extensions in the project. However, the user will be required to resolve any conflicts such as duplicate attribute names for the same object class.

Another option open to users if they have Data Extensions which they want to use routinely is to create them in the default project (in the Configuration, Default folder) so that every new project created in the database will automatically have them.

As the Data Extensions are part of the project, they are therefore retained when moving or copying projects. Likewise, they are also retained when projects are exported as \*.pdf, \*.pfds or snapshot export \*.dzs files.

# Chapter 21

# Data Management

## 21.1 Introduction

The basic elements of project management within the *PowerFactory* environment were introduced in Chapter 4 (*PowerFactory* Overview). They allow the user to generate network designs and administer all input information and settings related to *PowerFactory* calculations and analyses. The project itself is much more than a simple folder which stores all objects which comprise a power system model; it allows the user to do advanced management tasks such as: versioning, deriving, comparing, merging and sharing. These advanced features simplify data management in multi-user environments.

The following sections explain each of the data management functions in more detail:

- Project Versions;
- Derived Projects;
- Comparing and Merging Projects;
- How to update a Project;
- Sharing Projects;
- Combining Projects; and
- Database Archiving.

## 21.2 Project Versions

The section explains the *PowerFactory* concept of a version. The section first explains what a version is and when it can be used. Next the procedure for creating a version is explained. Specific procedures related to versions such as rolling back to a version, checking if a version is the basis for a derived project and deleting a version are then explained.

### 21.2.1 What is a Version?

A *Version* is a snapshot of a project taken at a certain point in time. Using versions, the historic development of a project can be controlled. Also, the previous state of a project can be recovered by rolling back a version. From the *PowerFactory* database point of view, a version is a read-only copy of

the original project (at the moment of version creation), which is stored inside a Version (*IntVersion*, ). Versions are stored inside the original project in a special folder called *Versions*.

The concept of versions is illustrated in Figure 21.2.1. At time  $t_0$ , the project 'SIN' is created. After a time,  $t_1$ , when the owner has made several changes they decide to make a copy of the project in its current state by creating the version 'V1'. After more time,  $t_2$ , and after more changes with respect to 'V1', another version 'V2' is created by the owner. The version control can continue with time like this, with versions accumulating with a periodicity of  $t$ .

After versions are created, the owner can revert the project to the state of the version by using the *Rollback* function. This *destroys* all modifications implemented after a version was created (including all versions created after the *rolled-back* version).

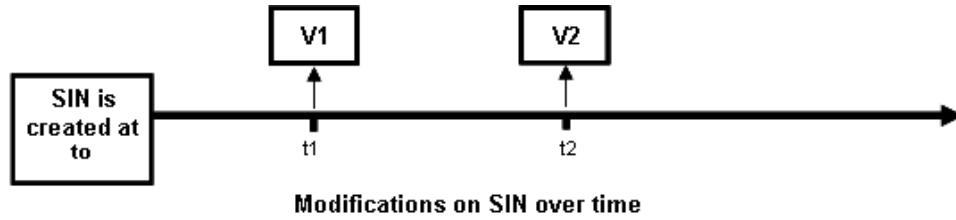


Figure 21.2.1: Project versions

## 21.2.2 How to Create a Version

This sub-section describes the procedure for creating a version. To create a version of the active project follow these steps:

1. Right-click on the active project.
2. Select *New* → *Version...* from the context menu. Alternatively, use the option *File* → *New Version* from the main *PowerFactory* menu. The dialog for the new version appears.
3. Set the desired options (explained in the next section) and press **OK**. *PowerFactory* automatically creates and stores the version in the *Versions* folder (which is automatically created if it does not yet exist).

### 21.2.2.1 Options in the Version Dialog

#### Point in Time

By default this is set to the system clock time when the version was created. However, it is also possible to enter an earlier time (back to the beginning of retention period of the project).

---

**Note:** Setting a *Point in Time* earlier than the clock time means that the version is created considering the state of the project at the time entered. This can be used for example, to revert the project to a previous state, even though other versions have not yet been created.

---

#### Notify users of derived projects

If this option is enabled, when a user of a project that is derived from the active project activates their derived project, they are informed that the new version is available. Thereafter, updates of the derived project can be made (for further information about derived projects refer to Section 21.3).

### Complete project approval for versioning required

If this option is enabled, *PowerFactory* checks if all the objects in the active project are approved. If *Not Approved* objects are found, an error message is printed and the version is not created.

**Note:** The *Approval Status* is found on the *Description* page in the dialog of most grid and library objects.

### 21.2.3 How to Rollback a Project

This sub-section describes the use of the *Rollback* function to revert a project to the state of a version of that project. For example, consider a project with different versions at different times. If a *rollback* to time 'V1' is performed, the project will revert to the state it was in when Version 1 was created. All changes made after 'V1' will be deleted, including the newer versions Version 2 and Version 3. This concept is illustrated in Figure 21.2.2.

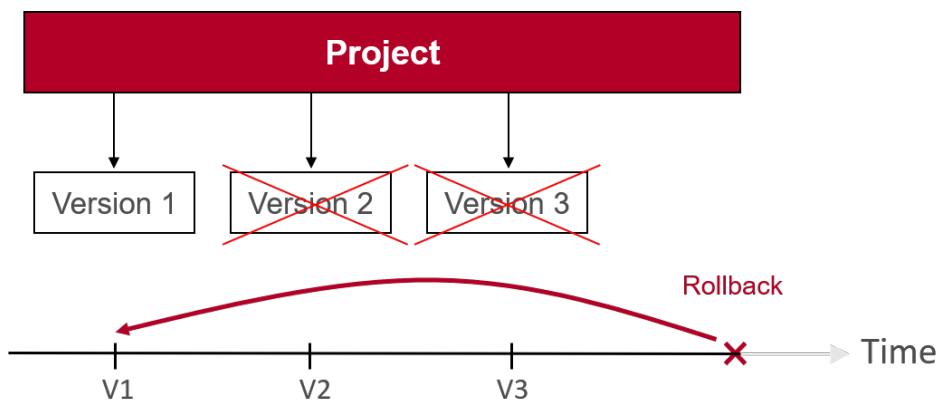


Figure 21.2.2: Example of a rollback

#### To complete a rollback

1. Deactivate the target project.
2. Right-click on the version that you wish to rollback to and select the option *Rollback to this version* from the context menu.
3. Press **OK** when the confirmation message appears.

Note that a *Rollback* is not allowed (and therefore not enabled in the context menu) if a newer version of the project exists and this version is the base of a derived project. **A rollback cannot be undone!**

**Note:** A version can only be deleted if it does not have derived projects.

### 21.2.4 How to Check if a Version is the Base for a Derived Project

The following steps should be followed to check if a version is the base for a derived project:

1. Activate the project.

2. Go to the *Versions* folder inside the project.
3. Right-click on the *Version* that should be checked. This should be done via the right window pane in the Data Manager, not the main Data Manager tree.
4. Select the option *Show Derived Projects*.
5. A list of derived projects will be shown in *PowerFactory*'s output window.

### 21.2.5 How to Delete a Version

To delete a version:

1. Activate the project containing the version.
2. Go to the *Versions* folder inside the project.
3. Right-click on the Version that should be deleted.
4. Select the option *Delete*.

## 21.3 Derived Projects

This section explains the concept of a derived project. For background information regarding the use of derived projects, see sub-Section 21.3.1. In addition, sub-Section 21.3.2 describes the procedure for creating a derived project.

### 21.3.1 Derived Projects Background

As is often the case, several users might wish to work on the same project. To avoid large amounts of data duplication that would be required to create a project copy for each user, DlgSILENT has developed a *virtual copy* approach called *derived* projects. From the user's point of view, a derived project is like a normal copy of a project version. However, only the differences between the original project version (the *base project*) and the virtual copy (the *derived project*) are stored in the database. Because the derived project is based on a version, changes made to the base project do not affect it. Like 'normal' projects, derived projects can be controlled over time by the use of versions, but these *derived* versions cannot be used to create further derived projects.

---

**Note:** A derived project is a local 'virtual copy' of a version of a base project (master project):

- It behaves like a "real copy" from the user's point of view.
  - Internally, only the data differences between the *base project* and the *derived project* are stored in the database.
  - This approach reduces the data overhead.
- 

In a multi-user database, the data administrator might publish a *base project* in a public area of the database. Every user can subsequently create their own derived project and use as if it is the original base project. Changes made by individual users are stored in their respective derived projects, so that the base project remains the same for all users.

In a single-user database, the data administrator must export the *base project*. The user of the derived project must always have this project imported. However, different users of the same *base project* can exchange their derived project. Therefore the derived project should not be exported with option *Export derived project as regular project* enabled. See Section 9.1.5 for further details.

The purpose of a derived project is that all users work with an identical power system model. The derived project always remains connected to the base project.

The concept of derived projects is illustrated in Figure 21.3.1; here version 'Version3' of the base project ('MasterProject') was used to create 'DerivedProject'. After 'DerivedProject' was created, two versions of it were created.

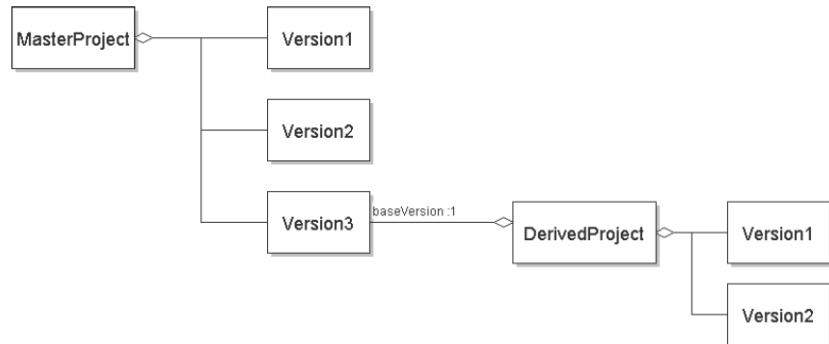


Figure 21.3.1: Principle of derived projects

At any stage, the data administrator might create a version of a base project which has derived projects from other versions of the base project. The user might wish to update their derived project with one of these new versions. Alternatively, the data administrator might like to incorporate changes made in a derived project to the base project. All of these features are possible, by using the Compare and Merge Tool, explained in Section 21.4.

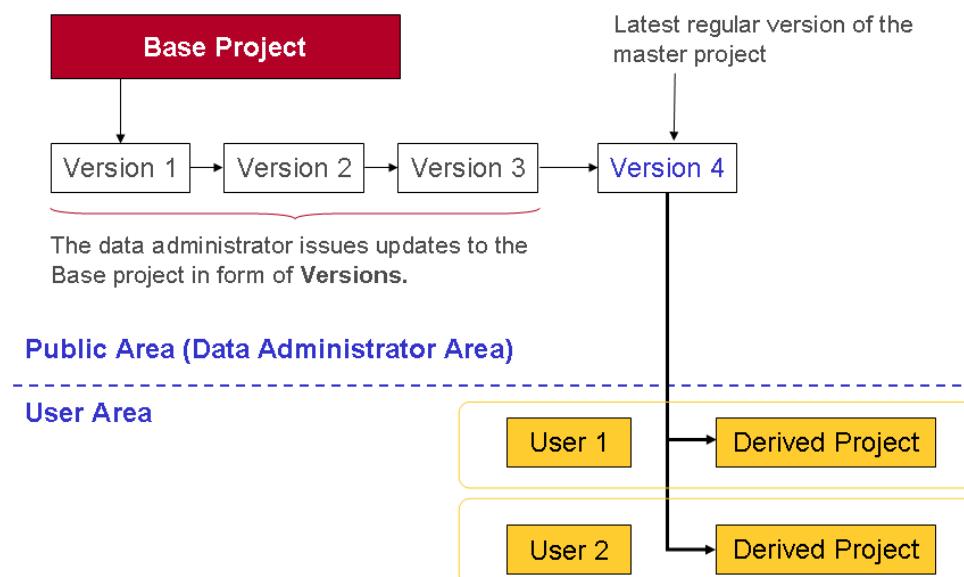


Figure 21.3.2: Derived projects in a multi-user database

In the Data Manager, a derived project looks like a normal project. The *Derived Project* page of its dialog has a reference where the user can see the base project and the version used to derive the project.

Users are notified of changes in the base project if: there is a new version of the base project (newer than the currently-used version) which has the option *Notify users of derived projects* enabled (the user/administrator enables this option when creating a new version), and the option *Disable notification at activation* disabled (found on the *Derived Project* page of the project dialog).

The user may update a derived project when they next activate it, provided that the conditions stated above are met. The newest version that can be used to update a derived project is referred to (if available) in the *Most recent Version* field of the dialog. Users can compare this new version with their own derived project and decide which changes to include in the derived project. For comparing and accepting or refusing individual changes, the Compare and Merge Tool is used. For information about the Compare and Merge Tool refer to Section 21.4.

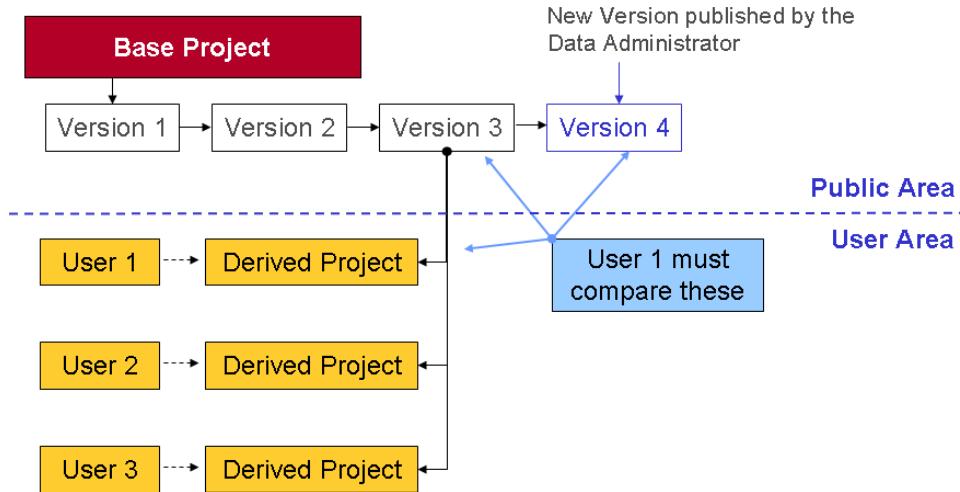


Figure 21.3.3: New version of base project in a multi-user database

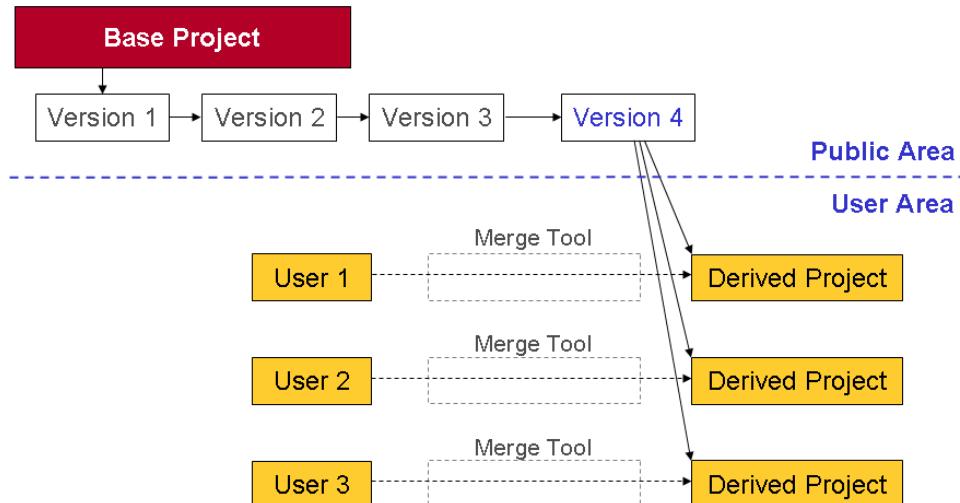


Figure 21.3.4: Merging the new version of the base project into derived projects

### 21.3.2 How to Create a Derived Project

The most direct and intuitive method for creating a Derived Project is by means of the Data Manager, as explained below:

1. Right-click on the desired base project in the left pane of the Data Manager,
2. Select the option *Create Derived Project* from the context menu,
3. Select the source Version of the base project using the data browser that appears automatically,
4. Edit, if required, the predefined settings of the Derived Project and
5. Press **OK**.

After the derived project is created, it can be used in a similar way as a regular project.

---

- Note:**
- The base or master project has to have at least one version before other projects can be derived from it.
  - A derived project cannot be used as a base project, i.e. it is not possible to create a derived project from another derived project.
  - Whether a project is derived or not can be checked on the *Derived Project* page of the project dialog.
  - To create a derived project from a base project stored in another user's account, read access is required. See Section [21.6](#) for further details.
- 

The derived project can be exported as a “Regular Project” or with the base project. This option can be selected from the Export dialog.

## 21.4 Comparing and Merging Projects

This section describes the procedure for comparing and merging projects within the *PowerFactory* database. There are many circumstances whereby it may be desirable and/or necessary to merge data from multiple projects. For example, when the data administrator updates a master project that is the base project for a derived project that a user is working with. The Compare and Merge Tool (CMT) can be used to update the user's project with the data changes, yet also allows the user control over which changes are implemented.

This section is separated into six sub-sections. Firstly, the background of the CMT is presented. The following sub-section explains the procedure needed for merging together or comparing two projects. Sub-Section [21.4.3](#) explains the procedure for merging or comparing three projects. In sub-Section [21.4.4](#), the advanced options of the CMT are explained. The CMT uses a *diff browser* for showing the differences and conflicts between compared projects and also for allowing data assignments. This is explained in sub-Section [21.4.5](#).

### 21.4.1 Compare and Merge Tool Background

When working collaboratively in a multi-user environment, a data administrator might often need to update the *master* project to create a version based on updates completed by one or more users to projects derived from the master project. The Compare and Merge Tool (CMT) is used for this purpose. This tool can be used for project comparison in addition to the merging of project data. It is capable of a *two-way comparison* between two projects and also a *three-way comparison* for three projects.

Internally, *PowerFactory* refers to each of the compared projects according to the following nomenclature:

- <Base> Project - the base project for comparison.
- <1st> - the first project to compare to the <Base> project.
- <2nd> - the second project to compare to the <Base> project and to the <1st> project (three-way comparison only).

The CMT compares the chosen projects and generates an interactive window known as the CMT *diff browser* to show the differences. For a two-way merge, the changes found in the <1st> project can be implemented in the <Base>, provided that the user selects <1st> as the *source* (<Base> is by default the *target*). When merging three projects together, the *target* is either the <1st> or <2nd> project.

## 21.4.2 How to Merge or Compare Two Projects Using the Compare and Merge Tool

This section describes the procedure for merging together or comparing two projects using the Compare and Merge Tool (CMT) (*ComMerge*). Note that the comparison is completed using a similar procedure but with slight differences that will also be explained here.

To merge or compare two projects:

1. In the Data Manager, right-click on an inactive project and choose *Select as Base to Compare*.
2. Right-click on a second (inactive) project and select *Compare to [Name of Base Project]*. The CMT options dialog will appear. The <Base> and the <1st> project are listed in the *Compare* section of the dialog.
3. *Optional:* If a third project should be included in the comparison, the box next to <2nd> must be checked. The third project can then be selected with a data browser by using the  icon. See Section 21.4.3 for a more detailed explanation of the 3-way comparison.
4. *Optional:* If the base and compare projects should be swapped around, press the  button. This would be the case if Project A should be the <1st> project and Project B should be the <Base>.
5. Select one of the options *Compare only*, *Manually* or *Automatically*. The differences between these three choices are:
  - *Compare only:* If the two projects should only be compared and no merge is desired, then select *Compare only*. This disables the merge functionality and only the differences between the two projects will be shown.
  - *Manually:* When this option is selected, the user will be asked to make assignments (i.e. to choose the source project data for common objects that are merged together). For this option, the target project can also be selected. Selecting <Base> will merge changes into the <Base> project, whereas selecting <1st> will instead merge changes into the <1st> comparison project.
  - *Automatically:* When this option is selected, *PowerFactory* will attempt to automatically merge the two projects together, by automatically making data assignments. In a two-way comparison, merging will be automatically into the base project (the base is automatically assumed to be the 'target' for the merging procedure). Note that if *conflicts* are detected during an automatic merge, the CMT will automatically switch to manual mode.
6. Press **Execute** to run the compare or merge. The CMT *diff browser* will appear (unless an automatic merge was selected and no conflicts were identified by *PowerFactory*). Interpreting and using the *diff browser* is described in Section 21.4.5.

---

**Note:** It is possible to assign user-defined names to each of the compared projects. This makes it easier to recognise which project is being referred to by the CMT later on in the diff browser (see Section 21.4.5). For example, the user might wish to name two compared projects 'Master' and 'User', respectively. User-defined names can be implemented by typing the desired name in the *as ...* field in the CMT options dialog. These user-defined names are limited to a maximum of 10 characters.

---

## 21.4.3 How to Merge or Compare Three Projects Using the Compare and Merge Tool

This section describes the procedure for merging or comparing three projects using the Compare and Merge Tool (CMT). The comparison procedure is completed using a similar method to that used for a two-way merge or compare, but with minor differences that will be explained here.

To merge or compare three projects:

1. In the Data Manager, right-click an inactive project and choose *Select as Base to Compare*.
2. In the window on the right of the Data Manager, hold the **CTRL** key to multi-select a second and third inactive project.
3. Right-click the multi-selection and select the option *Compare to <project>*. The CMT options dialog will appear as shown in Figure 21.4.1. The <Base>, the <1st> and the <2nd> project are listed in the *Compare* section of the dialog.

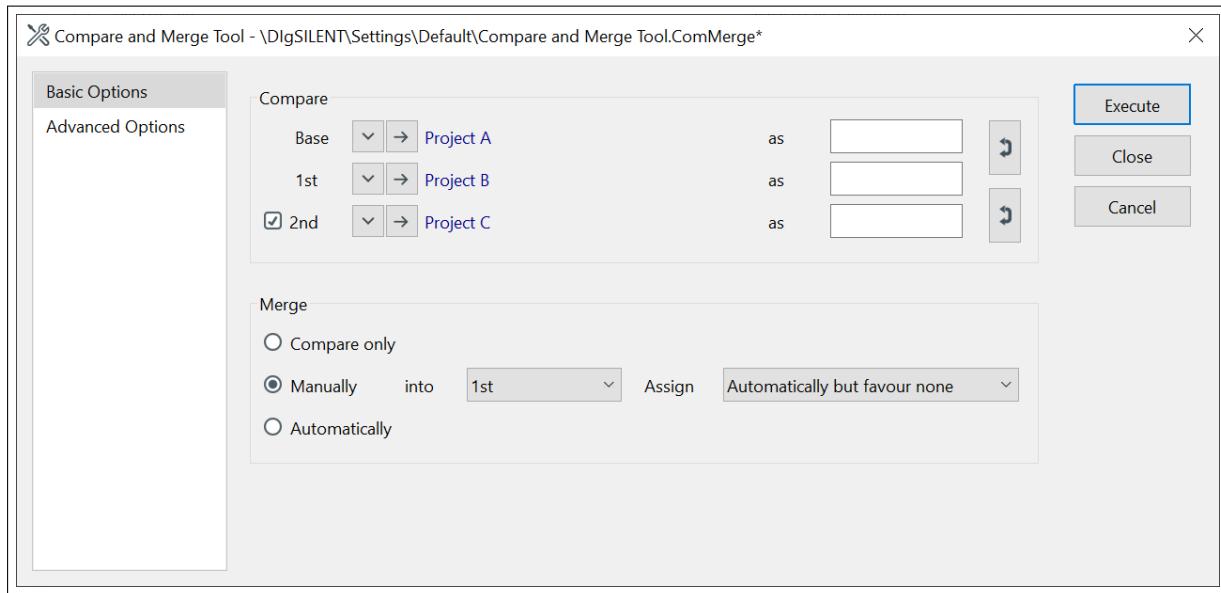


Figure 21.4.1: Compare and Merge Tool options dialog for a three-way merge

4. Select one of the options *Compare only*, *Manually* or *Automatically*. The differences between these three choices are:
  - *Compare only*: If only two projects should be compared and no merge is required, then select the radio button *Compare only*. This disables the merge functionality and only the differences between the two projects will be shown.
  - *Manually*: When this option is selected, the user will be asked to make assignments (to choose the source project data for common objects that are merged together). Using this option, the target project can also be selected. For a three-way merge, merging cannot be done into the <Base>, meaning that either the <1st> or the <2nd> project must be selected.
  - *Automatically*: When this option is selected, *PowerFactory* will attempt to merge the three projects together, via automatic data assignments. As for the option *Manually*, the target can be either the <1st> or <2nd> project. Note that if 'conflicts' are detected during an automatic merge, the CMT will automatically switch to manual mode.
5. If using options *Manually* or *Automatically*, the assignment priority must also be selected, by choosing an option from the *Assign* drop-down menu. This defines the default assignment in the CMT diff browser (or automatic merge) when *PowerFactory* identifies conflicts. For example, say the CMT identifies that the load 'L1' has an active power of 10 MW in <Base>, 12 MW in <1st> and 13 MW in <2nd>. By choosing the option *Automatically and favour 1st*, the default assignment for 'L1' would be <1st>, and a power of 12 MW would be assigned to this load in the target project (provided that the user did not manually alter the assignment).
6. Press **Execute** to run the compare or merge. The CMT *diff browser* will appear (unless an automatic merge was selected and no conflicts were identified by *PowerFactory*). Using the *diff browser* is described in Section 21.4.5.

---

**Note:** It is possible to assign user-defined names to each of the compared projects. This makes it easier to recognise which project is being referred to by the CMT later on in the diff browser (see Section 21.4.5). For example, the user might wish to name two compared projects 'Master' and 'User', respectively. User-defined names can be implemented by typing the desired name in the *as ...* field in the CMT options dialog. These user-defined names are limited to a maximum of 10 characters.

---

## 21.4.4 Compare and Merge Tool Advanced Options

### 21.4.4.1 General

#### Object Identification

- **Use a key value:** Used to define the key value that *PowerFactory* will use to align the objects irrespective of their name and location. This can be useful when combining grid models from different sources when different names are used and/or the folder structure may not fit perfectly. The following options are available for the key value:
  - Foreign key: the foreign key defined on the *Description* page of the object will be used
  - CIM RDF ID: the CIM RDF ID Mapping defined on the *Description* page, *Advanced* tab of the elements will be used.

- **Search correspondents for added objects:** This option is only available for a three-way merge and is enabled by default. If enabled, *PowerFactory* can automatically align two independently added objects as being the same object. This option can be useful when completing a comparison on projects where users have added the same object (same name) in each of their respective projects, and the user would like to ensure that *PowerFactory* identifies this object as being the same object.

Note that this option is only considered when the *Use a key value* is set to *None*.

#### Attribute Modifications

- **Consider approval information:** By default this option is disabled, which means that information on the *Description* page under *Approval Information* is not compared. For example, if this option is disabled and an object's *Approval status* changes from *Not Approved* to *Approved* or vice versa, then this modification would not be registered by the CMT comparison engine.
- **Consider CIM Data:** the information located within the CIM Model library will also be considered by the CMT, this is the default option.
- **Consider Data Extensions:** the Data Extensions defined in the project will also be compared/merged, this is the default option.

#### Create versions of the target project

- **Before merging:** a version of the status of the project before the merging will be created on the target project.
- **After merging:** a version of the status of the project after the merging will be created on the target project.

### 21.4.4.2 Advanced

#### Depth

This option controls whether the CMT compares only the selected objects or also all objects contained within the compared objects. By default, *Chosen and contained objects* is enabled which means the CMT compares all objects within the selected comparison objects. This is generally the most appropriate option when merging projects.

#### Ignore differences <

This field defines the tolerance of the comparison engine when comparing numerical parameters. If the difference between two numerical parameters is less than the value entered into this field, then the comparison will show the two values as equal, =.

## 21.4.5 Compare and Merge Tool 'diff browser'

After the CMT options have been set, press the **Execute** button to start the CMT comparison. The comparison and assignment results are then presented in a data browser window (the CMT *diff browser* window shown in Figure 21.4.2). The *diff browser* is divided into three parts:

- Data Tree window on the left;
- Comparison and Assignment window on the right; and
- Output window at the bottom.

These features are explained in the following sections.

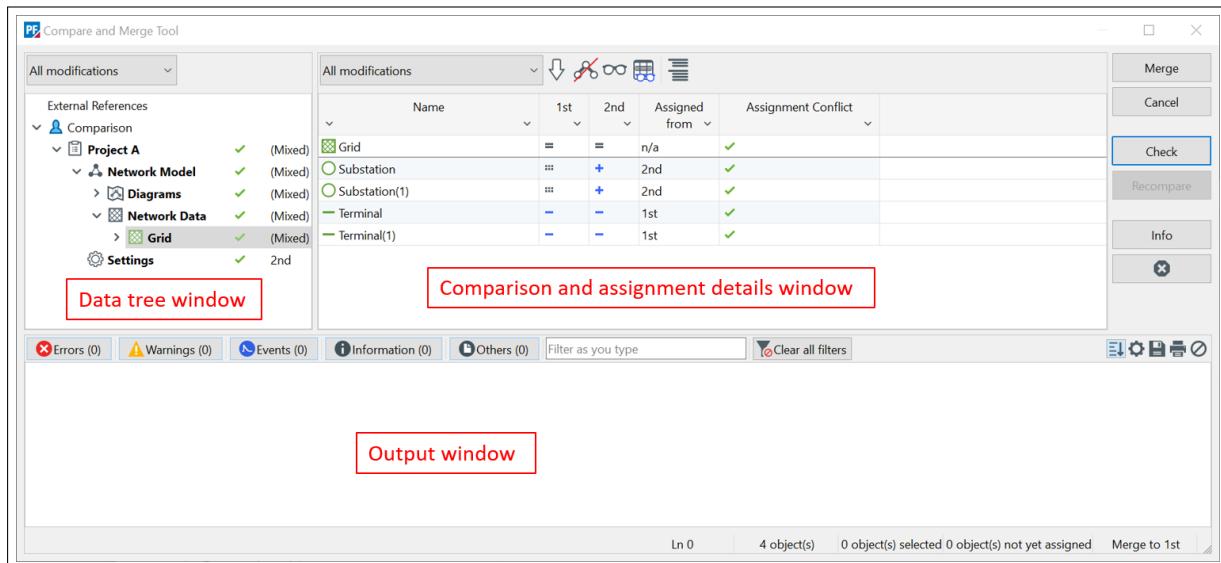


Figure 21.4.2: Compare and Merge Tool *diff browser* after a three-way merge

### 21.4.5.1 Output Window

The output window displays reports from the context menu (right-click), and other error information.

### 21.4.5.2 Comparison and Assignment window

In the Comparison and Assignment Window, a list of the compared objects is shown. The window appears slightly different depending on whether a two-way merge, a three-way merge or a comparison has been performed. For instance, after a comparison, the *Assigned from* and *Assignment Conflict* columns are not shown. After a two-way merge, the columns with the project names will show <Base> and <1st> (or user-defined names), whereas after a three-way merge they will show <1st> and <2nd>. A comparison result symbol, indicating the differences found for each object from the list, is displayed in the columns <Base> and <1st> after a two-way merge and in columns <1st> and <2nd> after a three-way merge. The possible combinations of these symbols are shown and explained in Tables 21.4.1 and 21.4.2.

Base	1st	Comment
+	-	The object has been removed from the <1st> project
-	+	The object has been added to the <1st> project
△	△	A parameter of the object has been modified in the <1st> project
=	=	The object is identical in both projects

Table 21.4.1: Possible results after a two-way comparison or merge

1st	2nd	Result	Comment
=	=	=	Objects are identical in all projects
-	△	✓	A parameter of the object is modified in the <2nd> project
△	=	✓	A parameter of the object is modified in the <1st> project
⋮	+	✓	A new object in the <2nd> project
+	⋮	✓	A new object in the <1st> project
=	-	✓	Object removed from the <2nd> project
-	=	✓	Object removed from the <1st> project
△	△	✓	Modified in both projects but the same modifications in both
△	△	!	Modified in both projects but the modifications are different
△	-	!	Modified in the <1st> project and removed from the <2nd> project
-	△	!	Modified in the <2nd> project and removed from the <1st> project
+	+	✓	Identical object added to both projects
+	+	!	Object added to both projects but parameters are different
-	-	✓	Object removed from both projects

Table 21.4.2: Possible results after a three-way comparison or merge

For a project merge (i.e. the *Merge* option was enabled in the command dialog), the *Assigned from* must define the source project of the changes to implement in the target project. All listed objects must have an *Assignment*. If a certain change should not be implemented in the target; then the 'target' project must be selected as the source.

Special attention should be paid to all results indicated by the 'conflict' symbol ! . This symbol shows that the objects are different in both compared projects or that another error has occurred. In the case of conflicts, the user must always indicate the source project for the data.

In a two-way merge, the only available sources for assignment are the <Base> (which is also the target) and <1st>. In a three-way merge, the possible sources are <Base>, <1st> and <2nd>. The assignment can be made manually by double-clicking on the corresponding cell in the *Assigned from* column and selecting the desired source, or double-clicking the <Base>, <1st> or <2nd> cell that the user wishes to assign. However, this task can be tedious in large projects where there are many differences. To rapidly assign many objects, the objects can be multi-selected and then *Assign from ...* or *Assign with Children from ...* can be selected from the context menu (right-click).

Following the assignment of all the objects, the projects can be merged by pressing the **Merge** button. The changes are then automatically implemented in the target project.

**Note:** The Comparison and Assignment window always shows the selected object in the Data Tree window in the first row.

#### 21.4.5.3 Data Tree Window

The window on the left side of Figure 21.4.2 shows the *Data Tree*, which is similar in appearance to PowerFactory's Data Manager tree. This window shows the compared objects in a normal project tree structure. At each level of the tree, there is an indication on the right showing the status of the comparison of the contained objects (and the object itself).

The legend for the comparison indication is shown in Table 21.4.3.

Icon/Text	Meaning
✓	Assignments/comparison is okay
!	Conflicts exist
Mixed/<Base>/<1st>/<2nd>	The text indicates the assignments within by indicating the assigned project. If assignments within are from multiple different sources, then 'Mixed' will be shown.
?	Assignments missing
Bold red font	Three-way merge - information will be lost during the merge Two-way merge - information could be lost during the merge

Table 21.4.3: Data Tree window legend

#### 21.4.5.4 Diff Browser Toolbar

As previously mentioned, the objects displayed in the CMT window can be sorted and organised by the toolbar as shown in Figure 21.4.3. The buttons available are explained in this section.



Figure 21.4.3: Compare and Merge Tool 'diff browser' toolbar

#### Modifications to be shown

The *Modifications to be shown* drop-down menu allows the results in the comparison windows to be filtered according to their comparison status. Possible filter options for a three-way comparison are:

- All objects
- All modifications (default)
- All modifications in <1st> (show all modifications, additions and deletions in the <1st> project)
- All modifications in <2nd> (show all modifications, additions and deletions in the <2nd> project)

- All modifications in both (show only those objects which exist in both projects and have been modified in both projects)
- All modifications in both but different (show only those objects which exist in both projects and have been modified in both projects to different values)
- Added in <1st> (show only objects added to the <1st> project)
- Modified in <1st> (show only objects modified in the <1st> project)
- Deleted in <1st> (show only objects deleted from the <1st> project)
- Added in <2nd> (show only objects added to the <2nd> project)
- Modified in <2nd> (show only objects modified in the <2nd> project)
- Deleted in <2nd> (show only objects deleted from the <2nd> project)

The following options are available for a two-way comparison:

- All objects
- All modifications
- Added in <1st>
- Modified in <1st>
- Deleted in <1st>

Only one option can be selected at a time.

#### Show all objects inside chosen object

This button will list all compared objects and also all contained objects (at every level of the tree).

#### Show graphical elements

Pressing this button will prevent graphical differences from appearing in the comparison window. Because graphical changes often occur, and are usually trivial (i.e. a slight adjustment to the x-axis position of an object), this button is extremely useful for organising the data.

#### Detail mode and Detail mode class select

These two buttons can be used to show only objects of a selected class.

#### Group dependent objects

If this option is enabled, dependent objects are listed indented underneath each listed comparison object. A dependent object is defined as an object that is referenced by another object. For example, a line type (*TypLne*) is a dependent object of a line element (*ElmLne*), as are the cubicles that connect the line element to a terminal. If the objects are grouped and not filtered otherwise, every object has to be listed at least once but can be listed several times as a dependency. Non-primary objects (such as graphical elements) are only listed separately if they are not listed as a dependency for another object.

Dependent objects are not filtered. By default, the grouping of dependent objects is not displayed because this type of display can quickly expand to become unusable (because in a typical project there are many dependencies).

---

**Note:** The filters on the columns can be used to get only those object with conflict or not assigned.

---

#### 21.4.5.5 Diff window right-click menu options

A context menu can be accessed by right-clicking on a cell or an object in the Data Tree window or in the Comparison and Assignment window. The following options are available:

##### Show Object...

A project selection window will appear so that the user can select to show specific object data. After the reference project has been selected, the dialog of the selected object is then displayed. The dialog is read-only.

##### Output Modification Details

This prints a report to the output window showing the details of the differences for the selected objects. The format of the report is an ASCII table with the modified parameters as rows and the parameter values in each compared project as columns. The date and time of the last modification along with the database user who made the last change are always shown in the first two rows.

##### Output Non-OPD Modification Details

This option is similar to the *Output Modification Details* option, but it only shows the modifications that are not classed as *Operational Data*.

##### Align Manually

This option allows the compared objects to be *realigned* across the compared projects. What this means is that *disparate* objects can instead be compared directly. This could be useful for example when two different users have added an object to their derived projects but each has given it a slightly different name, even though the objects are representing the same 'real world' object. The CMT would see these objects as different objects by default. In this case, the data administrator might wish to tell *PowerFactory* that these two *different* objects are the same object. This can be achieved using the *Align Manually* function.

##### Ignore Missing References

For every compared object, missing references can be optionally ignored. The assignment check will then not check the references of the object. Missing references can also be considered again by using the *Consider Missing References* option. By default missing references are not ignored.

##### Set Marker in Tree

A right-click in the Data Tree window allows the user to set a marker within the Data Tree. This has the functionality similar to a bookmark and the user can return to this point in the Data Tree at any time by using the *Jump to Marker "..." in Tree*. Note that it is only possible to set one marker at a time; setting a new marker will automatically overwrite the last marker.

### 21.4.5.6 Diff window buttons

The various diff window buttons (as highlighted in Figure 21.4.4) will now be explained.

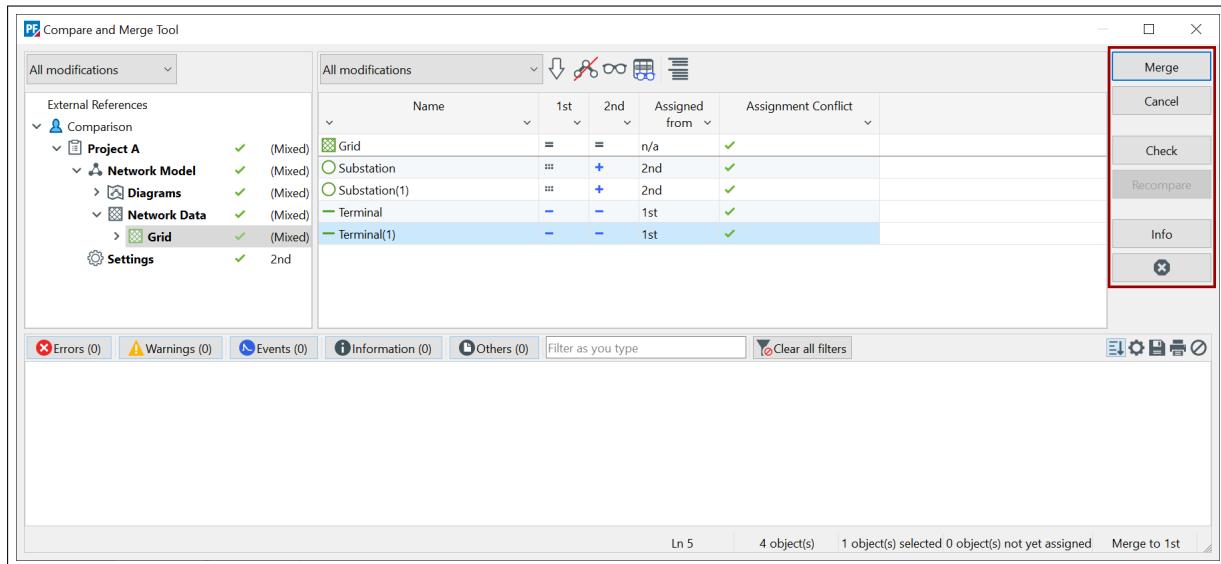


Figure 21.4.4: Compare and Merge Tool 'Diff window' with buttons highlighted

#### Check

This button checks that all assignments are okay. The following conflicts are checked for all compared objects:

- Missing assignment;
- Missing parent (parent object of an assigned object will not exist in the target after merge.)
- Missing reference (referenced object of an assigned object will not exist in the target after merge.)

All conflicts are printed as errors to the output window of the CMT. Conflicts are listed in groups and with the ! icon in the Data Tree and in the Comparison and Assignment window.

#### Recompare

After a *realignment*, it is necessary to run the CMT again using this button to update the comparison results.

#### Merge

The merge procedure updates the target by copying objects or parameters or by deleting objects according to the assignments.

Before the merge procedure is started, an assignment check is done. The merge procedure is cancelled if the check detects conflicts. If no conflicts are detected, the diff browser is closed and then the merge procedure is started.

After the merge procedure is complete, all data collected by the CMT is discarded.

## Info

The Info dialog called by the *Info* button shows more information about the comparison:

- Database path of the top-level projects/objects that are being compared;
- Target for merge (only if merge option is active);
- Selected comparison options;
- Number of objects compared;
- Number of objects modified; and
- Number of objects with conflicts (only if merge option is active).

## 21.5 How to Update a Project

There are two common procedures that users and data administrators need to complete when working with master projects and other user projects that are derived from versions of this master project:

- Updating a derived project with information from a new version; and
- Updating a master project with information from a derived project.

This section explains these two procedures and also provides *tips* for working with the CMT.

### 21.5.1 Updating a Derived Project from a new Version

When a derived project is activated after a new version of the *Base* project has been created (provided that the flag *Notify users of derived projects* was checked when the version was created and that the derived project option *Disable notification at activation* is unchecked), then the user will be presented with the dialog shown in Figure 21.5.1.

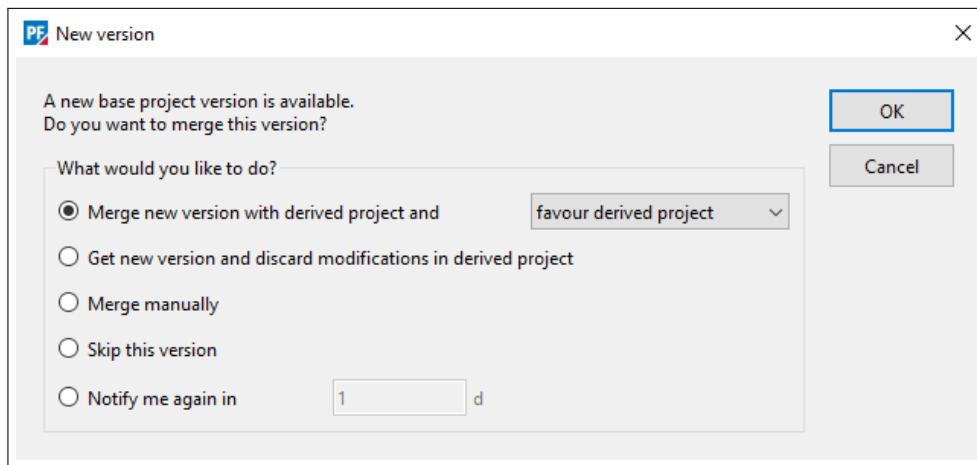


Figure 21.5.1: New version available dialog

The options offered in the notification dialog are:

- **Merge new version with derived project and.** PowerFactory automatically generates a temporary copy derived from the new version. It then executes a 3-way comparison with the base version of the user's project (as the base), the derived project (as <1st>) and the temporary copy (as <2nd> and target). In the case of a conflict, one of the following actions will be taken:

- ***favor none***: The CMT diff browser is displayed, and the user can then resolve the conflict(s) by defining how the changes should be assigned.
  - ***favor derived project***: Conflicts are resolved automatically by favouring the user's modifications, thereby discarding modifications in the base.
  - ***favor new version***: Conflicts are resolved automatically by favouring the base's modifications, thereby discarding the user's modifications.
- **Get new version and discard modifications in derived project.** The derived project is automatically replaced by the new version. All user modifications will be lost.
  - **Merge manually.** Use the CMT to merge the modifications manually. The results of the comparison are displayed in a CMT diff browser, where the user defines how the changes should be assigned. After these assignments have been defined, the new version and the derived project are merged to the temporary copy, when the user clicks on the *Merge* button. The derived project is then automatically replaced by the temporary copy (now containing information from the new version), which is deleted.
  - **Notify me again in....** The user enters the desired time for re-notification, and the derived project is activated according to how it was left in the previous session. The notification is deactivated for the indicated number of days.

---

**Note:** In a multi-user environment, updated versions of the *base* project can be released regularly and the user will often be presented with the new version notification in Figure 21.5.1. In many cases, the user will not want to apply the updated version because they will be in the middle of a project or other calculation and may not want to risk corrupting or changing their results. Therefore, the option *Notify me again in...* is the recommended choice because it will leave the user's project unchanged.

---

If the **Cancel** button is used, the project is activated as it was left in the previous session. The notification will appear following the next activation.

An alternative way to manually initiate the above procedure is to right-click on the derived project and select the option *Merge from base project*. This feature is only possible with deactivated projects.

## 21.5.2 Updating a Base Project from a Derived Project

Changes implemented in derived projects can also be merged to the base project. In this case, the option *Merge to base project* must be selected from the context menu available by right-clicking on the derived project. As in previous cases, the CMT is started and conflicts can be manually resolved using the *diff browser*.

## 21.5.3 Tips for Working with the Compare and Merge Tool

One of the most common uses of the CMT is for merging changes made by users to their derived projects back into the *master* project to create an updated version for all users. This kind of task is often performed by the data administrator. For this task it can help to follow the steps outlined below:

1. Check the user's modifications with a 2-way merge (derived vs. base; What changes were made? Are all changes intended? Modifications which were made by mistake should be corrected in the user's derived model before continuing with the merge procedure.). The check of the modifications should be done by the user and the data administrator.
2. The data administrator creates a new derived project based on the most recent version of the 'master' model.

3. A three-way merge is performed, selecting the version on which the user's derived project is based on as 'base', the derived project created in the previous step as <1st> and the user's derived project as <2nd>. The changes are merged into <1st> (target).
  4. The resulting model is then validated. Conflicts which could not be resolved automatically by the CMT are corrected manually.
  5. The validated model (derived project in data administrator account) is merged to the base model by using the context menu entry *Merge to Base Project*. This will not cause problems if the master model has not been changed since deriving the model in step 2.
  6. A new version is created by the data administrator and the users are informed.
- 

**Note:** The Compare and Merge Tool can be used to compare any kind of object within a *PowerFactory* project. The functionality and procedure to follow is similar to that explained in this section for project comparison and merging.

---

## 21.6 Sharing Projects

In *PowerFactory*, any project can be shared with other users according to the rules defined by its owner. Projects are shared with groups of users and not directly with individuals. Therefore, users must be part of a group (created and managed by the data administrator) in order to access shared projects.

Depending on the access level that the owner assigns to a group, other users can get:

- Read-only access to the shared project, which allows the copying of objects and the creation of derived projects from versions within the shared project;
- Read-write access, which allows users full control over all objects within the project.  
This includes project activation, but does not include the creation of versions.
- Full access, which allows the user to modify the sharing properties and create versions.

Each access level includes the rights of the lower levels.

Deletion of a project is only possible by the project owner.

To share a project:

1. Open the project dialog by right-clicking on the project name and selecting the option *Edit*.
2. Select the *Sharing* page;
3. Right-click within the *Groups* or *Sharing access level* columns on the right side of the *Sharing information* table to insert (or append) a row(s);
4. Double-click in the *Groups* cell of the new line and select the group with whom the project is shared using the data browser;
5. Double-click on the *Sharing access level* to select the desired access level.

A shared project is marked with the  symbol in the Data Manager. To display all the available users on the Data Manager, click on the *Show All Users* icon (). Only the shared projects of the other users will be displayed.

For information regarding user groups, refer to Chapter [PowerFactory Administration](#).

## 21.7 Combining Projects

The combination of projects is a two-stage process: first, the Project Combination Assistant is used to bring the two networks and all associated data into one project, then the Project Connection Assistant can be used to make connections between the two networks at known common points. Buttons which give easy access to these functions can be found in the “Additional Tools” toolbar.

---

**Note:** Within any individual *PowerFactory* project, if a foreign key is defined for any element (on the Description page of the element), that foreign key must be unique. However, when projects are to be combined it is quite possible that there will be duplication of foreign keys. This can be deliberate, to facilitate the connection process (see Section 21.7.2.2) but may also be coincidental. In either case, the situation is managed by prefixing all foreign keys with characters which make them unique. For example, a foreign key of “LA234” from the first project would be changed to “001:LA234”.

---

### 21.7.1 Project Combination Assistant

The Project Combination Assistant  can be used either to combine two or more projects into one new project, or to incorporate further projects into an already active project.

#### 21.7.1.1 New Combined Project

Before starting the combination process, it is first necessary to ensure that the source projects all have a Version defined; the user will need to specify which Version is to be used when the combination process is executed. Please see Section 21.2 for more information about versions. It may also be worth thinking about how the networks will be connected in the next step, to ensure that the necessary data configuration has been made, although this can also be done after the two projects are combined.

To start the project combination, first of all ensure that there is no project active, then bring up the Project Combination Assistant tool, either via the icon on the Additional Tools toolbar, or via the *File* → *New* → *Combined Project*... option from the main menu, or by right-clicking on the user name in Data Manager, then *New* → *Combined Project*....

A list of projects is then made by adding project and version references. Once this has been done, the process is run by pressing the Execute button. The new project is created and activated.

#### 21.7.1.2 Structure of Combined Project

The resultant structure of the combined project can be seen in Figure 21.7.1, below. In this example two projects have been combined.

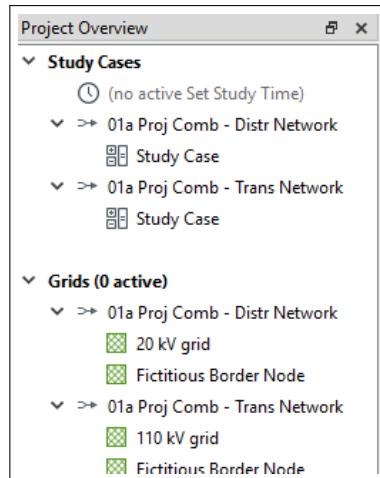


Figure 21.7.1: Structure of combined project

The folders seen in the figure, which are automatically created, take the names of the source projects and allow the user to clearly identify where each object came from. Initially no study cases or grids are active. The user can now use the Project Connection Assistant (21.7.2) to establish the links between the two networks.

### 21.7.1.3 Incorporating additional Projects

Another possibility for combining projects is to start with an active project and incorporate an additional project or projects into it. In this case, the target project must be active but there should be no study case active. The Project Combination Assistant is then launched using the button on the Additional Tools toolbar. As described above, the source project is then specified, including the required version.

## 21.7.2 Project Connection Assistant

The Project Connection Assistant  offers two methods for automatically making the connections between two networks at the common points in their grids: connection via terminals, or connection via switches.

Following the process of creating a new project described in Section 21.7.1.1, a new active study case must first be created before the connection process can start. To do this, right-click on one of the source study cases and select the option *Apply network state*. The same can be done on the other source study case(s), in which case all the settings of those study cases, i.e. the active scenarios, variations and the summary grid will also be copied to the currently active Study Case. These settings might affect the network topology so it can make a difference for the connection algorithm if they are not added, but it is optional.

At this point it is still possible to make any adjustments to the data (e.g. foreign keys or node names) to ensure that the next step runs smoothly.

### 21.7.2.1 Connection Using Common Terminals

With this approach, the common points in the two networks are identified as nodes, i.e terminal elements *ElmTerm*. In each of the source projects, all the relevant *ElmTerm* objects should be separated into a designated grid; in the example in Figure 21.7.1, the grid is called Fictitious Border Node. There is no restriction on the name of the grid but the same name must be used in each source project. The

default method by which the nodes within the connecting grids are matched up is to use the node names (*loc\_name*). However the user can specify an alternative parameter such as the CIM RDF id (*cimRdfId*).

The Project Connection Assistant is launched from the Additional Tools toolbar. It should be noted that a new Variation will be created during the process, which will record all the changes made. The user then selects the connection method “by virtual nodes” from the drop-down menu and specifies the virtual nodes grid. It doesn’t matter which of the virtual nodes grid is selected; the tool searches for all grids of this name.

When the Execute button is pressed, the matching nodes in the various virtual node grids are consolidated into new terminals in a new virtual node grid. The source virtual node grids are deactivated.

### 21.7.2.2 Connection Using Elements with Foreign Keys

As an alternative to specifying terminals as connection points, it is possible to identify the connection points as elements with identical foreign keys. Currently, only switch elements are permitted for this process. Such switches must only be connected to one terminal, indicating that the other side is available for connection. If necessary, the terminal on the outer side of the switch can simply be deleted.

The connection tool searches from switches which have the same foreign key in the two models and will then execute a connection process using any switches connected only on one side. The process involves the removal of one switch and connecting the other switch in its place, as shown in Figure 21.7.2. The switch is always left open after this process.

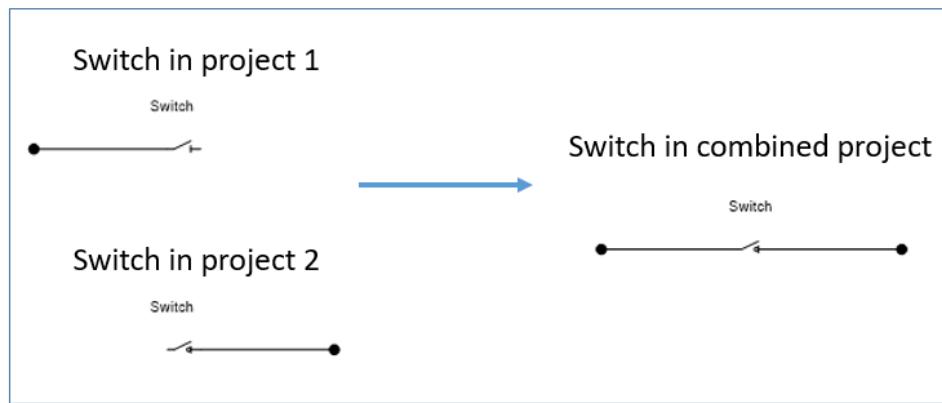


Figure 21.7.2: Network connection using paired switches

To carry out the connection process, the Project Connection Assistant is launched from the Additional Tools toolbar. The user should select the connection method “by foreign key” from the drop-down menu, then press Execute.

Once the connection process is complete, a report is presented to the user in tabular format, listing the matching switches. Any switches found which had identical foreign keys but could not be connected because they were already connected to terminals at both sides will be highlighted.

### 21.7.3 Final Project State

Once the project combination and connection processes are complete, the result will be a useable project for the whole network. The project structure, with separate folders for components originating from the different source projects, will remain. The connection activities are captured in variations, which can be deactivated if the user wishes to see the state before connection. As mentioned above, foreign keys will have been modified to avoid duplication.

The user should review the connected network to ensure that it correctly represents the desired state. For example, it may be necessary to remove load objects that previously represented lower voltage networks which are now modelled. If connections were made using terminals, the user should consider whether changes need to be made as a result of the original terminals being deleted, for example check that station controllers which referenced terminals in the virtual node grid are now pointing to the new terminals at the interface points.

If it is found that the networks have not been connected as expected, the user should check carefully that the names or foreign keys of the matching elements are precisely the same (remembering that these names/keys are case-sensitive).

Before executing any calculations, the user should be aware that when projects are combined no command objects are copied from the source projects (as there would be no way of knowing which are the preferred settings). Any calculations will therefore initially take the default settings.

#### 21.7.4 Project Normalisation

As described in Section [21.7.1.2](#), the contents of the project are organised into dedicated folders, making it easy to identify the original sources of the different data items. In some cases, users would like to take a further step of completely integrating the data into a single folder structure. Although it is possible to do this “normalisation” manually, there is also a tool provided, to do this very easily:

- Deactivate the combined project
- Edit the project and go to the Combined Project page
- Click on the **Normalise** button

### 21.8 Database Archiving

An archiving function for decreasing the used database storage space and increasing performance of large multi-user databases is available. Older projects that are currently not used, but which are still important for possible future use can be archived.

Archiving describes the process of automatically exporting and deleting projects from the database and storing them in a restorable way in the file system. The actual workload is shifted to the housekeeping job which can be run overnight, where export and delete operations do not interfere with the users. Archiving can either be done by the user selecting a project for archiving, or by using DPL scripts.

In multi-user database environments, the user can easily send projects to the archive folder via the context menu (right-click) for each project, and selecting “Archive”. The archived projects are exported from the database and are placed in a separate folder (“Archived Projects”) for long-term storage. The user thereby increases system performance and the speed of general database operations (e.g. project loading/closing). All information regarding the initial project location is also saved allowing the user to restore projects to the exact location from which they originated.

Projects can be restored into the active database by executing the “Restore” command in the context-sensitive (right-click) menu of each project.

For more information on this topic, see Chapter [PowerFactory Installation and Configuration](#), Section [Housekeeping](#).

# Chapter 22

## Task Automation

### 22.1 Introduction

The Task Automation command (*ComTasks*) enables the *PowerFactory* user to run a list of various tasks ranging from specific *PowerFactory* power system analysis calculation functions up to generic data handling/processing jobs (via scripts) in parallel or sequentially. Using this command it is possible to execute tasks defined in multiple study cases (with any number of calculation commands per case) or multiple independent calculation commands (organised within a single study case). The *Parallel Computation* feature makes full use of a host machine with multi-core processor architecture.

To successfully execute the *Task Automation* command the user first needs to configure a list of calculation functions (e.g. *ComLdf*, *ComSim*) or scripts (e.g. *ComDpl*, *ComPython*) for every designated study case and then *PowerFactory* processes automatically the assigned tasks. Depending on the selected configuration options, a task may represent one study case or one calculation command within a study case (refer to Section 22.2.3 for more information). Most calculation commands can be used within the Task Automation tool given that the specific command actions are acting on the same *PowerFactory* project data. Generally speaking, a calculation command is designated in *PowerFactory* by the object class prefix “Com”.

Task Automation offers enhanced possibilities for power network studies execution, with examples such as:

- Already developed *PowerFactory* projects containing complete power grid analyses which are organised in various study cases, as is usually the case, can be directly used for parallel computation;
- Calculation intensive dynamic simulations can be configured with individual simulation events / operation scenarios by creating multiple study cases. Then, the list of study cases can be passed to the Task Automation command for parallel computation.

For information on how to configure the *Task Automation* command, refer to Section 22.2.

For information on how to locate and manage the results generated by the *Task Automation* command, refer to Section 22.3.

### 22.2 Configuration of Task Automation

A new Task Automation command can be created via:

- the Menu Bar: Click on *Calculation* → *Additional Tools*→ *Task Automation...*

- the Main Toolbar:
  - Click on *Change Toolbox* icon (▼) and select the *Additional Tools* toolbox
  - Click on *Task Automation* icon (⌚)
- the Data Manager:
  - With a project active, open the Data Manager and click on the *Study Cases* project folder
  - From the Data Manager toolbar click on the *New Object* icon (✚)
  - Select Task Automation (*ComTasks*) from the list and click on the **OK** button.
- Scripting, by creating a *Task Automation* object (*ComTasks*). Special care needs to be taken when configuring the *Task Automation* object using a script in the sense that assigning references (e.g. study cases or specific commands) to the *Basic Options* page fields (e.g. *vecCases* and *curTasks*) is done via the dedicated DPL functions described in the [DPL Reference](#) document.

The Task Automation command dialog has the following pages:

- Study Cases
- Tasks
- Parallel Computing
- Output

### 22.2.1 Study Cases Page

**Selection of study cases:** This dialog pane contains a list of existing study cases that may be considered for the Task Automation command. Study Cases can be added to the list via the **Add** button. The **Remove all** button removes all items within the current list. The checkboxes in the *Ignore* column exclude a specific study case from the cases being considered by the calculation without removing it from the list.

### 22.2.2 Tasks Page

**Selection of commands/additional results:** This dialog pane stores information on the calculation commands (*Com\**) to be executed by the Task Automation for each study case that is added to the *Selection of study cases* list as previously described. The commands list is unique for each study case. The currently shown list is valid for one specific study case as selected via the drop down menu *Study case*. Calculation commands (*Com\**) can be added to the list either via the **Add from all study cases** or the **Add from selected study case** button. This gives the option to add commands from a specific or from several study cases. As a prerequisite, each selected command must be located within the according study case folder. The *Ignore* checkbox excludes a specific command from the list without removing its entry.

**Additional Results:** Several commands generate results files during their execution, such as for example the Contingency Analysis command (*ComSimoutage*). Others, like a conventional load flow calculation (*ComLdf*), do not, while the results are stored temporarily in the memory. To address this latter case, the user can choose to write an additional results file per command (by ticking the checkbox in the *Additional results* column). Variables that shall be recorded in that results file after the execution of the command can be configured by double-clicking/right-clicking on the corresponding cell of the *Result variables* column. Preconfigured result files (i.e. their Variable Selections) can also be copied from one command defined in the table to another. Whether the Variable Selections are overwritten or appended can be specified by the appropriate drop down menu under *Action on pasting result variable selection*.

**Results:** This field reference defines the folder where the additional results files of the currently selected study case will be located after the Task Automation command is executed. Moreover, this folder will

contain references to all results files that have been generated by a calculation command within this study case.

---

**Note:** There is one Results-folder per study case. The shown field reference corresponds to the currently selected study case as shown in the drop down menu *Study case*.

---

**Action on pasting result variable selection:** The according drop-down menu defines how to proceed when pasting additional result variables from one command to another. The two available options are *Overwrite* and *Append*.

**Show:** Via the **All tasks** button all selected tasks from all study cases can be shown. The button **All selected commands** only shows the according calculation commands (*Com\**) without possible duplications (if a command is selected several times in a study case). Within the *All tasks* selection also the additional results and result variables can be configured.

### 22.2.3 Parallel Computing Page

The Task Automation command can be executed sequentially, thereby processing command after command, or in parallel mode, using the built-in process parallelisation algorithm.

The following settings are subject to user configuration:

**Parallel computation:** By ticking this checkbox, the user switches from the sequential execution to the parallel task processing; by unchecking it, the sequential execution of tasks is adopted. If parallel computation is selected, a minimum number of tasks can be specified via the setting *Minimum number of packages*. If the user selects fewer tasks than this number, the parallelisation will be executed sequentially.

Clicking the *Edit* button (→) next to the *Parallel computation settings* reference, will open the user settings on the *Parallel Computing page*. Here the user can define the number of used processes and, on the *Advanced* tab, the Max. waiting time for process response. For large projects in particular, it may be necessary to extend this time. For more information about the parallel computing settings within the user see Section 7.11.

An internal copy of the project is used for each parallel process. By default, to improve performance, only the required data is transferred to the virtual copy of the project, and although the program logic is designed to collect all the necessary data, it may be the case that some data is not transferred correctly. This can either be because it is outside the project, or because the project has very different structure from that of a standard project. To avoid missing relevant data, additional folders can be selected with the *Additional object to transfer* option.

The general configuration of the parallel computation via the Parallel Computation Manager is described in Section 6.6.1.

The use of the Parallel Computing feature is dependent on the particular *PowerFactory* user settings as defined via the *PowerFactory* Administrator account. Enabling *Parallel Computing* for a particular user is achieved by following the procedure below:

- Log in *PowerFactory* using the Administrator account;
  - Within the Data Manager, open the specific *PowerFactory* user account edit dialog;
  - Click on the *Parallel Computing* page;
  - Tick the *Allow Parallel Computing* checkbox.
-

**Note:** If a user is not allowed to perform a parallel computation an info-message is displayed in the *Parallel Computing* page.

---

**Distribute packages:** The radio-button *Distribute packages* determines the definition of a task (package) in the context of distribution of tasks to the parallel processes:

- If *By study case* is selected, a task is defined as a study case and all commands configured for a specific study case are processed sequentially by the Task Automation command within a single parallel process. This setting is handy when commands belonging to one study case list depend on each other.
  - If *By command* is selected, a task is defined as an individual command. Every command is executed by the Task Automation independently of the other commands within the same study case. Furthermore, commands within the same study case may be queued for execution in different parallel processes in order to maximise performance. This option can be used when commands are independent of each other.
- 

**Note:** The *Distribute packages* option is disabled for sequential execution of the Task Automation. In this case, the option *By study case* is always chosen and commands are executed in the defined order (as specified in the *Selection of commands/additional results* list). If the option *By study case* is chosen for parallel computation, different study cases are assigned to different parallel processes. In particular, the execution of a command in a later study case in the list should not rely on the execution of commands in a previous study case.

---

**Transfer of database changes to master process:** this radio-button defines, which data shall be transferred from the parallel process to the master process and merged into the database:

- If *All* is selected, all changes that have been made within the parallel process are transferred to the master and merged into the database. In this case the changes of the database correspond to a sequential execution.
- 

**Note:** This option should be used with caution, especially if the tasks to be performed result in network changes that are recorded in the base network. The changes made by each process are transferred successively and, depending on the parallelisation configuration, the results of the processes may be affected.

---

- If *Only result files* is selected, the parallel processes will run in read-only mode. That means, all modifications are temporarily stored in the internal memory of the computer. After a parallel process has finished, only the results (i.e. pointers to results files) are transferred to the master process, to be written back into the database. In addition to the advantage that the database is not changed by the parallel processes in this way, the amount of data, which has to be transferred to the master process, is significantly reduced. This results in a performance increase.
  - The *Result files and selected objects* option works similar to the before mentioned option. But in addition to the result file also other defined objects are transferred back. This can be modified operation scenarios, network elements or other objects that were modified during the parallel process and should be used later on. The according objects are set under the *Selection* option. Synchronisation of selected objects should be used with caution and each object should only be changed by one parallel process to ensure a deterministic project state.
  - The transfer of the data, either all changes or only the results file, is performed after the time interval defined in the *Database synchronisation time* field. If the option is unchecked, the data will be transferred after **all** the calculations in the parallel process have been completed.
-

**Note:** The *Database changes of parallel processes* option will only be available if a parallel calculation is possible (*Parallel computation* box is checked and settings of the Task Automation allow a parallel execution → emphasised by the blue text in the *Parallel computation* field). For a sequential execution (emphasised by the red text in the *Parallel computation* field) the *Database changes of parallel processes* option will be disabled.

---

## 22.2.4 Output Page

**Output per package** defines the behaviour of the Task Automation command with respect to Output Window reporting. The *Output per package* radio button has the following settings subject to user configuration:

- **Detailed calculation status** The behaviour of this option is dependent on the task execution mode:
  - Sequential Task Execution: All messages of executed commands are shown in the output window.
  - Parallel Task Execution: A message is issued when the calculation of a task starts and one on success or failure at the task end. Details about which command failed in the task are additionally issued.
- **Short (only issue errors)** The behaviour of this option is dependent on the task execution mode:
  - Sequential Task Execution: Only errors issued during the calculation command execution are displayed.
  - Parallel Task Execution: one message is issued when the calculation of a task starts and another one at the calculation end (reporting execution success or failure).

## 22.3 Task Automation Results

The *Task Automation* executes a series of commands either sequentially or in parallel using different local machine processor cores and parallel processes. Therefore, there will be no single set of results readily available after executing the *Task Automation* command. The results of an individual command (i.e. from the *Selection of commands/additional results* list) are recorded during its execution or right after it finished by means of an additional results file.

The available tools for obtaining results from the Task Automation command are summarised below:

- The calculation status of individual commands is issued in the output window during the task processing as described in Section 22.2.4. Moreover, there is an error summary of all failed commands per study case printed to the output window at the end of executing the Task Automation command.
- Beside results files created during the execution of individual commands, additional results files can be defined which are created after the individual command execution. Pre-defined variables can be recorded, as shown in Section 22.2.1. Note that all such results files together with references to results files generated during the calculation are added to a results folder per study case (as defined in the Task Automation command).
- Access all these results files in a summarised tree-structure manner, where the icon *Task Automation - Show results* ( ) can be used. The icon is available from the main toolbar, *Additional Tools* toolbox, next to the Task Automation command.
- Result log files are created for each parallel process. The log files are saved under the *PowerFactory* workspace folder, “db-process-1\log” subfolder (e.g. C:\Users\MyUser\AppData\Local\DIgSILENT\PowerFactory\Workspace.nnnnnnnn\db-process-1\log). These files provide further information on the execution details of each parallel process.

# Chapter 23

## Scripting

This chapter describes the options available for scripting in *PowerFactory*, which are based around two programming languages: the *DIGSILENT* Programming Language **DPL** and Python.

Section 23.1 looks at the in-built programming language DPL and Section 23.2 shows how this can be used to build tabular reports. Section 23.3 introduces the open source programming language **Python**.

The remaining two sections of the chapter provide information about the text editor used for scripting and the concept of Add On Modules, which can be used with either DPL or Python.

### 23.1 The *DIGSILENT* Programming Language - DPL

The *DIGSILENT* Programming Language **DPL** serves the purpose of offering an interface for automating tasks in the *PowerFactory* program. The DPL method distinguishes itself from the command batch method in several aspects:

- DPL offers decision and flow commands
- DPL offers the definition and use of user-defined variables
- DPL has a flexible interface for input-output and for accessing objects
- DPL offers mathematical expressions

The DPL adds a new dimension to the *DIGSILENT PowerFactory* program by allowing the creation of new calculation functions. Such user-defined calculation commands can be used in all areas of power system analysis, such as

- Network optimising
- Cable-sizing
- Protection coordination
- Stability analysis
- Parametric sweep analysis
- Contingency analysis
- etc.

Such new calculation functions are written as program scripts which may use

- Flow commands like “if-then-else” and “do-while”

- *PowerFactory* commands (i.e. load-flow or short-circuit commands)
- Input and output routines
- Mathematical expressions
- *PowerFactory* object procedure calls
- Subroutine calls

### 23.1.1 The Principle Structure of a DPL Command

The principle structure of a DPL script is shown in Figure 23.1.1.

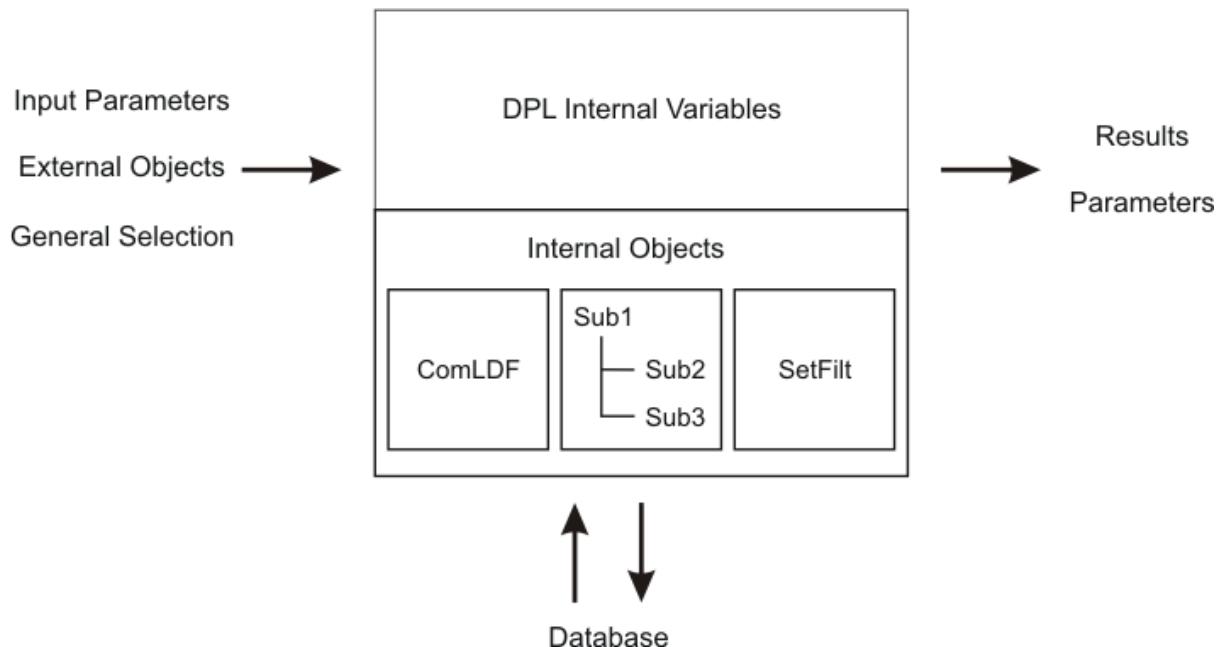


Figure 23.1.1: Principle structure of a DPL command

The DPL command *ComDpl* is the central element, which connects different parameters, variables or objects to various functions or internal elements and then outputs results or changes parameters.

The input to the script can be predefined input parameters, single objects from the single line diagram or the database or a set of objects/elements, which are then stored inside a so called “General Selection”.

This input information can then be evaluated using functions and internal variables inside the script. Also internal objects can be used and executed, like

- a calculation command, i.e. *ComLdf*, *ComSim*, etc., especially defined with certain calculation options
- subscripts also released in DPL
- filter sets, which can be executed during the operation of the script

Thus the DPL script will run a series of operations and start calculations or other functions inside the script. It will always communicate with the database and will store changed settings, parameters or results directly in the database objects. Almost every object inside the active project can be accessed and altered.

A DPL script can perform a large number of calculations in a loop. This can be used to analyse the sensitivity of element or type parameters to load flow, short circuit or other calculations. However, this

iterative looping is not fully supported for all parameters, some parameters (such as rated voltage, rated current, rated apparent power of elements and types) are stored at the beginning (first execution of the calculation) and are only read again if a reset (`ResetCalculation()`) is forced. This behaviour has been implemented to improve the speed of DPL. The not supported parameters are parameters used to build the per unit system for the calculation.

During or at the end of the execution of the DPL script, the results can be output or parameters of elements may be changed. There is the possibility to execute a predefined output command `ComSh` or to define one's own outputs with the DPL commands available.

## 23.1.2 The DPL Command

The DPL command element `ComDpl` contains the script code (or a reference to a so called remote script), the definition of input and output parameters, a description and information about versions. DPL command objects can therefore be divided into:

- Root commands, which have their own scripts on the *Script* page of the dialog.
- Referring commands, which use the scripts of remote DPL commands by only adapting input and output parameters and external objects.

### 23.1.2.1 Creating a new DPL Command

A DPL Command `ComDpl` can be created by using the *New Object* (⊕) icon in the toolbar of the Data Manager and using the filter to select *DPL Command* from the list. After pressing **OK**, a new DPL command is created.

The dialog is now shown and the parameters, objects and the script can now be specified.

This dialog is also opened by double-clicking a DPL script, by selecting *Edit* from the context menu or by selecting the script from the list when pressing the icon (⊖).

### 23.1.2.2 Defining a DPL Commands Set

The DPL command holds a reference to a selection of objects (*General Selection*). At first this general selection is empty, but there are several ways to define a special set of object used in the DPL command. This “DPL Commands Set” (`SetSelect`) can be specified through:

- Select one or more elements in the single line diagram. Then right-click the selection (one of the selected elements) and choose the option *Selections* → *Set - DPL Commands* → *New...* from the context menu.
- It is also possible to select several elements in the Data Manager. Right-click the selection and choose the option *Selections* → *Set - DPL Commands* → *New...* from the context menu.

### 23.1.2.3 Executing a DPL Command

To execute a DPL command or to access the dialog of a script, the icon (⊖) can be activated. This will pop up a list of available DPL and Python scripts from the global and local libraries.

The easiest way to start a DPL command AND define a selection for it is to:

- Select one or more elements in the single line diagram or in the Data Manager and then right-click the selection.

- Choose the option *Execute Script* from the context menu.
  - Then select a DPL script from the list. This list will show DPL scripts from the global libraries, as well as from the local library.
  - Select a DPL script, insert/change the variables and then press the button **Execute**
- 

**Note:** The custom global library will only be shown if the library is set as *Used Library* in the User Settings (see chapter User Settings, Section 7.2).

---

In this way the selection is combined into a **DPL Commands Set** and the set is automatically selected for the script chosen.

Only one single DPL command set is valid at a time for all DPL scripts. This means that setting the DPL command set in one DPL command dialog, will change the DPL command set for all DPL commands in the database.

---

**Note:** To choose different sets for various DPL scripts you can either use different selection object *SetSelect* like the “General Set”. Or new DPL command sets can be created and selected inside the active study case.

This is done by pressing the *New Object* icon , and using the filter to select the object Set (*SetSelect*).

---

The interface section *Input Parameters* is used to define variables that are accessible from outside the DPL command itself. DPL commands that call other DPL commands as subroutines, may use and change the values of the interface variables of these DPL subroutines.

The list of *External Objects* is used to execute the DPL command for specific objects. A DPL command that, for example, searches the set of lines for which a short-circuit causes too deep a voltage dip at a specific busbar, would access that specific busbar as an external object. Performing the same command for another busbar would then only require setting the external object to the other busbar.

#### 23.1.2.4 Results

On this page, the *Result parameters* can be defined. These parameters are results from the script and they are stored inside the results object. Hence it is possible to access them through the variable monitor and display them in a plot. In addition to the value itself, the name, the type (if a string, object or number), the unit and the parameter description can be entered.

#### 23.1.2.5 DPL Script Page

The most important part of a DPL command is of course the DPL script code. That script is written on the *Script* page of the DPL command dialog. As an alternative to writing the code directly, the command can reference an existing script by selecting it as a *Remote script*.

On this page the DPL code of an already defined script is shown and/or new command lines can be inserted for modifying this script or writing a new script. The available commands and the DPL language are described in the following sections.

The edited program code also features highlighting specially suited for handling DPL scripts.

### 23.1.2.6 DPL Script Encryption

*PowerFactory* offers the possibility to encrypt the script code of a DPL command. The encryption action can be initiated by pressing the corresponding button in the edit dialog of the DPL command object (does not work for commands with a remote script referenced). The encryption process then asks in a dialog for a password and its confirmation. The password is only needed to decrypt the script at a later stage. The encrypted script can be executed without entering the password. After completing the encryption with **OK**, the code is hidden and only the name of the command itself, the values of the input parameters and external objects can be changed. If there are subscripts stored as contents, they will be encrypted with the same password, too.

---

**Note:** The user should be aware that encryption can never guarantee complete security. The chosen technology balances the requirements for security with the usability and performance of encrypted models. Generally, users are advised to share models only with trusted partners.

---

The encryption is reversible; an encrypted script can be decrypted using the corresponding button in the edit dialog of the encrypted ComDpl-object. After entering the password and confirming with **OK**, the script returns to its original status, where all properties may be changed and the script code is shown.

---

**Note:** The encrypt-action affects the script for which it is executed and does not create an encrypted copy of the ComDpl-object.

---

### 23.1.3 The DPL Script Editor

The *Script* page of the DPL command includes a built-in editor based on the *Scintilla* editing component (<http://www.scintilla.org>), which offers the following features:

- **Auto-completion:** when typing a new word, a list of suitable suggestions for keywords, global functions, variable names (defined within the current script or as input/result variables) and subscripts will pop up. Using the arrow keys the user may explore all suggestions, and insert the currently selected suggestion. The autocompletion can be deactivated in the editor settings.
- 

**Note:** The suggestion lists do not contain deprecated names.

---

- **Bracket match checking:** when the cursor stands before an opening or closing bracket, the editor will check if the brace is matched. If it is, the bracket and its partner are highlighted in blue. If the bracket, however, is not matched, it will be highlighted in red.
- **Automatic bracket insertion:** when typing in an opening bracket, the editor will automatically insert a matching closing bracket and position the caret between the two brackets. Additionally, if the caret stands before a matched closing bracket, typing a closing bracket of the same type will simply result in the caret moving forwards. This helps users who are not familiar with automatic bracket insertion avoid inserting unnecessary additional closing brackets.
- **Automatic quote character insertion:** similar to automatic bracket insertion; when typing in a single quote character ('), the editor will automatically insert an additional single quote character and position the caret between the two quote characters. Additionally, if the caret stands before a quote character, typing a quote character of the same type will simply result in the caret moving forwards.
- **Zoom-in/Zoom-out:** using the key combination **Ctrl + Mousewheel** will increase or decrease the zoom. Note that this only temporarily modifies the used font size and has no effect at all on the font size that the user chose in the editor font settings. The key combination **Ctrl + 0** restores the font to its original size.

- **Selection highlighting:** whenever text is selected (not counting column selections and selections that span more than one line), all occurrences of the selected text in the current document are lightly highlighted using the last known search settings.
- **Instance-independent search terms and search settings:** whenever the user opens the find (or find/replace) dialog, the chosen search term and search settings are used in every open editor component. This enables users to search the same term with the same settings in multiple documents without having to call the find (or find/replace) dialog for each one of them.
- **Advanced syntax styling:** the script will be coloured according to this scheme: keywords are blue, (recognised) global function and method names are light blue, string literals are red, number literals are turquoise, operators are light brown, identifiers are dark blue, comments are green.

To open the editor (23.4) in an additional window press the icon  on the bottom side of the *Script* page of the DPL Command dialog. Note that when the script is opened in an additional window, it cannot be edited via the DPL Command.

### 23.1.4 The DPL Script Language

The DPL script language uses a syntax quite similar to the C++ programming language. This type of language is intuitive, easy to read, and easy to learn. The basic command set has been kept as small as possible.

The syntax can be divided into the following parts:

- variable definitions
- assignments and expressions
- program flow instructions
- method calls

The statements in a DPL script are separated by semicolons. Statements are grouped together by braces. Example:

```
statement1;
statement2;
if (condition) {
    groupstatement1;
    groupstatement2;
}
```

#### 23.1.4.1 Variable Definitions

DPL uses the following internal parameter types

- **double**, a 15 digits real number
- **int**, an integer number
- **string**, a string
- **object**, a reference to a *PowerFactory* object
- **set**, a container of objects

Vectors and Matrices are available as external objects.

The syntax for defining variables is as follows:

```
[VARDEF] = [TYPE] varname, varname, ..., varname;
[TYPE]   = double | int | object | set
```

All parameter declarations must be given together in the top first lines of the DPL script. The semicolon is obligatory.

Examples:

```
double Losses, Length, Pgen;
int NrOfBreakers, i, j;
string txt1, nm1, nm2;
object O1, O2, BestSwitchToOpen;
set AllSwitches, AllBars;
```

#### 23.1.4.2 Constant parameters

DPL uses constant parameters which cannot be changed. It is therefore not accepted to assign a value to these variables. Doing so will lead to an error message.

The following constants variables are defined in the DPL syntax:

**SEL** is the general DPL selection

**NULL** is the “null” object

**this** is the DPL command itself

Besides these global constants, all internal and external objects are constant too.

#### 23.1.4.3 Assignments and Expressions

The following syntax is used to assign a value to a variable:

```
variable = expression;
variable += expression;
variable -= expression;
```

The add-assignment “`+=`” adds the right side value to the variable and the subtract-assignment “`-=`” subtracts the right-side value.

Examples:

```
double x,y;
x = 0.5*pi();      ! x now equals 1.5708
y = sin(x);        ! y now equals 1.0
x += y;            ! x now equals 2.5708
y -= x;            ! y now equals -1.5708
```

#### 23.1.4.4 Standard Functions

The following operators and functions are available:

- Arithmetic operators: `+`, `-`, `*`, `/`
- Standard functions ( all trigonometric functions based on radians (RAD))

function	description	example
<b>sin(x)</b>	sine	sin(1.2)=0.93203
<b>cos(x)</b>	cosine	cos(1.2)=0.36236
<b>tan(x)</b>	tangent	tan(1.2)=2.57215
<b>asin(x)</b>	arcsine	asin(0.93203)=1.2
<b>acos(x)</b>	arccosine	acos(0.36236)=1.2
<b>atan(x)</b>	arctangent	atan(2.57215)=1.2
<b>sinh(x)</b>	hyperbolic sine	sinh(1.5708)=2.3013
<b>cosh(x)</b>	hyperbolic cosine	cosh(1.5708)=2.5092
<b>tanh(x)</b>	hyperbolic tangent	tanh(0.7616)=1.0000
<b>exp(x)</b>	exponential value	exp(1.0)=2.718281
<b>ln(x)</b>	natural logarithm	ln(2.718281)=1.0
<b>log(x)</b>	log10	log(100)=2
<b>sqrt(x)</b>	square root	sqrt(9.5)=3.0822
<b>sqr(x)</b>	power of 2	sqr(3.0822)=9.5
<b>pow(x,y)</b>	power of y	pow(2.5, 3.4)=22.5422
<b>abs(x)</b>	absolute value	abs(-2.34)=2.34
<b>min(x,y)</b>	smaller value	min(6.4, 1.5)=1.5
<b>max(x,y)</b>	larger value	max(6.4, 1.5)=6.4
<b>modulo(x,y)</b>	remainder of x/y	modulo(15.6,3.4)=2
<b>trunc(x)</b>	integral part	trunc(-4.58823)=-4.0000
<b>frac(x)</b>	fractional part	frac(-4.58823)=-0.58823
<b>round(x)</b>	closest integer	round(1.65)=2.000
<b>ceil(x)</b>	smallest larger integer	ceil(1.15)=2.000
<b>floor(x)</b>	largest smaller integer	floor(1.78)=1.000

Table 23.1.1: DPL Standard Functions

- Constants:

pi()	pi
twopi()	2 pi
e()	e

Table 23.1.2: DPL Internal Constants

#### 23.1.4.5 Program Flow Instructions

The following flow commands are available.

```
if ( [boolexpr] ) [statlist]
if ( [boolexpr] ) [statlist] else [statlist]
do [statlist] while ( [boolexpr] )
while ( [boolexpr] ) [statlist]
for ( statement ; [boolexpr] ; statement ) [statlist]
```

in which

```
[boolexpr] = expression [boolcomp] expression
[boolcomp] = "<" | ">" | "=" | "<=" | ">=" | "<>"
[statlist] = statement; | { statement; [statlist] }
```

- Unary operators: “.not.”
- Binary operators: “.and.” | “.or.” | “.nand.” | “.nor.” | “.eor.”

- Parentheses: {logical expression}

**Examples:**

```

if (a<3) {
    b = a*2;
}
else {
    b = a/2;
}
while (sin(a)>=b*c) {
    a = 0:dline;
    c = c + delta;
}
if ({.not.a}.and.{b<>3}) {
    err = Ldf.Execute();
    if (err) {
        Ldf:iopt_lev = 1;
        err = Ldf.Execute();
        Ldf:iopt_lev = 0;
    }
}
for (i = 0; i < 10; i = i+1){
    x = x + i;
}
for (o=s.First(); o; o=s.Next()) {
    o.ShowFullName();
}

```

### Break and Continue

The loop statements “do-while”, “while-do” and “for” may contain “break” and “continue” commands. The “break” and “continue” commands may not appear outside a loop statement.

The “break” command terminates the smallest enclosing “do-while”, “while-do” or “for” statement. The execution of the DPL script will continue with the first command following the loop statement.

The “continue” command skips the execution of the following statements in the smallest enclosing “do-while”, “while-do” or “for” statement. The execution of the DPL script is continued with the evaluation of the boolean expression of the loop statement. The loop statement list will be executed again when the expression evaluates to TRUE. Otherwise the loop statement is ended and the execution will continue with the first command following the loop statement.

**Example:**

```

O1 = S1.First();
while (O1) {
    O1.Open();
    err = Ldf.Execute();
    if (err) {
        ! skip this one
        O1 = S1.Next;
        continue;
    }
    O2 = S2.First();
    AllOk = 1;
    DoReport(0); !reset
    while (O2) {
        err = Ldf.Execute();
        if (err) {
            ! do not continue
            AllOk = 0;
            break;
        }
    }
}

```

```
        }
    else {
        DoReport(1); ! add
    }
O2 = S2.Next();
}
if (AllOk) {
    DoReport(2); ! report
}
O1 = S1.Next();
}
```

### 23.1.4.6 Input and Output

The “input” command asks the user to enter a value.

```
input(var, string);
```

The input command will pop up a window with the string and an input line on which the user may enter a value. The value will be assigned to the variable “var”.

The “printf” command can be used to write text to the output window.

```
printf(string);
```

The string may contain “=–” signs, followed by a variable name. The variable name will then be replaced by the variable’s value.

Example:

```
double diameter;
input(diameter, 'enter diameter');
printf('the entered value = %f', diameter);
```

The example results in the pop up of a window as depicted in Figure 23.1.2.

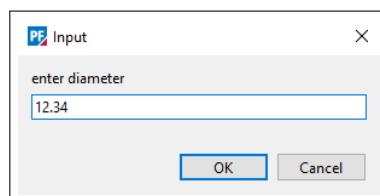


Figure 23.1.2: The input window

The following text will appear in the output window:

```
the entered value = 12.3400
```

Refer to the [DPL Reference](#) for more information about the printf command.

### 23.1.5 Access to Other Objects

With the syntax for the parameter definitions, program flow and the input and printf, it is already possible to create a small program. However, such a script would not be able to use or manipulate variables of “external” objects. It would not be possible, for instance, to write a script that replaces a specific line by

possibly better alternatives, in order to select the best line type. Such a script must be able to access specific objects (the specific line) and specific sets of objects (the set of alternative line types).

The DPL language has several methods with which the database objects and their parameters become available in the DPL script:

- The most direct method is to create an object, or a reference to an object, in the DPL command folder itself. Such an object is directly available as “object” variable in the script. The variable name is the name of the object in the database.
- The DPL command set may be used. This method is only useful when the order in which the objects are accessed is not important. The DPL command set is automatically filled when a selection of elements is right-clicked in either the single line graphic or the Data Manager and the option *Execute Script...* is selected.
- The list of external objects is mainly used when a script should be executed for specific objects or selections. The list of external objects is nothing more than a list of “aliases”. The external object list is used to select specific objects for each alias, prior to the execution of the script.

### 23.1.5.1 Object Variables and Methods

If a database object is known to the DPL command, then all its methods may be called, and all its variables are available. For example, if we want to change a load-flow command in order to force an asymmetrical load-flow calculation, we may alter the parameter `iopt_net`. This is done by using an assignment:

```
Ldf:iopt_net = 1; ! force unbalanced
```

In this example, the load-flow objects is known as the objects variable “Ldf”. The general syntax for a parameter of a database object is

```
objectname:parametername
```

In the same way, it is possible to get a value from a database object, for instance a result from the load-flow calculations. One of such a result is the loading of a line object, which is stored in the variable `c:loading`. The following example performs the unbalanced load-flow and reports the line loading. Reported value is always represented in the unit selected in *PowerFactory*. In our case returned value is in % but for example returned value for active power (`m:P:bus1`) can be represented in MW, kW, etc.

#### Example

```
int error;
double loading;
Ldf:iopt_net = 1; ! force unbalanced
error = Ldf.Execute(); ! execute load-flow
if (error) {
    exit();
} else {
    loading = Line:c:loading; ! get line loading
    printf('loading=%f', loading); ! report line loading
}
```

This examples is very primitive but it shows the basic methods for accessing database objects and their parameters.

### 23.1.6 Access to Locally Stored Objects

Locally stored objects (also called “internal objects”) can be accessed directly. They are known in the DPL script under their own name, which therefore must be a valid DPL variable name. It will not be possible to access an internal object which name is “My Load-flow\~{}1\*”, for instance.

Internal objects may also be references to objects which are stored elsewhere. The DPL command does not distinguish between internal objects and internal references to objects.

The example DPL script may now access these objects directly, as the objects “Ldf” and “Line”. In the following example, the object “Ldf”, which is a load-flow command, is used in line 01 to perform a load-flow.

```
int error;
error = Ldf.Execute();
if (error) {
    printf('Load-flow command returns an error');
    exit();
}
```

In line 01, a load-flow is calculated by calling the method `Execute()` of the load-flow command. The details of the load-flow command, such as the choice between a balanced single phase or an unbalanced three phase load-flow calculation, is made by editing the object “Ldf” in the database. Many other objects in the database have methods which can be called from a DPL script. The DPL contents are also used to include DPL scripts into other scripts and thus to create DPL “subroutines”.

### 23.1.7 Accessing the General Selection

Accessing database objects by storing them or a reference to them in the DPL command would create a problem if many objects have to be accessed, for instance if the line with the highest loading is to be found. It would be impractical to create a reference to each and every line.

A more elegant way would be to use the DPL global selection and fill it with all lines. The Data Manager offers several ways in which to fill this object **DPL Command Set** with little effort. The selection may then be used to access each line indirectly by a DPL “object” variable. In this way, a loop is created which is performing the search for the highest loading. This is shown in the following example.

#### Example

```
int error;
double maxi;
object O, Omax;
set S;

error = Ldf.Execute();      ! execute a load-flow
if (error) exit();          ! exit on error

S = SEL.AllLines();         ! get all selected lines
Omax = S.First();           ! get first line
if (Omax) {
    maxi = Omax:c:loading; ! initialise maximum
} else {
    printf('No lines found in selection');
    exit();                  ! no lines: exit
}
O = S.Next();               ! get next line
while (O) {                 ! while more lines
    if (O:c:loading>maxi) {
        maxi = O:c:loading; ! update maximum
    }
}
```

```

        Omax = 0;           ! update max loaded line
    }
    O = S.Next();
}
printf('max loading=%f', maxi); !print results
Omax.ShowFullName();

```

The object **SEL** used in line 08 is the reserved object variable which equals the *General Selection* in the DPL command dialog. The **SEL** object is available in all DPL scripts at all times and only one single “General Selection” object is valid at a time for all DPL scripts. This means that setting the **General Selection** in the one DPL command dialog, will change it for all other DPL commands too.

The method `AllLines()` in line 08 will return a set of all lines found in the general selection. This set is assigned to the variable “S”. The lines are now accessed one by one by using the set methods `First()` and `Next()` in line 09, 16 and 22.

The line with the highest loading is kept in the variable “Omax”. The name and database location of this line is written to the output window at the end of the script by calling “`ShowFullName()`”.

### 23.1.8 Accessing External Objects

The DPL contents make it possible to access external objects in the DPL script. The special general selection object (“SEL”) is used to give all DPL functions and their subroutines access to a central selection of objects. i.e. the DPL Command Set.

Although flexible, this method would create problems if more than one specific object should be accessed in the script. By creating references to those objects in the DPL command itself, the DPL command would become specific to the current calculation case. Gathering the objects in the general selection would create the problem of selecting the correct object.

To prevent the creation of calculation-specific DPL commands, it is recommended practice to reserve the DPL contents for all objects that really “belong” to the DPL script and which are thus independent on where and how the script is used.

If a DPL script must access a database object dependent on where and how the DPL script is used, an “External Object” must be added to the external object list in the DPL root command. Such an external object is a named reference to an external database object. The external object is referred to by that name. Changing the object is then a matter of selecting another object.

In Figure 23.1.3, an example of an external object is given. This external object may be referred to in the DPL script by the name “Bar1”, as is shown in the example.

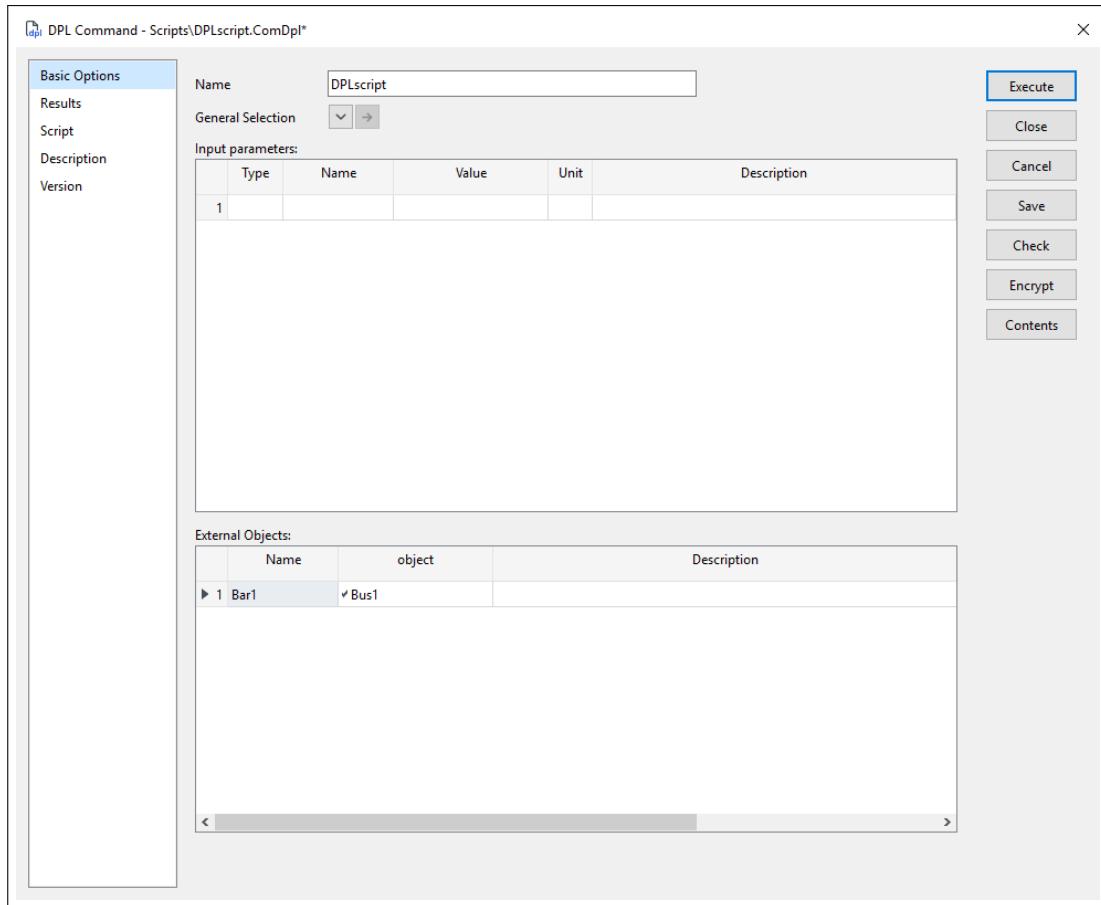


Figure 23.1.3: DPL external object table

Example:

```
sagdepth = Bar1:u;
```

### 23.1.9 Remote Scripts and DPL Command Libraries

To understand the DPL philosophy and the resulting hierarchical structure of DPL scripts, it is important to understand the following:

- A DPL command either executes its own script or the script of another, remote, DPL command. In the first case, the DPL command is called a “**root command**” and the script is called a “**local script**”. In the second case, the DPL command is called a “**referring command**” and the script is called a “**remote script**”.
- A root command may define interface variables that are accessible from outside the script and which are used to define default values.
- Each root command may define one or more **external objects**. External object are used to make a DPL command run with specific power system objects, selections, commands, etc.
- A referring command may overrule all default interface values and all selected external objects of the remote command.
- Each DPL command can be called as a subroutine by other DPL commands.

The use of remote scripts, external objects and interface variables makes it possible to create generic DPL commands, which may be used with different settings in many different projects and study cases.

The easiest way to develop a new DPL command is to create a new *ComDpl* in the currently active study case and to write the script directly in that DPL object. In such a way, a DPL “root command” is made. If this root command needs DPL subroutines, then one or more DPL command objects may be created in its contents. Each of these subroutines will normally also be written as root functions.

The newly written DPL command with its subroutines may be tested and used in the currently active study case. However, it cannot be executed when another study case is active. In order to use the DPL command in other study cases, or even in other projects, one would have to copy the DPL command and its contents. This, however, would make it impossible to alter the DPL command without having to alter all its copies.

The solution is in the use of “remote scripts”. The procedure to create and use remote scripts is described as follows.

Suppose a new DPL command has been created and tested in the currently active study case. This DPL command can now be stored in a safe place making it possible to use it in other study cases and projects.

This is done by the following steps:

- Copy the DPL command to a library folder. This will also copy the contents of the DPL command, i.e. with all its DPL subroutines and other locally stored objects.
- “Generalise” the copied DPL command by resetting all project specific external objects. Set all interface variable values to their default values. To avoid deleting a part of the DPL command, make sure that if any of the DPL (sub)commands refers to a remote script, all those remote scripts are also stored in the library folder.
- Activate another study case.
- Create a new DPL command (*ComDpl*) in the active study case.
- Set the “Remote script” reference to the copied DPL command.
- Select the required external objects.
- Optionally change the default values of the interface variables
- Press the **Check** button to check the DPL script

The **Check** or **Execute** button will copy all parts of the remote script in the library that are needed for execution. This includes all subroutines, which will also refer to remote scripts, all command objects, and all other objects. Some classes objects are copied as reference, other classes are copied completely.

The new DPL command does not contain a script, but executes the remote script. For the execution itself, this does not make a change. However, more than one DPL command may now refer to the same remote script. Changing the remote script, or any of its local objects or sub-commands, will now change the execution of all DPL commands that refer to it.

---

**Note:** PowerFactory is delivered with several ready-to-use scripts, which are located in the corresponding folder in the global library (see Section 14.2: *DlgSILENT Library*). They can be used as root commands for remote scripts or adapted as required, enhancing their functionality. The description and version page contain information about their functionalities, parameters and handling.

---

### 23.1.9.1 Subroutines and Calling Conventions

A DPL command may be included in the contents of another DPL command. In that case, the included DPL “subroutine” may be called in the script of the enclosing DPL command. In principle, this is not different from calling, for example, a load-flow command from a DPL script.

As with most other commands, the DPL command only has one method:

**int Execute();** executes the DPL script.

The difference is that each DPL subroutine has different interface parameters, which may be changed by the calling command. These interface parameters can also be set directly at calling time, by providing one or more calling arguments. These calling arguments are assigned to the interface parameters in order of appearance. The following example illustrates this.

Suppose we have a DPL sub-command “Sub1” with the interface section as depicted in Figure 23.1.4.

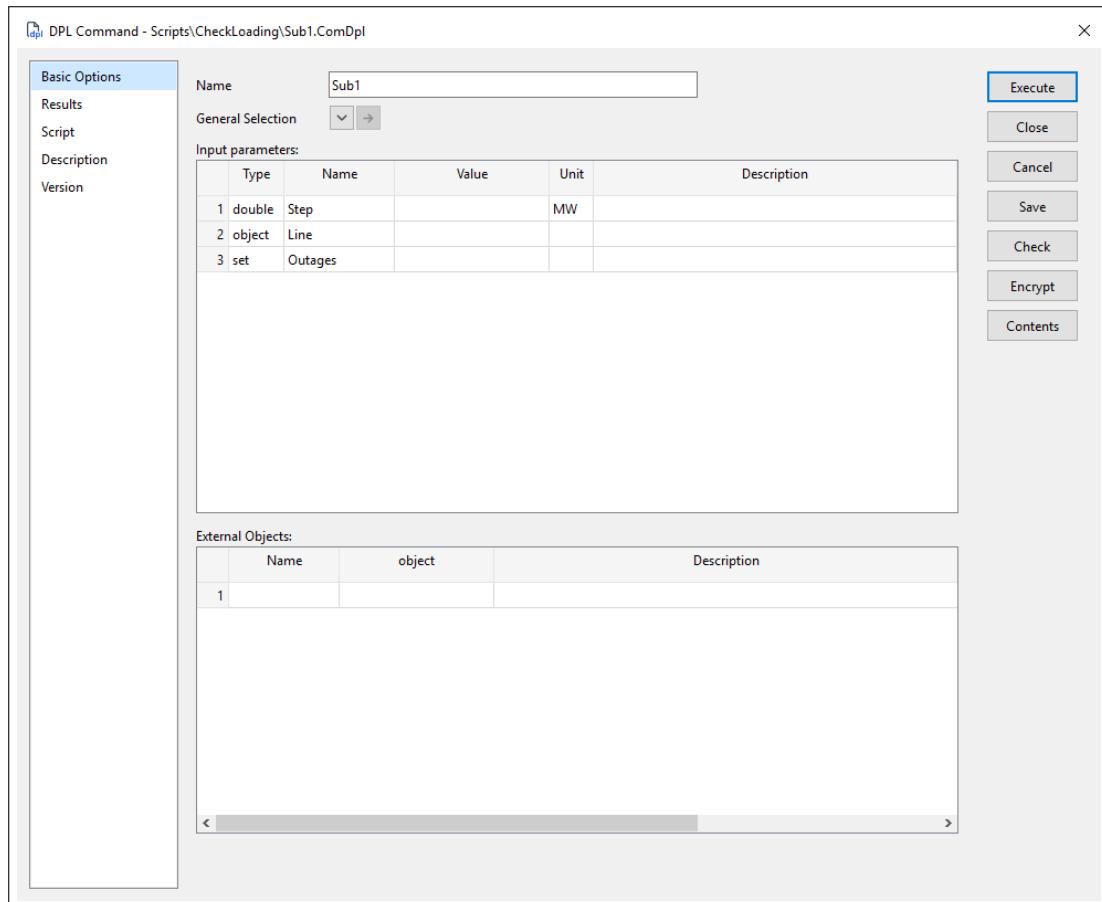


Figure 23.1.4: Interface section of subroutine

The calling command may then use, for example:

```
! set the parameters:
Sub1:step      = 5.0;
Sub1:Line      = MyLine;
Sub1:Outages   = MySelection;
! execute the subroutine:
error = Sub1.Execute();
```

However, using calling arguments, we may also write:

```
! execute the subroutine:
error = Sub1.Execute(5.0, MyLine, MySelection);
```

### 23.1.10 DPL Functions and Subroutines

The DPL syntax is very small because it mainly serves the purpose of basic operations like simple calculations, if-then-else selections, do-while loops, etc..

The strength of the DPL language is the possibility to call functions and to create subroutines. A function which can be called by a DPL command is called a “method”. Four types of methods are distinguished:

**Internal methods** These are the build-in methods of the DPL command. They can always be called.

**Set methods** These methods are available for the DPL “set” variables.

**Object methods** These methods are available for the DPL “object” variables.

**External methods** These are the methods which are available for certain external *PowerFactory* objects, such as the load-flow command, the line object, the asynchronous machine, etc.

Refer to the [DPL Reference](#) for a description of these functions including implementation examples. The DPL Reference is also accessible selecting *Help → Scripting References → DPL* from the main menu.

### 23.1.11 Data Container Objects

Vectors, maps and matrices can be used to store data related to the script. They can be created and manipulated in one of these five types:

**IntVec:** a vector of decimal numbers (of type double)

**IntVecobj:** a vector of object

**IntDplvec:** a vector of objects, sets, integers, strings or decimal numbers

**IntMat:** a matrix of decimal numbers (of type double)

**IntDplmap:** a vector of objects, sets, integers, strings or decimal numbers mapped to key values

It is good practice to store the data container objects to be used by the script within the DPL command.

#### 23.1.11.1 Vectors

There are three types of vectors that can be used when scripting: **IntVec**, **IntVecobj** and **IntDplvec**.

*IntVec* is used to describe arrays of decimal numbers, *IntVecobj* offers the functionality of storing objects and *IntDplvec* is a more general purpose vector that can describe an array / container of objects, sets, strings or numbers.

The other key differences between the two vector types are as follows:

- *IntVec* and *IntVecobj* objects can be edited manually in the Data Manager, whereas *IntDplvec* objects can only be manipulated within a DPL script.
- The first item in *IntVec* and *IntVecobj* objects is at index “1”, while in *IntDplvec* objects, it is at index “0”.
- *IntDplvec* objects have no resizing functionality, and the vector size grows (or shrinks) organically with item insertions and removals.
- *IntDplvec* objects have a reverse lookup feature, i.e. one can find the index number of a vector item

- *IntDplvec* objects are sortable.
- 

**Note:** The values in an *IntDplvec* must have the same type. The first value inserted into the *IntDplvec* determines the type. For example, if an object is inserted, then all other values in the vector must also be of type object.

---

The functions of the *IntVec*, *IntVecobj* and *IntDplvec* objects are available in the [DPL Reference](#).

### 23.1.11.2 Maps

DPL map objects (*IntDplmap*) are so-called *associative arrays*, which are essentially arrays of key - value pairs. For example, consider the DPL map shown below:

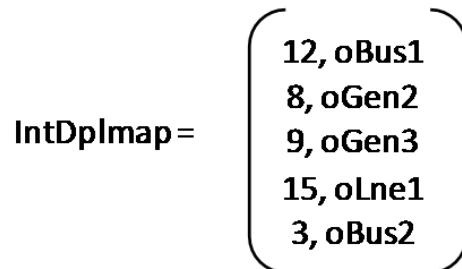


Figure 23.1.5: Example for *IntDplmap*

In the example above, there are five key-value pairs, e.g. “12, oBus1”, “8, oGen2”, etc. Thus for the key-value pair “9, oGen3”, the key “9” corresponds to the value “oGen3”. We can say that we’ve mapped the key “9” to the object “oGen3”.

In DPL map objects, the keys can be represented by numbers, strings, objects or sets. In the example above, all of the keys are shown as integer numbers, while their corresponding paired values are objects. Note that the keys and values must have homogenous types. The first key-value pair inserted into the map determines the types for the keys and values. For example, if you insert an object key and a set value, then all other keys-value pairs must be of type object and set.

The functions and usage of DPL map objects are available in the [DPL Reference](#).

### 23.1.11.3 Matrices

Matrix objects (*IntMat*) are two dimensional matrices of decimal numbers.

The functions and usage of matrix objects are available in the [DPL Reference](#).

## 23.2 Tabular Reports

Tabular reports are a powerful tool for generating your own personalised data views and showing all the information you want to extract from the *PowerFactory* model in one table.

Before starting to create your own tabular reports, you should be familiar with the DPL programming language (Section [23.1](#)).

The command *ComTablereport* extends the DPL programming language with a simple framework to create tables.

The command *ComTablereport* is not available in Python. In Python the use of a external GUI framework like TKInter or Qt is recommended. It is also possible to create a Tabular Report with DPL and use Python subroutines to calculate the contents.

Tabular Reports provide the following features:

- Display user-defined data in a table:
  - Allows the user to sort the table by each column.
  - Allows the user to use a data filter for each column.
  - Provides a callback function to add cell data editing features.
  - Provides a callback function to add additional entries to cells context menu.
  - Allows copy and paste of the table data.
- Allows the addition of global programmable selection filters and input data.
- Allows the addition of extra buttons to trigger user defined actions.
- Allows a user-defined data plot to be displayed instead of the table.
- Allows the user to export the table content to a HTML or Excel file.

### 23.2.1 Basic Structure of a Tabular Report

A Tabular Report always consists of a *ComTablereport* object and one or more callback *ComDpl* objects as children. The *ComTablereport* only supports the following predefined set of *ComDpl* objects:

- **Init optional** The Init script is called only once, when the report is displayed the first time.
- **Create mandatory** The Create script is called every time, when the report is displayed or rebuilt.
- **Edit optional** The Edit script is called after the user changes a cell.
- **Action optional** The Action script is called after the user selects a user-defined entry from the context menu.
- **ButtonPressed optional** The ButtonPressed script is called after the user presses a button.

Only the *Create* object is mandatory. The concept is quite simple. The whole report is defined with the *Create* function. If something changes, the whole report has to be rebuilt from scratch with the *Create* function.

### 23.2.2 The Tabular Report Command

The Tabular Report element *ComTablereport* itself contains only a few attributes:

- **Name loc\_name** The name of the ComTablereport object.
- **Use Selection iSelection** A flag indicating whether the report use a Selection as input.
- **Class Filter sFilter** A class filter, which is applied to the input selection to extract only the relevant elements for the report.
- **Description attributes** Additional attributes to provide a short and a detailed description.
- **Version attributes** Additional attributes to provide version, author and copyright information.

The *ComTablereport* object provides a wide variety of functions, to define the tabular report, as shown in Figure 23.2.1 below. Please refer to the function descriptions in the [DPL Reference](#).

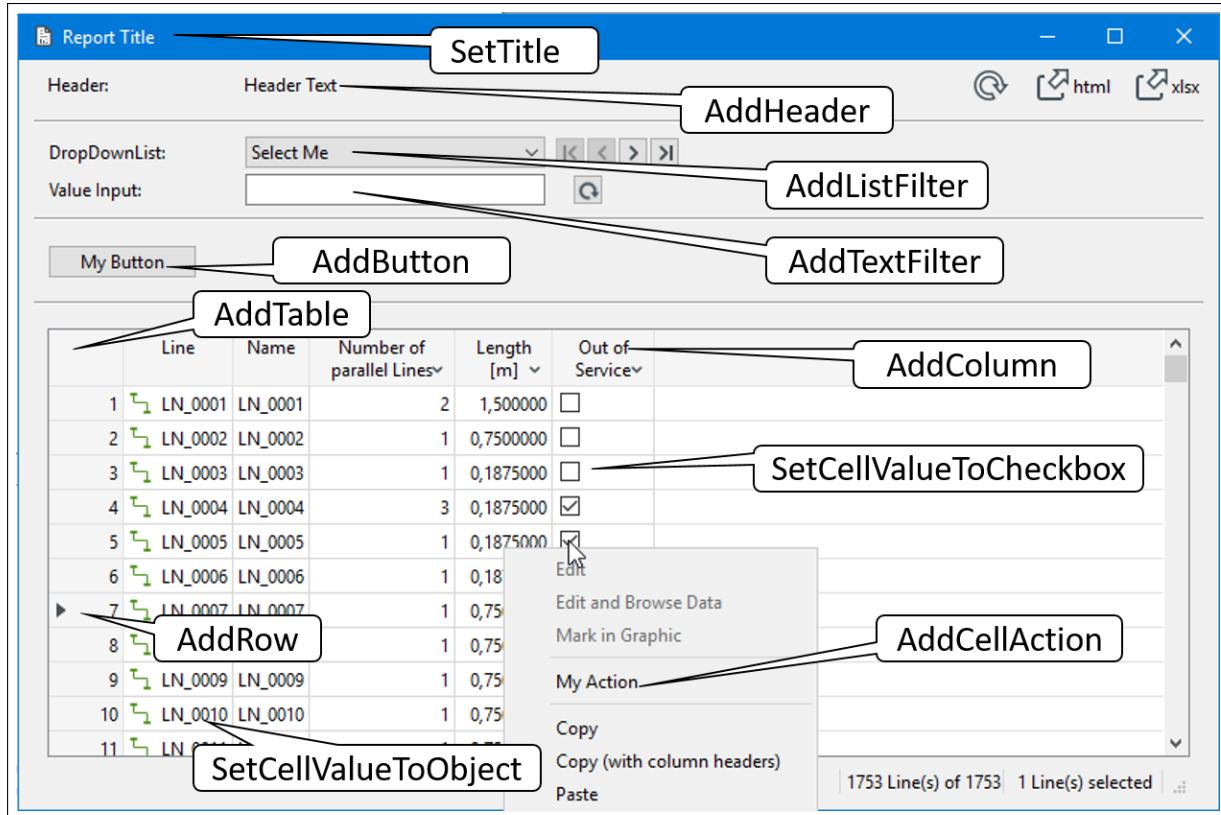


Figure 23.2.1: ComTablereport functions

### 23.2.2.1 Creating a new Tabular Report Command

A tabular report command element *ComTablereport* can be created by pressing the *New Object* icon , and using the filter to select *Tabular Report* (*ComTablereport*) from the list.

**Note:** By default, a tabular report that has been generated will be closed when the active desktop is closed. The *ComTablereport* offers options in the *Scope* panel to modify this behaviour, either by executing the report again when the desktop is opened again or by only closing it when the active project is deactivated.

After creating the *ComTablereport* object, a new *ComDpl* object with the name *Create* has to be added as a child of the *ComTablereport* object. See Section 23.2.3 for an example. Other *ComDpl* functions (see Section 23.2.6) may be added optionally.

### 23.2.2.2 Executing a Tabular Report Command

To execute and show a Tabular Report:

- Right-click at the *ComTablereport* object and choose *Execute* from the context menu.
- or Open or Edit the *ComTablereport* object and press the *Execute* button.

If the Tabular Report uses a Selection:

- Select some elements in a single line diagram or in the data manager.
- Then right-click and choose from the context menu *Show Tabular Report*
- Then select a *ComTablereport* object from the list. (This step is omitted if there is only one Report with a Selection.)

### 23.2.3 A minimal Tabular Report

A minimal tabular report is built from a *ComTablereport* with a DPL script *ComDpl Create* as child. Section 23.2.2.1 describes how to create these. With the following code in the *Create* script, the report will show a table with 2 columns and a row with the words *Hello World*.

**Example 1 “Hello world report” *Create* script:**

```
object oReport;
oReport = this.GetParent();
oReport.AddTable('Table ID');
oReport.AddColumn('Table ID', 'Column 1 ID', 'First Column');
oReport.AddColumn('Table ID', 'Column 2 ID', 'Second Column');
oReport.addRow('Table ID', 'Row 1 ID');
oReport.SetCellValueToString('Table ID', 'Column 1 ID', 'Row 1 ID', 'Hello');
oReport.SetCellValueToString('Table ID', 'Column 2 ID', 'Row 1 ID', 'World');
```

As the first step, the script has to access the *ComTablereport* object. This command object is always the parent of the *Create* script. So a simple *this.GetParent()* will work. As the second step, a table with an identifier is defined with the function *AddTable('Table ID')*. Use this table identifier in all subsequent functions to define the table. In the next step the table columns will be defined. With each call to the *AddColumn* function a new column is defined with its own column identifier and a column header text. In a similar way the rows of the table will be defined. The function *SetCellValueToString* fills the content of a cell with a string value. To refer a specific cell, the defined table, column and row identifiers must be used.

To test the report, you have to execute the tabular report (see Section 23.2.2.2), but not the *Create* script.

### 23.2.4 Handling different kinds of data

In the first example, the function *SetCellValueToString* was used to fill the table cells with content. To view different kinds of data, there are several similar functions. See the [DPL Reference](#) for a description of all these functions. The following example demonstrates the use of some of these functions. This example requires an active project with at least some lines (*ElmLne* objects) in the network model. You could use one of the *PowerFactory* example projects to test the report.

**Example 2 “Lines report” *Create* script:**

```
set aLines;
object oReport,
oLine;
string sRowId;

oReport = this.GetParent();
oReport.AddTable('Table ID');
oReport.AddColumn('Table ID', 'line column', 'Line');
oReport.AddColumn('Table ID', 'name column', 'Name');
oReport.AddColumn('Table ID', 'parallel column', 'Number of\nparallel Lines');
```

```
oReport.AddColumn('Table ID', 'length column', 'Length\n[m]');
oReport.AddColumn('Table ID', 'out column', 'Out of\nService');

aLines = GetCalcRelevantObjects('*.*.ElmLne');
for(oLine = aLines.First(); oLine; oLine = aLines.Next()) {
    sRowId = oLine.GetFullName();
    oReport.AddRow('Table ID', sRowId);
    oReport.SetCellValueToObject('Table ID', 'line column', sRowId, oLine);
    oReport.SetCellValueToString('Table ID', 'name column', sRowId, oLine:loc_name);
    oReport.SetCellValueToInt('Table ID', 'parallel column', sRowId, oLine:nlnum);
    oReport.SetCellValueToDouble('Table ID', 'length column', sRowId, oLine:dline);
    oReport.SetCellValueToCheckbox('Table ID', 'out column', sRowId, oLine:outserv);
}
```

Some of these functions not only show plain data, but provide additional features. If a *PowerFactory* object is shown with the function *SetCellValueToObject*, a double click in the cell opens the edit dialog of this object. In addition, there is a new entry in the context menu of the cell, which will mark the object in a graphic. The *SetCellValueToCheckbox* expects a boolean value, which is displayed as a checkbox.

## 23.2.5 Advanced Features

### 23.2.5.1 Programmable Filters and Input Values

The *ComTableReport* function *AddListFilter* adds a drop-down list to the report. The drop-down list has an identifier and a visible name. The function *AddListFilterEntries* can be used to add entries to the list. To get the selected entry, the *Create* script has to define a *string* input parameter with the name *s<identifier>*. When the *Create* script is called the first time to generate the report, this parameter contains the default value, or an empty string if no default value is set. Every time the user selects a new entry from the drop-down-list or another event refreshes the report, the *Create* script is called again with the selected entry in the input parameter *s<identifier>* to recreate the report. In the example below, the drop-down list is given the identifier *UFilt*. So the *Create* script should contain a *string* input parameter *sUFilt*. This parameter defines the currently selected entry. The function *SetListFilterSelection* can be used to set the selected entry in the drop-down list.

**Example 3 “Drop-down list report” *Create* script:**

```
object oReport;
oReport = this.GetParent();
oReport.AddListFilter('UFilt', 'Usage');
oReport.AddListFilterEntries('UFilt', 'Busbar');
oReport.AddListFilterEntries('UFilt', 'Junction');
oReport.AddListFilterEntries('UFilt', 'Internal');
oReport.SetListFilterSelection('UFilt', sUFilt);
```

If all the list information is available at once, it is possible to shorten the above code and create the whole drop-down list with one call to *AddListFilter*. To distinguish the list entries, they have to be separated by “\n”.

**Example 3 “Drop-down list report” *Create* script change:**

```
oReport.AddListFilter('UFilt', 'Usage:', 'Busbar\nJunction\nInternal', aDummy, sUFilt);
```

With this code it is possible to create different views of the report, depending on the input parameter *sUFilt*.

The *ComTableReport* function *AddTextFilter* creates an additional text input field in the report. In contrast to the drop-down list, the report will not trigger an automatic rebuild if the user changes the text. Instead,

the user has to press the **Refresh** button at the end of the text field. To obtain the user input, the *Create* script has to define the input parameter `s<identifier>`

**Example 4 “Input value report”** Create script:

```
object oReport,  
oReport = this.GetParent();  
oReport.AddTextFilter('TFilt', 'Text input', '', sTFilt);
```

The *Create* script can also modify the user input. For example, if the user must be able to input numbers, the script code can just convert the input in a number and discard any invalid input data.

**Example 4 “Input value report”** Create script:

```
object oReport,  
string sCurrentValue,  
int iNumber, iRes;  
  
oReport = this.GetParent();  
iNumber = 0;  
iRes = sscanf(sNFilt, '%d', iNumber);  
sCurrentValue = sprintf('%d', iNumber);  
oReport.AddTextFilter('Nfilt', 'Nominal Voltage above', 'KV', sCurrentValue);
```

The above code always resets the input to 0 if the user enters invalid data. Sometimes it may be useful to remember the last valid value to restore. In this case a hidden filter can be used with the function *AddInvisibleFilter*. The hidden filter does nothing but preserve the data between 2 consecutive calls of the *Create* function. Again the *Create* script has to define the *string* input parameter *s<identifier>* to get the value back in the call of the function.

**Example 4 “Input value report”** Create script:

```
object oReport,  
string sCurrentValue,  
int iNumber, iRes;  
  
oReport = this.GetParent();  
iNumber = 0;  
iRes = sscanf(sOldValue, '%d', iNumber);  
iRes = sscanf(sNFilt, '%d', iNumber);  
sCurrentValue = sprintf('%d', iNumber);  
oReport.AddInvisibleFilter('OldValue', sCurrentValue, NULL);  
oReport.AddTextFilter('Nfilt', 'Nominal Voltage above', 'KV', sCurrentValue);
```

The third parameter of the function *AddInvisibleFilter* takes a *PowerFactory* object as input. To get this object back in the next call, the *Create* script has to define an external object *o<identifier>* as input.

### 23.2.5.2 Edit Cells

The *ComTableReport* function *SetCellEdit* makes an individual cell editable. To identify the cell the defined table, column and row identifiers must be used. In addition, the callback *ComDpl* script *Edit* must be provided as a child of the *ComTableReport*. This callback report takes five input parameters:

- **sColumnId** *string* The column identifier of the changed cell.
- **sRowId** *string* The row identifier of the changed cell.
- **sNewValue** *string* The new value, if the cell contains a string.
- **iNewValue** *int* The new value, if the cell contains an int.
- **dNewValue** *double* The new value, if the cell contains a double.

It is the responsibility of the programmer to change the underlying value with the *Edit* callback script. With the help of this script the user input can also be changed or rejected, if it is invalid. Every time the user changes a cell, the *Edit* callback script is called. Following the execution of the *Edit* script, the Tabular Report is rebuilt with a call of the *Create* script. To assist the edit process, a set of cell-specific objects can be passed with the call of the function *SetCellEdit*. These objects will be available in the *Edit* callback script as external input objects:

- ***oEditObject1*** The first object passed from the *Create* script with *SetCellEdit*.
- ***oEditObject2*** The second object passed from the *Create* script with *SetCellEdit*.
- ***oEditObject<X>*** The Xth object passed from the *Create* script with *SetCellEdit*.

As an example of a tabular report with editable cells, you can extend the script from Chapter 23.2.4. Add a new *set* variable to the **Create** script at line 1.

#### Example 2 “Lines report” *Create* script extension:

```
set aEditSet;
```

Add the following code to the *Create* script before line 23.

#### Example 2 “Lines report” *Create* script extension:

```
aEditSet.Clear();
aEditSet.Add(oLine);
oReport.SetCellEdit('Table ID', 'name column', sRowId, aEditSet);
oReport.SetCellEdit('Table ID', 'parallel column', sRowId, aEditSet);
oReport.SetCellEdit('Table ID', 'length column', sRowId, aEditSet);
oReport.SetCellEdit('Table ID', 'out column', sRowId, aEditSet);
```

Add a new *ComDpl* object *Edit* to the tabular report. Define the input parameter (see above) and an external object *oEditObject1*. Fill the script content of the *Edit* script with the following code:

#### Example 2 “Lines report” *Edit* script:

```
object oLine;
int iColumnCmp;
oLine = oEditObject1;
iColumnCmp = strcmp(sColumnId, 'name column');
if (iColumnCmp = 0){
    oLine:loc_name = sNewValue;
}
iColumnCmp = strcmp(sColumnId, 'parallel column');
if (iColumnCmp = 0){
    oLine:nlnum = iNewValue;
}
iColumnCmp = strcmp(sColumnId, 'length column');
if (iColumnCmp = 0){
    oLine:dline = dNewValue;
}
iColumnCmp = strcmp(sColumnId, 'out column');
if (iColumnCmp = 0){
    oLine:outserv = iNewValue;
}
```

With this script all visible attributes of the line could be changed. **Note:** this example modifies the attributes of the lines of your active project!

### 23.2.5.3 Additional Context Menu Entries

The *ComTablereport* function *AddCellAction* adds a new entry to the context menu of a cell. To identify the cell the defined table, column and row identifiers must be used. The fourth parameter is the *Action Identifier* and the fifth parameter defines the text of the context menu entry. In addition, the callback *ComDpl* script *Action* must be provided as a child of the *ComTablereport*. This callback script takes two input parameters:

- **sActionId** *string* The *Action Identifier*, which was defined with the call of *AddCellAction*.
- **aObjects** *set* The set of objects, which were passed through with the call of *AddCellAction*.

The callback function *Action* is always called if the user selects an additional entry from the cell context menu. The function *AddCellAction* provides an optional parameter *refresh* to define whether the report shall be rebuilt after the execution of the *Action* script or not. The default is set to rebuild the report with a call of the *Create* script.

### 23.2.5.4 Additional Buttons

The *ComTablereport* function *AddButton* adds a new button with an identifier and a label to the top of the tabular report. The callback *ComDpl* script *ButtonPressed* must be provided as a child of the *ComTablereport*. This callback script takes one input parameter.

- **sButtonId** *string* The *Button Identifier* which was defined with the call of *AddButton*.

The callback function *ButtonPressed* is always called if the user presses a button in the tabular report. The function *AddButton* provides an optional parameter *refresh*, which defines whether the report shall be rebuilt after the execution of the *ButtonPressed* script or not. If the callback script *ButtonPressed* changes the content of table, the parameter *refresh* will be always set to 1.

### 23.2.5.5 Using a Selection

If the tabular report must be able to show user defined pre-selected elements, it must have set the **Use Selection** *iSelection* attribute of the *ComTableReport*.

See Section [23.2.2](#) for a description of the *ComTableReport* attributes.

In addition, the report must be stored in the report folder *Library\Reports\Tables* of the project. The exact name of the report folder depends on the *PowerFactory* language settings at the time of project creation.

If both conditions are fulfilled, the report can be executed with a user selection (see Section [23.2.2.2](#)).

Inside the *Create* or *Init* callback scripts the *ComTablereport* function *GetSelection* provides access to this selection.

If the *ComTablereport* attribute **Class Filter** *sFilter* is set, the function *GetSelectedElements* could be used instead. This function returns an already filtered set of selected objects.

## 23.2.6 Tabular Report Callback Script Reference

### 23.2.6.1 Init

- optional
- called only when the report is opened the first time

- no input parameter

#### 23.2.6.2 Create

- mandatory
- called if the report is created or rebuilt
- input parameter passed through from *AddInvisibleFilter*, *AddListFilter*, *AddMultiListFilter*, *AddTabularFilter* or *AddTextFilter*:
  - **s<FilterId>** *string* The selected value of a filter.
- external input objects passed through from *AddInvisibleFilter*, *AddListFilter* or *AddMultiListFilter*:
  - **o<FilterId>** The selected object of a filter.

#### 23.2.6.3 Edit

- optional
- called if:
  - a cell defined with *SetCellEdit* is changed by the user *or*
  - a filter defined with *AddInvisibleFilter*, *AddListFilter*, *AddMultiListFilter*, *AddTabularFilter* or *AddTextFilter* is changed by the user
- input parameters:
  - **sColumnId** *string* The column identifier of the changed cell or the filter identifier of the changed filter.
  - **sRowId** *string* The row identifier of the changed cell.
  - **sNewValue** *string* The new value, if the cell contains a string, or the new value of the changed filter.
  - **iNewValue** *int* The new value, if the cells contains an integer.
  - **dNewValue** *double* The new value, if the cells contains a double.
- external input objects:
  - **oEditObject1** The first object passed from the Create script with *SetCellEdit* or the selected object of a changed filter.
  - **oEditObject2** The second object passed from the Create script with *SetCellEdit*.
  - **oEditObject<X>** The Xth object passed from the Create script with *SetCellEdit*.

#### 23.2.6.4 Action

- optional
- called if the user selects an additional context menu entry defined with *AddCellAction*.
- input parameters:
  - **sActionId** *string* The *Action Identifier* which was defined with the call of *AddCellAction*.
  - **aObjects** *set* The set of objects, which were passed through with the call of *AddCellAction*.

### 23.2.6.5 ButtonPressed

- optional
- called if the user presses a button defined with *AddButton*.
- input parameter:
  - **sButtonId** string The *Button Identifier* which was defined with the call of *AddButton*.

## 23.3 Python

This section describes the integration of the Python scripting language in *PowerFactory* and explains the procedure for developing Python scripts. The Python scripting language can be used in *PowerFactory* for:

- Automation of tasks
- Creation of user-defined calculation commands
- Integration of *PowerFactory* into other applications

Some of Python's notable features include:

- General-purpose, high-level programming language
- Clear, readable syntax
- Non-proprietary, under liberal open source licence
- Widely used
- Extensive standard libraries and third-party modules
  - Interfaces to external databases and Microsoft Office-like applications
  - Web services, etc.

The integration of Python into *PowerFactory* makes the above-mentioned features accessible to users of *PowerFactory*.

To execute a Python script the following steps have to be considered:

1. Python interpreter has to be installed. (SubSection 23.3.1)
2. Python file .py that contains code of the script has to be created by external editor. After being created .py file can be link to the *ComPython* object inside of *PowerFactory*. (SubSection 23.3.3)
3. In each .py file *PowerFactoryPython* module 'powerfactory.pyd' has to be imported. (SubSection 23.3.2)
4. To run the script the User has to execute the (*ComPython*) object. (SubSection 23.3.3.2)

### 23.3.1 Installation of a Python Interpreter

When *PowerFactory* is installed, the installation does not include a Python interpreter and therefore this must be installed separately. The recommended Python versions are available on <https://www.python.org/downloads/>. All supported Python versions can be checked inside of the *PowerFactory* Configuration dialog (see Figure 23.3.1) here is also where a preferred Python version can be selected.

*PowerFactory* requires a Python interpreter for 64-bit. To avoid issues with third-party software, the Python interpreter should be installed with default settings (for all users), into the directory proposed by the installer.

Depending on the functions to be performed by a particular Python script, it may be necessary to install a corresponding Python add-on/package. For example, Microsoft Excel can be used by Python scripts if the “Python for Windows Extensions” *PyWin32* (<http://sourceforge.net/projects/pywin32/>) package is installed, which includes Win32 API, COM support and Pythonwin extensions.

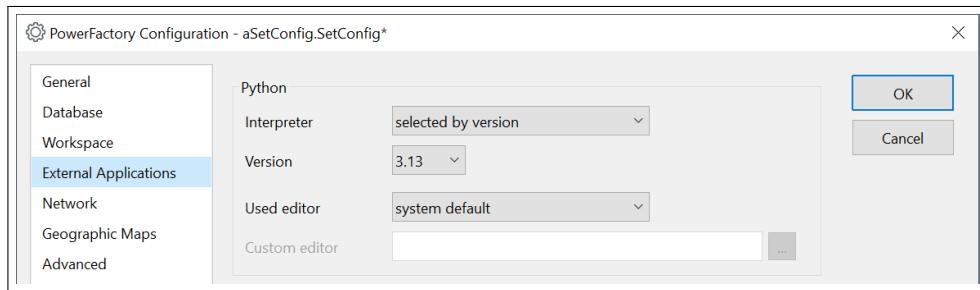


Figure 23.3.1: Selection of a preferred Python editor program

### 23.3.2 The Python *PowerFactory* Module

The functionality of *PowerFactory* is provided in Python through a dynamic Python module (“powerfactory.pyd”) which interfaces with the *PowerFactory* API (Application Programming Interface). This provides Python scripts with access to a comprehensive range of data in *PowerFactory*:

- All objects
- All attributes (element data, type data, results)
- All commands (load flow calculation, etc.)
- Most special built-in functions (DPL functions)

A Python script which imports this dynamic module can be executed from within *PowerFactory* through the new Python command *ComPython* (see Section 23.3.3), or externally (*PowerFactory* is started by the Python module in non-interactive mode) (see Section 23.3.4).

#### 23.3.2.1 Python *PowerFactory* Module Usage

To allow access to the Python *PowerFactory* module it must be imported using the following Python command:

```
import powerfactory
```

To gain access to the *PowerFactory* environment the following command must be added:

```
app = powerfactory.GetApplicationExt()
```

A Python object of class `powerfactory.Application` is called an application object. Using the application object from the command above (“`app`”), it is possible to access global *PowerFactory* functionality. Several examples are shown below:

```
user = app.GetCurrentUser()
project = app.GetActiveProject()
script = app.GetCurrentScript()
objects = app.GetCalcRelevantObjects()
lines = app.GetCalcRelevantObjects("*.ElmLine")
sel = app.GetDiagramSelection()
sel = app.GetBrowserSelection()
project = app.CreateProject("MyProject", "MyGrid")
```

```
ldf = app.GetFromStudyCase("ComLdf")
```

The listed methods return a data object (Python object of class `powerfactory.DataObject`) or a Python list of data objects. It is possible to access all parameters and methods associated with a data object. Unlike DPL syntax, Python syntax requires use of the dot (.) operator instead of the colon (:) in order to access element parameters of objects (i.e. name, out of service flag, etc.).

#### Examples:

```
project = app.GetActiveProject()
projectName = project.loc_name
project.Deactivate()
```

All other object parameters (calculated, type, measured, ...) are to be called by `GetAttribute()` method and using the colon (:), as is done in DPL.

```
lines = app.GetCalcRelevantObjects("*.ElmLne")
line = lines[0]
currLoading = line.GetAttribute("c:loading")
```

For printing to the *PowerFactory* output window, the following application object (e.g. "app" object) methods are provided:

```
app.PrintPlain("Hello world!")
app.PrintInfo("An info!")
app.PrintWarn("A warning!")
app.PrintError("An error!")
```

Printing the string representation of data objects to the *PowerFactory* output window makes them selectable (i.e. creates a hyperlink string in the output window):

```
project = app.GetActiveProject()
app.PrintPlain("Active Project: " + str(project))
```

A list of all parameters and methods associated with an object can be obtained using the `dir()` function as shown below:

```
project = app.GetActiveProject()
app.PrintPlain(dir(project))
```

#### 23.3.2.2 Python *PowerFactory* Module Reference

A Python Module Reference document is available in the Help menu containing a list of offered functions.  
[Python reference](#)

#### 23.3.3 The Python Command (*ComPython*)

In contrast to DPL, the Python command only links to a Python script file. It stores only the file path of the script and not the file itself.

The script may be executed by clicking on the **Execute** button of the corresponding dialog. Editing the script file is possible by clicking the **Open in External Editor** button.

The preferred editor may be chosen in the *External Applications* page of the *PowerFactory Configuration* dialog by selecting the *Tools → Configuration...* menu item from the main menu (see Section 5.2.4).

Python scripts may be created in any text editor as long as the script file is saved using the UTF-8 character encoding format. The Python version can be selected in the *PowerFactory Configuration* dialog.

On the *Basic Options* page of the Python command (*ComPython*), the user can define *Input parameters* and *External objects*. The *Input parameter* table in the dialog is used as in DPL to define variables that can be accessed from outside of the Python script. *Input parameters* may be the following data types: double, int and string. All other fields (Name, Value, Unit, Description) are user definable.

The *External object* table allows the direct configuration of objects under investigation. An external object is an object external to the Python command that the user wants to access in the script. By defining the *External object* here, the user avoids accessing it via Python methods inside the script (thereby allowing the script to execute faster).

**Important:** To access an external object or input parameter in Python, the user has first to get the script and can then access the parameter through `ScriptObj.ExternObjName`:

```
script=app.GetCurrentScript()    #to call the current script
extObj=script.NameOfExternObj #to call the external object
inpPar=script.NameOfInpPram   #to call input parameter
```

*Result parameter* section is to be found on the *Results* page. The *Result parameters* represents results from the script and they are stored inside the specified results object. The *Script* page contains three sections: *Remote script*, *Script file* and *Interface version*. *Remote script* offers a selection of a remote script, which is used instead of the code defined in the *Script file* section. A remote script can be advantageous in cases where the user has multiple Study Cases using the same script code, but different input parameters. If the user wants to use a remote script, then modifying the master script will affect all the Python scripts that refer to it. If the user had locally-defined scripts, then they would need to change the code in each of them.

*Python Script*: The source of the Python script can be selected. Two options are available:

- *External*: An external \*.py Python script file has to be selected under *Script file*.
- *Embedded*: The script is embedded in the ComPython-object and can directly be developed using the internal editor (see Section 23.4). Please note that only single file Python scripts are supported.

*Script file* will be available if *External* (see above) is selected. It is a field that contains the path of the Python script file (\*.py).

*Embedded Code*: Available if option *Embedded* (see above) is selected. The Python script is embedded in the ComPython-object and can directly be developed in the editor.

*Interface Version* refers to the returned value of some Python methods. For example if selecting

Version 1-“old interface version”. Data object methods with arguments such as the method `GetPage()` from the class `SetDesktop` return list containing [[returned value], argument of the following entries] :

```
list (DataObject, ...) SetDesktop::GetPage (str, [int])
```

Version 2- “new interface version”. Methods like the method `GetPage()` returns just the result, without input parameters.

```
DataObject SetDesktop::GetPage (str, [int])
```

The Python command may also contain objects or references to other objects available in the *PowerFactory* database. These can be accessed by clicking on the **Contents** button. New objects are defined by first clicking the *New Object*  icon in the toolbar of the Python script contents dialog and then selecting the required object from the *New Object* window. References to other objects are created by

defining a reference object “IntRef”. **Note:** Python supports different access to this objects than DPL. See example

```
script=app.GetCurrentScript()
ContainedObject=script.GetContents('Results.ElmRes')
```

### 23.3.3.1 Creating a New Python Command

To create a new Python command click on the *New Object* ( icon) in the toolbar of the Data Manager and use the filter to select Python Script (*ComPython*) from the list. Then press **OK** and a new Python command will be created. The Python command dialog is then displayed. The name of the script, its input parameters and the file path to the script, etc, can now be specified. The Python command dialog is also opened by double-clicking a Python script; by selecting *Edit* from the context menu or by selecting the script from the list after clicking on the main toolbar icon *Execute Script* ().

### 23.3.3.2 Executing a Python Command

A Python command may be executed by clicking the **Execute** button in the dialog.

Alternative methods for executing a Python script include:

- From the Data Manager:
  - Right-click on the Python command and select *Execute* from the context menu.
  - Right-click in a blank area and select *Execute Script* from the context menu. A list of existing DPL and Python scripts contained in the global and local libraries will pop up. Select the required Python script and click **OK**.
- From the single line diagram:
  - Select one or more elements in the single line diagram. Right-click the marked elements and select *Execute Script* from the context menu. A list of existing DPL and Python scripts contained in the global and local libraries will pop up. Select the required Python script and click **OK**.
  - A button may be created in the single line diagram to automate the execution of a specific Python script.
- From the main toolbar:
  - Click the icon *Execute Script* . A list of existing DPL and Python scripts from the global and local libraries will appear. Select the specific Python script and click **OK**.

---

**Note:** The custom global library will only be shown if the library is set as *Used Library* in the User Settings (see chapter User Settings, Section 7.2).

---

### 23.3.4 Running *PowerFactory* in Non-interactive Mode

*PowerFactory* may be run externally by Python. In order to do this, the script must additionally add the file path of the dynamic module (“powerfactory.pyd”) to the system path. Example:

```
# Add powerfactory.pyd path to python path.
import sys
sys.path.append("C:\\Program Files\\DIgSILENT\\PowerFactory 2025 SP1\\Python\\3.13")

#import PowerFactory module
```

```

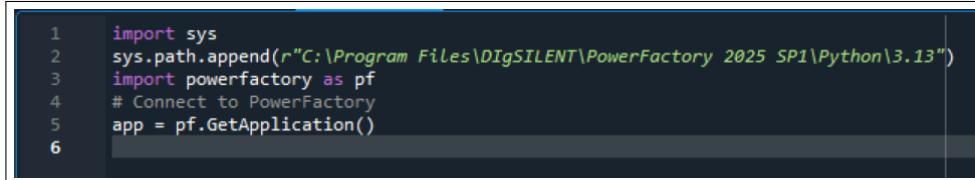
import powerfactory

#start PowerFactory in non-interactive mode
app = powerfactory.GetApplicationExt()

#run Python code below
.....

```

The *PowerFactory* environment can be accessed directly from the Python console as shown in Figure 23.3.2



```

1 import sys
2 sys.path.append(r"C:\Program Files\DIgSILENT\PowerFactory 2025 SP1\Python\3.13")
3 import powerfactory as pf
4 # Connect to PowerFactory
5 app = pf.GetApplication()
6

```

Figure 23.3.2: Python console

**Note:** If an error message appears when importing the *powerfactory* module stating “DLL load failed: the specified module could not be found”, this often means that the corresponding Microsoft Visual C++ Redistributable are not installed on the computer.

### 23.3.5 Performance of Python Scripts

The execution time of a Python script can vary strongly depending on the used environment set up. There are dedicated environment functions [Python Reference](#) to improve the performance of a script. One example for a possible performance improvement is disabling the *User Break*-button. This button has a huge impact on the execution time of some scripts, because every few simulation steps Python has to check, whether the button has been pressed. The *SetUserBreakEnabled()*-function allows to disable and enable the break button freely during the execution of a script.

### 23.3.6 Debugging Python Scripts

As with any other Python script, it is possible to remotely debug scripts written for *PowerFactory* by using specialised applications.

#### 23.3.6.1 Prerequisites

The recommended IDE for debugging is Eclipse ([www.eclipse.org](http://www.eclipse.org)) with the Python add-on PyDev ([www.pydev.org](http://www.pydev.org)).

1. Install Eclipse Standard from [www.eclipse.org/downloads/](http://www.eclipse.org/downloads/)
2. Open Eclipse
3. Click “Install New Software …” in the “Help” menu
4. Add the repository <http://pydev.org/updates> and install PyDev

### 23.3.6.2 Debugging a Python script for *PowerFactory*

The following is a short description of remote debugging with PyDev. For more information please consult the remote debugger manual of PyDev ([http://pydev.org/manual\\_adv\\_remote\\_debugger.html](http://pydev.org/manual_adv_remote_debugger.html)).

1. Start Eclipse
2. Open Debug perspective
3. Start the remote debugger server by clicking “Start Debug Server” in the “Pydev” menu
4. Start *PowerFactory*
5. Prepare the Python script for debugging:
  - Add “pydevd.py” path to sys.path
  - Import PyDev debugger module “pydevd”
  - Start debugging calling pydevd.settrace()

Example:

```
#prepare debug
import sys
sys.path.append ("C:\\Program Files\\eclipse\\plugins\\
                 org.python.pydev_2.8.2.2013090511\\pysrc")
import pydevd

#start debug
pydevd.settrace()
```

6. Execute the Python script
7. Change to Eclipse and wait for the remote debugger server

It is not possible to stop and restart the remote debugger server while running *PowerFactory*.

### 23.3.7 Example of a Python Script

The following example Python script calculates a load flow and prints a selection of results to the output window. The script can be executed from within *PowerFactory*.

```
if __name__ == "__main__":
    #connect to PowerFactory
    import powerfactory as pf
    app = pf.GetApplicationExt()
    if app is None:
        raise Exception("getting PowerFactory application failed")

    #print to PowerFactory output window
    app.PrintInfo("Python Script started..")

    #get active project
    prj = app.GetActiveProject()
    if prj is None:
        raise Exception("No project activated. Python Script stopped.")

    #retrieve load-flow object
    ldf = app.GetFromStudyCase("ComLdf")

    #force balanced load flow
    ldf.iopt_net = 0
```

```

#execute load flow
ldf.Execute()

#collect all relevant terminals
app.PrintInfo("Collecting all calculation relevant terminals..")
terminals = app.GetCalcRelevantObjects("*.ElmTerm")
if not terminals:
    raise Exception("No calculation relevant terminals found")
app.PrintPlain("Number of terminals found: %d" % len(terminals))

for terminal in terminals:
    voltage = terminal.GetAttribute("m:u")
    app.PrintPlain("Voltage at terminal %s is %f p.u." % (terminal , voltage))

#print to PowerFactory output window
app.PrintInfo("Python Script ended.")

```

## 23.4 Editor

*PowerFactory* has an integrated editor which can be used to write and execute scripts or as normal text editor. The editor can be reached by pressing “ctrl e” on the keyboard or from the script page of a ComDpl/ComPython dialog by right-clicking somewhere in the code and selecting “Open Text Editor”.

The editor will open as a tab in the active tab group, and the available tools are shown in the *Editor Toolbar*; note that the same tools are available when using the page *Script* of ComDpl/ComPython command by using the context menu.

-  Open external text file
-  Save file. If the file is a script the changes will be saved to the script object. Otherwise the file will be saved as a .txt file
-  Print the current file
-  Check the syntax of the script. Only available for DPL scripts
- ▶ Execute the current script. The button is interpreted as save and execute.
-  A browser with the script contents is shown.
-  With this *Edit* icon the *edit* dialog of the script is opened.
-  Cut part of the text.
-  Copy part of the text.
-  Paste the copied or cut text.
-  The text of the script will be deleted
-  Undo the last operation.
-  Redo the last operation.
-  With the *Search* icon the user can activate a *Find*, a *Replace* or also a *Go To* function inside the editor.
-  Find/replace/go to the next matching word.
-  Find/replace/go to the previous matching word.

 With this icon bookmarks can be set in the editor.

 Go to the next bookmark.

 Go to the previous bookmark.

 Clear all the existing bookmarks.

 Open the User Settings dialog in the *Editor* page. More information is given in Section 7.7.

When editing is complete, press the  icon or  icon and the script will be synchronised with the main dialog.

The editor can be closed by clicking on the x in the tab. If not closed, the edit window will be retained as part of the active desktop and will be available again when that desktop is reactivated. It is possible to modify this by right-clicking on the tab and selecting “Detach from Desktop”.

## 23.5 Add On Modules

The purpose of Add On Modules is to allow the user more flexibility in the processing of calculations and the presentation of results. By using Add On Modules the user can, for example, execute a number of different calculations and present the results together on a graphic or in a report. Furthermore, the concept of user-defined variables is introduced, these being variables created by the user in order to store results, parameters, or any other information relevant to the process. After a module has been executed, the user-defined variables may be accessed in the same way as the standard results variables, for example being selected to display in a flexible data page. An Add On Module is created and executed using an Add-On Command (*ComAddon*), which can be stored within a project or stored in a central location and accessed via the user-defined toolbar.

### 23.5.1 Add On Module framework

The Add On Module is created using an *ComAddon* command, which typically consists of the following components:

- A DPL or Python script, which:
  - uses a *CreateModule* command to create the Add On module;
  - may include a range of calculation commands such as *ComLdf* and *ComShc*;
  - includes commands to transfer results variables or other information to the user-defined variables;
  - uses a *FinaliseModule* command to complete the Add On module and make the results in the user-defined variables available to the user.
- One or more User-defined variable definitions *IntAddonvars*; these are where the user specifies a set of user-defined variables for a particular element class.

As can be seen in Figure 23.5.1 below, the Add On command object also specifies two other pieces of information:

**Module Name:** This is the name that will be given to the module itself. When the command is executed, the module is created and this name will appear, for example, when the user selects variables to display on a flexible data page.

**Module Key:** This is a key used internally by *PowerFactory*. It is used as a reference so that the appropriate flexible data, graphic colouring and result boxes are used when the command is executed. It is possible, if appropriate, for several Add On commands to use the same module key.

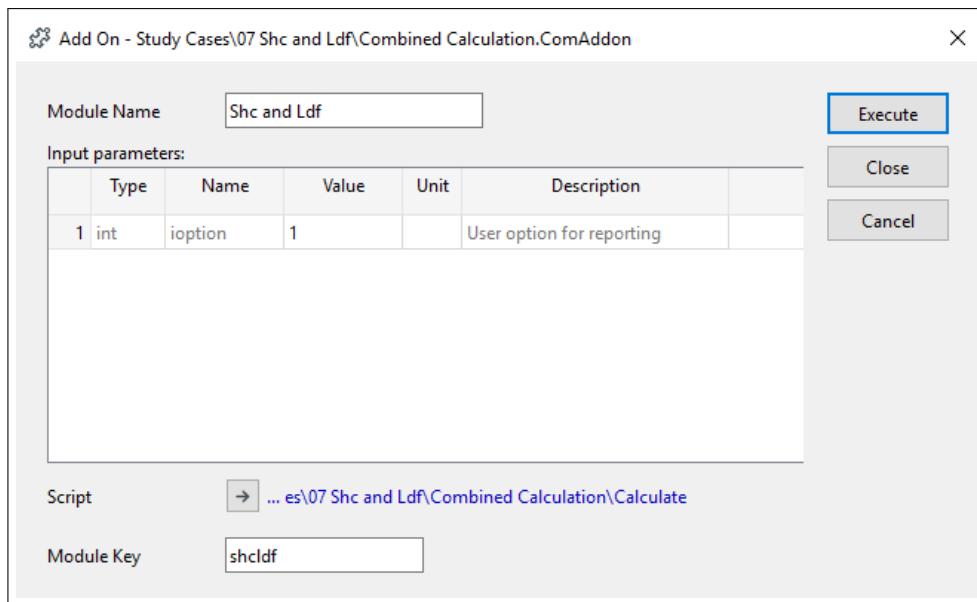


Figure 23.5.1: Add On Command object

So when the Add On command is executed, the Add On module that is created can be regarded as a bespoke *PowerFactory* function, handled in the same way as other functions such as a load flow or RMS simulation, but offering great flexibility for the user.

### 23.5.2 Creating a new Add-on Module command

To create a new Add On Module in a study case, the following steps can be followed:

- In a Data Manager, select the study case and click on the *New Object* icon
- Use the filter to select from the list an “Add On” command (*ComAddon*) and confirm with OK.
- Give the Add On a name and a key and confirm the edit dialog with Close.
- Select the Add On on the left side in the tree hierarchy of the Data Manager and press the *New Object* icon again.
- Four options are available initially. The DPL Command (*ComDpl*) or Python Script(*ComPython*) can be selected to create the script for the module. The third option, Variable Definition of Add On (*IntAddonvars*) is used to create the user-defined variable definitions (*IntAddonvars*). The fourth option, Flexible Data, offers the possibility to define a new tab (like the flexible data page) and a predefined set of displayed variables.
- Once a script exists, any additional objects created in the module can only be user-defined variable definitions *IntAddonvars* or the definition of an additional flexible data page *SetFoldflex*.
- The *IntAddonvars* is configured to set up the user-defined variables. In the example in Figure 23.5.2 below, short-circuit results variables are defined, one for each phase, and also line loading, line length and an internal counter from the script.

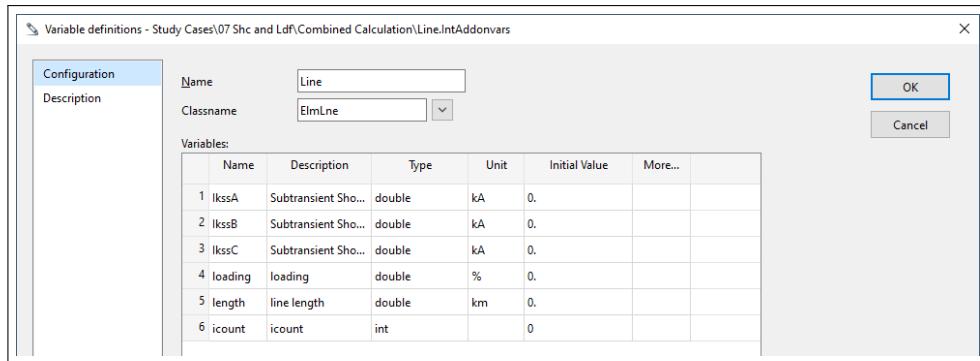


Figure 23.5.2: Example of user-defined variable definitions for line elements

The setup of an additional flexible data page could be done as follows:

- Create a Flexible Data definition (*SetFoldflex*) inside the Add On.
  - Execute the Add On.
  - Open the Network Model Manager and look at the object classes for which Add On variable definitions were created.
  - Switch to the newly created, user defined flexible data tab at the bottom.
  - Add user defined and whatever parameters of interest to the displayed variables via the *Variable Selection* .
  - With the variable selection done for all object classes of interest, the corresponding settings can be copied into the Add On. Navigate in the Data Manager to the Settings folder of the project and within it to the newly created flexible data page.
  - Copy the *Flexible Page Selector* objects (*IntMonsel*) and paste them inside the Add On Flexible Data definition (see Figure 23.5.3).
  - When this Add On is then executed in another project, the additional flexible data page will contain the configured sets of variables.

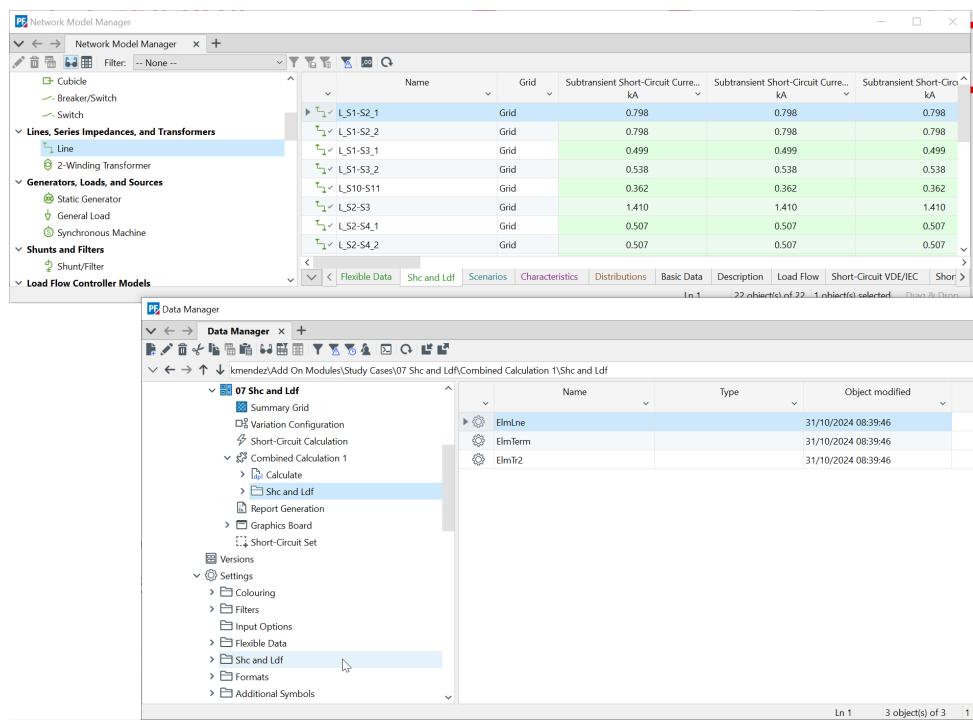


Figure 23.5.3: Definition of additional flexible data page and corresponding variable selections for the object classes.

For the user-defined variables, supported data types include integer, double, string, object (reference), arrays and matrices; for edge elements, variables can be defined as per phase and/or per connection quantities.

Within the script itself, important features are the command `CreateModule()`, which is followed by all the required calculations and data manipulation, new DPL and Python methods used for the handling of the variables, and the `FinaliseModule()` command, which is used once the calculations and data manipulation are complete and which defines the point at which no more changes are made to the user-defined variables and the results are ready to be viewed.

### 23.5.3 Executing an Add-on Module command

Add-on Modules can be executed directly or from an icon on the Additional Tools toolbar as shown in Figure 23.5.4 below. Clicking on this icon will bring up a list of all Add On commands stored in the active study case or in the Add On folder in the Configuration area (see Section 23.5.4).



Figure 23.5.4: Running Add On modules from the Additional Tools toolbar

When the script has been executed, it is possible to access the user-defined variables in a flexible data page as shown in Figure 23.5.5 below, or add them to a result box.

If an additional flexible data page is defined in the Add On, the corresponding flexible data page can be selected and further adapted.

In Figure 23.5.5, it can be seen how the Add On Module name appears on the *Calculation* filter together with the standard *PowerFactory* calculation functions.

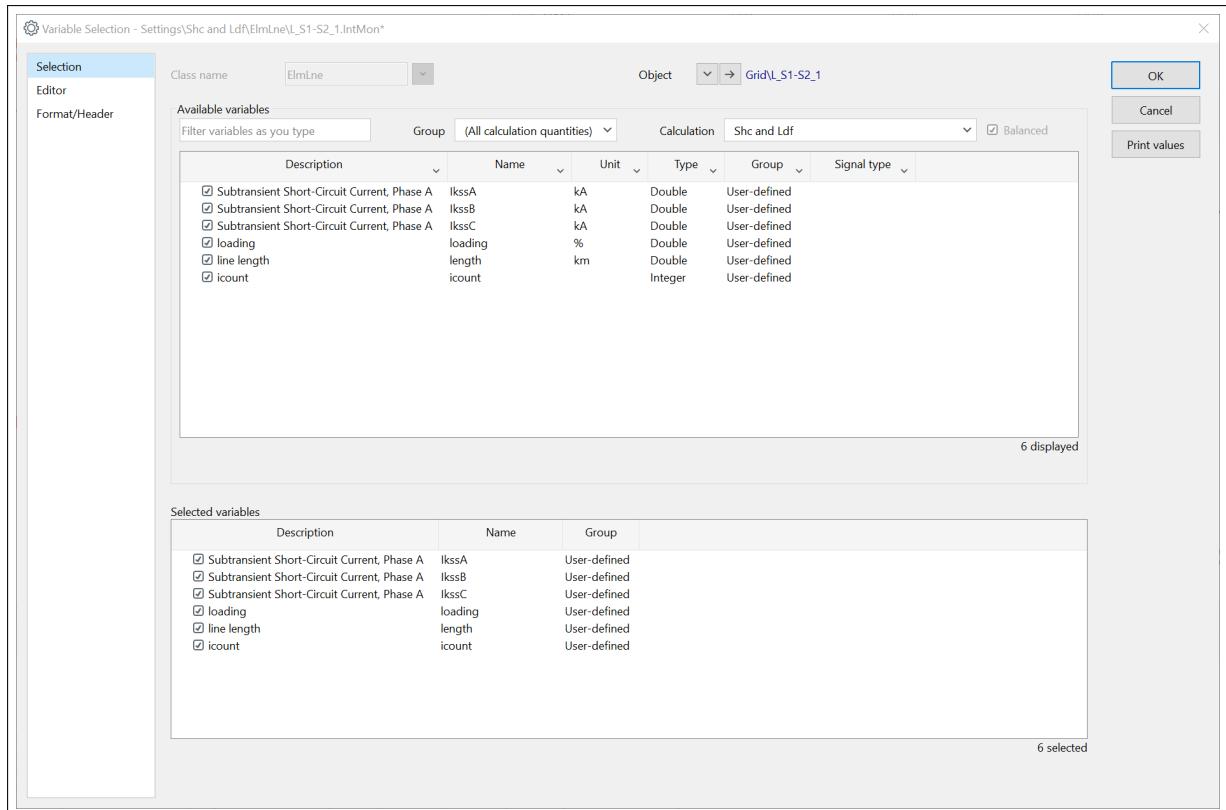


Figure 23.5.5: Selecting user-defined variables for a flexible data page

### 23.5.4 Adding Add On Modules to the User-Defined Tools toolbar

Add On Modules can be made available for use in different projects and (in the case of a multi-user database) by other users, by adding the Add-On commands to the User-defined Tools toolbar. Section 6.7.1 describes how the User-defined Tools toolbar is configured; the Add On Module command, including contents, is placed in the Add On folder of the Configuration folder.

## 23.6 Command Buttons

Command Buttons (*VisButton*) are a convenient way to allow DPL or Python scripts to be easily executed by clicking on a button in a single line diagram. The creation and attributes of a Command Button are described in the [Network Graphics Chapter](#), in Section 10.7.

# Chapter 24

## Interfaces

### 24.1 Introduction

*PowerFactory* supports a wide set of interfaces. Depending on the specific data exchange task the user may select the appropriate interface.

The interfaces are divided as follows:

- Interfaces for the exchange of data according to *DIGSILENT* specific formats:
  - DGS
  - *StationWare* (*DIGSILENT* GmbH trademark)
- Interfaces for the exchange of data using proprietary formats:
  - PSS/E (Siemens/PTI trademark)
  - NEPLAN (NEPLAN AG trademark)
  - INTEGRAL
  - PSS/SINCAL (Siemens/PTI trademark)
  - Functional Mock-up Interface (Modelica Association trademark)
- Interfaces for the exchange of data according to standardised formats:
  - UCTE-DEF
  - CIM
  - OPC
- Programming interfaces for integration with external applications
  - C++ API

The above mentioned interfaces are explained in the following sections.

### 24.2 DGS Interface

DGS (**DIGSILENT**) is *PowerFactory*'s standard bi-directional interface specifically designed for bulk data exchange with other applications such as GIS and SCADA, and, for example, for exporting calculation results to produce Crystal Reports, or to interchange data with any other software package.

Figure 24.2.1 illustrates the integration of a GIS (Graphical Information System) or SCADA (Supervisory Control And Data Acquisition) with *PowerFactory* via the DGS interface

Here, *PowerFactory* can be configured either in GUI-less or normal mode. When used in GUI-less mode (engine mode), *PowerFactory* imports via DGS the topological and library data (types), as well as operational information. Once a calculation has been carried out (for example a load flow or short circuit), the results are exported back so they are displayed in the original application; which in this example relates to the SCADA or GIS application. The difference with *PowerFactory* running in normal mode (see right section of Figure 24.2.1) is that, besides the importing of data mentioned previously, the graphical information (single line graphics) is additionally imported, meaning therefore that the results can be displayed directly in *PowerFactory*. In this case, the exporting back of the results to the original application would be optional.

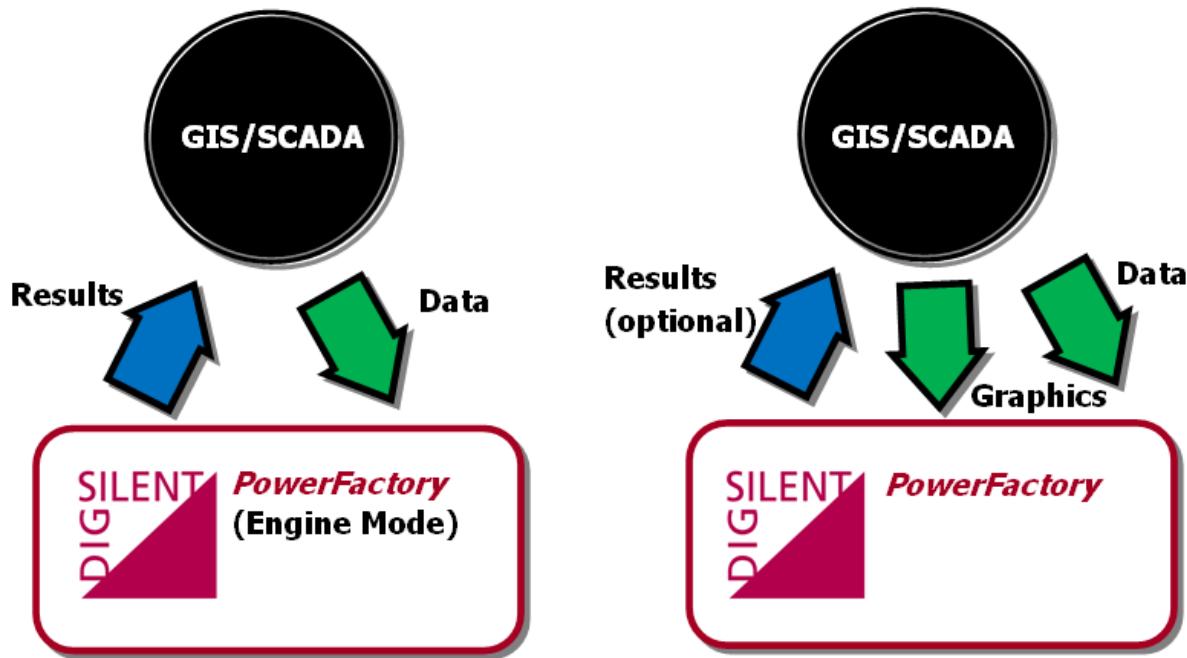


Figure 24.2.1: DGS - GIS/SCADA Integration

Although the complete set of data can be imported in *PowerFactory* every time a modification has been made in the original application, this procedure would be impractical. The typical approach in such situations would be to import the complete set of data only once and afterwards have incremental updates.

### 24.2.1 DGS Interface Typical Applications

Typical applications of the DGS Interface are the following:

- **Importing to PowerFactory**
  - Data Import/Update into *PowerFactory* from external data sources such as GIS (Network Equipment), SCADA (Operational Data) and billing/metering systems (Load Data) in order to perform calculations.
- **Exporting from PowerFactory**
  - Performing calculations in *PowerFactory* and exporting back the results to the original application.
- **Integration**
  - Importing data sets to *PowerFactory* from GIS or SCADA, performing calculations, and exporting back results to GIS or SCADA.

### 24.2.2 DGS Structure (Database Schemas and File Formats)

*PowerFactory*'s DGS interface is strictly based on the *PowerFactory* data model. Data can be imported and exported with DGS using different file formats and database schemas.

The following databases or file formats are supported:

- **Databases**
  - Oracle DB Server (ODBC client 10 or newer)
  - Microsoft SQL Server (ODBC driver 2000 or newer)
  - System DSN (ODBC)
  - Generic ODBC
- **File Formats**
  - DGS File - ASCII
  - XML File
  - Microsoft Excel File (2003 or newer)
  - Microsoft Access File (2003 or newer)

Important to note here is that the content of the files is the same, the only difference being the format.

---

**Note:** Due to changes in the format, DGS is available in several versions. It is highly recommended to always use the latest available DGS version.

---

The core principle of DGS is to organise all data in tables. Each table has a unique name (within the DGS file or database/table space) and consists of one or more table columns, where generally all names are case-sensitive.

More information on DGS and examples can be accessed by selecting from the main menu *Help* → *Additional Packages*→ *DGS Data Exchange Format*

### 24.2.3 DGS Import

To import data via the DGS interface, the general procedure is as follows:

- From the main menu go to *File* → *Import*→ *DGS Format...* which opens the DGS-Import dialog.
- Specify the required options in both the *General* and *Options* pages, and click on the Execute button.

When importing DGS files, the user has two options:

1. Importing into a new project. With this option selected a newly generated project is left activated upon completion.
2. Importing into an existing project. If an operational scenario and/or a variation is active at the moment the import takes place, the imported data set will be divided correspondingly. For example importing breaker status (opened/closed) while an operational scenario is active will store this information in the operational scenario.

All the options for the DGS Import, depending on the selected format and DGS version, are described in the DGS Documentation, Section [Import](#).

## 24.2.4 DGS Export

In contrast to the *DGS Import*, where it is not relevant if a project is active or not; the *DGS Export* is based on what information is active at the moment the export takes place. In other words, only the active project, with the corresponding active *Study Case*, active *Scenario*, and active *Variations* are exported (objects are exported in their current state). Furthermore, the export can be fully configured, meaning that the user has the option of selecting the amount of information to be exported per class object. In general, the following data can be exported:

- Element data
- Type data
- Graphic data
- Result data (e.g. load flow results)

To export data via the DGS interface, the general procedure is as follows:

- Create an Export Definition.

---

**Note:** Pre-defined export definitions are provided with the *PowerFactory* installation and are available by selecting from the main menu *Help* → *Additional Packages* → *DGS Data Exchange Format*

---

- Activate the project to be exported, considering which *Study Case*, *Scenario* and *Variations* should be active.
- From the main menu go to *File* → *Export...* → *DGS Format...* which opens the DGS-Export dialog.
- Specify the required options in both the *General* and *Options* pages, and click the **Execute** button.

All the options for the DGS Export, depending on the selected format and DGS version, are described in the DGS Documentation, Section [Export](#)

## 24.3 ANAREDE and ANAFAS Interface

ANAREDE is a network calculation software product from Electrobras Cepel (Brazil), for the analysis of steady state phenomena in electrical power systems. ANAREDE \*.pwf files can be imported and exported.

Similarly, importing and exporting of ANAFAS \*.ana files is possible, where ANAFAS is Electrobras Cepel's network calculation software for short-circuit analysis in electrical power systems.

The import interface also supports the import of \*.lst files, which can be used to provide single-line diagrams. An \*.lst file can be imported together with a \*.pwf or \*.ana file, or can be imported separately into an active project.

## 24.4 PSS/E File Interface

Although both import and export functions for PSS/E files are integrated commands of *PowerFactory*, the export function is licensed separately. For more information on prices and licensing contact the sales department at [mail@digsilent.de](mailto:mail@digsilent.de).

PSS/E Import supports versions 27 to 35 and can be carried out by going to the main menu and selecting *File* → *Import* → *PSS/E...*

In the same manner, and provided the appropriate licensing exists, a project can be exported in PSS/E format by selecting from the main menu *File* → *Export* → *PSS/E*....

For exports, PSS/E versions 27 to 33 are supported.

#### 24.4.1 PSS/E File Types and Versions

*PowerFactory* is able to convert the following PSS/E 'Source Format' file types:

**RAW** Power Flow Data (Import and Export) \*.raw or \*.rawx extension.

**SEQ** Sequence Data (Import and Export) \*.seq extension.

**DYR** Dynamics Data (Import and Export) \*.dyr extension.

*PowerFactory* is not able to convert the following PSS/E 'Binary Format' file types:

**SAV** Power Flow Data "Saved Case": needs to be converted to RAW format in PSS/E.

**SNP** Dynamics Data "Snapshot": needs to be converted to DYR format in PSS/E.

**SLD** Single Line Data "Slider Data": not supported by *PowerFactory*.

**DRW** Single Line Data: not supported by *PowerFactory*. Instead, diagrams can be created in *PowerFactory* using the diagram layout tool.

#### 24.4.2 Importing PSS/E Data

*PowerFactory* is able to convert both steady-state data (for load-flow and short-circuit analysis) and dynamic data files.

Before starting the next steps for importing a PSS/E file, make sure that no project is active. Once this has been confirmed, select from the main menu *File* → *Import* → *PSS/E*. The *PSS/E Import* command dialog will be displayed, asking the user to specify various options.

##### 24.4.2.1 Basic Options

- **Raw file:** Location of the PSS/E raw data file. By default the program searches for \*.raw or \*.rawx extensions.
- **Sequence file:** Location of the PSS/E sequence data file. By default the program searches for \*.seq extensions.
- **Dynamics file:** Location of the PSS/E dynamics data file. By default the program searches for \*.dyr extensions.

---

**Note:** The drw file is no longer converted/imported. Instead, network graphics can be generated after import into *PowerFactory*, using the *Diagram Layout Tool*.

---

##### Import into

- **Name:** The project name that will be assigned to the converted/imported file in *PowerFactory*.
- **in:** Location in the Data Manager tree where the imported file will be stored.

### 24.4.2.2 Advanced Options

**Unit of length:**

Lengths in the PSS/E raw files can be interpreted as kilometres or miles.

**Additional Parameters:**

This field does not need to be used.

### 24.4.3 Exporting a project to a PSS/E file

This function allows the export of the network model in PSS/E format. The export comprises both steady-state and dynamic data sets. The correct conversion of dynamic models is only possible for the models in folder *DIGSILENT Library → Dynamic Models → Synchronous Generator Power Plants → PSS/E compatible*. Models which the user implemented in *PowerFactory*'s DSL can not be automatically translated and must be modelled as user-defined controller types separately in PSS/E.

To export a project in PSS/E format select *File → Export... → PSS/E...* from the main menu.

#### 24.4.3.1 Basic Options

**PSS/E Version** - Version of the exported PSS/E file (27 to 33).

**Raw file** - Path and file name for the PSS/E RAW file, containing the symmetrical description of the model. By default the program selects the \*.raw extension.

**Sequence file** - Path and file name for the PSS/E SEQ file, containing the additional description of the model necessary for unbalanced conditions. By default the program selects the \*.seq extension.

**Dynamics file** - Path and file name for the PSS/E DYN file, containing the dynamic models of the project. By default the program selects the \*.dyn extension.

#### 24.4.3.2 Advanced Options

**Convert Motors to Generators if P<0** - With this option enabled, all asynchronous machines in generator mode will be converted to synchronous machines.

**Export branch as single equivalent line** - Selecting this option will convert the branch models to an equivalent line.

**Convert SVS to generator** - This option defines how the SVS elements will be exported. Three options are available:

- **No:** the SVS elements won't be exported.
- **Only voltage controlled:** will convert the SVS elements with control mode set to *Voltage Control* (Load Flow page of the element) to generator models.
- **Always:** all the SVS elements will be converted to generator models.

**Base Apparent Power** - Base for the power values given in per-unit system.

**Min (Zero) Impedance Branch** - Minimum impedance for ideal connections.

**PSS/E Bus Number** - This option defines the naming convention when exporting terminals *ElmTerm*. Three options are available:

- **Automatic:** the number assigned will be according to the name (in ascending/alphabetical order).

- **Use Serial Number:** the serial number information stated in the *Description* page of each terminal will be used for assigning the PSS/E bus number.
- **Use Characteristic Name:** the characteristic name information stated in the *Description* page of each terminal will be used for assigning the PSS/E bus number.

**Export PSS/E-Area index as** - The way the Area index is defined in PSS/E is defined here; two options are available:

- **Grid:** the exported file will have the areas defined according to the Grids defined in the *Power-Factory* model.
- **Area:** the exported file will have the areas defined according to the Areas defined in the *Power-Factory* model.

**Additional Parameters:**

This field does not need to be used.

## 24.5 PSS/U Interface

*PowerFactory* offers the user the option to import PSS/U files. The following files are supported:

- \*.dat

In addition, data dictionaries can also be imported.

### 24.5.1 Importing PSS/U Data

To import data via the PSS/U interface, the general procedure is as follows:

- From the main menu go to *File* → *Import* → *PSS/U...* which opens the *Convert PSS/U files* dialog.
- Click on the dots (...) to select the \*.dat-file and specify the required options and click on the **Execute** button.

Once the import process has been executed, the project (new or existing) is left activated upon completion.

The following section describes each of the dialog options.

#### 24.5.1.1 General Settings

**Nominal Frequency** Nominal frequency of the file to be converted/imported.

**File Type** Choose the \*.dat file from your system that should be converted.

**Save Converted Data in** *PowerFactory* offers the possibility to import the project into an existing project or to create new project for the import. If no project is selected *PowerFactory* will create a new project.

**Common Conversion Settings** Following settings are available:

- Line Length in miles instead of km: The unit of the line length of the input file.
- Import Graphic Information: A graphic will automatically be created based on the information of the input file.
- Create switch within branches: If this option is selected switches are created at the beginning and the end of every branch. It is recommended to choose this option, so that lines can be switched off later on.

**Construction Data Dictionary** Choose the \*.con file with the corresponding construction data dictionary. If no dictionary is selected the types are created automatically based on the \*.dat file.

**Machine Data Dictionary** Choose the \*.mot file with the corresponding machine data dictionary. If no dictionary is selected the types are created automatically based on the \*.dat file.

**Additional Parameters** Additional parameters for the import. This field is normally left blank.

#### 24.5.1.2 Options

On the options page additional parameters regarding the graphic are specified.

**Automatic Drawing of Network Elements** Defines which elements should be drawn automatically.

**Offset Factor for semi-orthogonal branches** Sets the offset factor for semi-orthogonal line routing in *PowerFactory*. This factor is applied if the nodes in PSS/U were plotted as bars rather than points.

**Scaling Factor** The scaling factor defines the graphical distance between the nodes.

## 24.6 PSS/ADEPT Import Converter

PSS/ADEPT is a former network calculation software product provided by Siemens, and for a number of years the import of older PSS/ADEPT formats (PSS/U) into *PowerFactory* has been possible. PSS/ADEPT HUB and Construction Dictionary files can be imported.

HUB files (\*.dmp) are ASCII files that contain all the significant data of the network, including the graphics. Construction Dictionary files (\*.con) contain line and cable type data.

Construction Dictionary files are not mandatory for the import.

The import dialog can be started in two ways:

- File - Import - PSS/ADEPT...
- Data Manager - Input Window - ed/adept

## 24.7 NEPLAN Interface

The NEPLAN interface in *PowerFactory* allows the import of NEPLAN files exported from version 5 of NEPLAN, specifically:

### NEPLAN 5

- Node Table (\*.ndt) containing the node data, such as rated voltages and loads.
- Element table (\*.edt) containing the branch data, such as lines and transformers.
- GIS/NMS Interface (\*.cde) containing the graphical information of all the networks which are part of the NEPLAN project.

#### 24.7.1 Importing NEPLAN Data

To import data via the NEPLAN interface, the general procedure is as follows:

- From the main menu go to *File* → *Import* → *Neplan...* which opens the NEPLAN-Import dialog.

- Specify the required options and click on the **Execute** button.

The NEPLAN data import always creates a new *PowerFactory* project. Once the import process has been executed, the newly generated project is activated.

The user has the option of importing the data with or without graphical information. That is, if the user selects importing the data without graphical information, only the topological and electrical data will get imported, and no single line graphic will be generated. In order to import NEPLAN 5 graphics, the path to the NEPLAN files should not contain spaces.

### **Importing NEPLAN 5 files**

When importing NEPLAN 5 files, the user is only required to select the \*.ndt. By doing so, the corresponding \*.edt file is automatically imported also. This basically means that a \*.edt file must be present otherwise the import will not be executed. The \*.cde file is however optional. Additionally, all three files must have the same name and must be in the same folder.

### **Importing NEPLAN 5 data from an Access database**

Alternatively, the NEPLAN data can be provided via an Access database (\*.accdb format).

---

**Note:** To open the NEPLAN database file (\*.accdb) during the import to *PowerFactory*, MS Access is required.

A NEPLAN import error message may appear if the Microsoft Office installation type is different from that of *PowerFactory*. If Microsoft Office 32bit is used, Microsoft only installs the 32bit ODBC driver by default. *PowerFactory* (which is 64bit) needs the 64bit ODBC driver.

The user should install the missing driver from here:

<https://www.microsoft.com/en-US/download/details.aspx?id=13255>

We recommend “Microsoft Access Database Engine 2010 Redistributable” instead of the newer 2016 version, which requires more rights and a work-around for a successful installation.

---

#### **24.7.1.1 Basic Options**

##### **Neplan Files**

- Location on the hard disk of the NEPLAN \*.ndt file or \*.accdb.

##### **Import into**

- **New Project:** The project name that will be assigned to the converted/imported file in *PowerFactory*.
- **in:** Location in the Data Manager tree where the imported file should be stored.

#### **24.7.1.2 Advanced Options**

##### **Automatic busbar system detection**

No longer used.

### Import Graphic Information

If this option is enabled, and a \*.cde file made available, the graphical information is imported and the single line diagram is generated.

- **Additional Rotation Angle for 1-port Elements (deg):** If a value different than 0 is stated, then the single port elements (loads, generators, motors, etc.) are rotated counter clockwise (degrees) with respect to the original position.
- **Automatically scale to A0:** If this option is selected, then the graphic is rescaled according to the A0 page format.
- **User defined scaling factor:** allows the user to customise the scaling.

### Additional Parameters:

This field does not need to be used.

## 24.8 INTEGRAL Interface

*PowerFactory* offers the user the option to import Integral files for Load Flow and Short Circuit analysis. The following files are supported:

- \*.dvg
- \*.dtf
- \*.xml

Furthermore Integral files can be export as \*.xml files.

### 24.8.1 Importing Integral Data

To import Integral data, the procedure is as follows:

- From the main menu go to *File* → *Import* → *Integral...* (this will open the Integral import dialog).

In the '**Import into**' field the user can enter a project name, and the *PowerFactory* user for this project can be selected.

The Integral data import always creates a new *PowerFactory* project.

The \*.xml Integral files contain graphical information. However, for older Integral files with the ending \*.dvg and \*.dtf it is necessary to select graphical data with the ending \*.bild.

More information about the Integral Import is available in the German version of the User Manual.

### 24.8.2 Export Integral Data

The Integral export converts the *PowerFactory* project into an \*.xml file in Integral format. Therefore '**XML Data**' must be defined as the path where to store the xml file. If the ending .xml is not given, it will automatically added.

More information about the Integral Export is available in the German version of the User Manual.

## 24.9 PSS SINCAL Interface

*PowerFactory* offers the user the option to import database files from PSS SINCAL for Load Flow and Short Circuit analysis. The following files are supported:

- \*.mdb (MS Access)
- \*.db (SQLite)

### 24.9.1 Importing PSS SINCAL Data

The procedure to import PSS SINCAL data is as follows:

- From the main menu go to *File* → *Import* → *Sincal...* (this will open the PSS SINCAL import dialog).
- Select the file location of the database file of the SINCAL project in the field *Database name*.
- In the *Import into* field, the user can enter a project name and the folder/location where the project should be created; this defaults to the active *PowerFactory* user folder.

The PSS SINCAL data import will always create a new *PowerFactory* project.

The SINCAL database files contain graphical information. This information is converted into a *PowerFactory* network diagram.

---

**Note:** To open an \*.mdb database during the import to *PowerFactory*, MS Access is required.

A SINCAL import error message may appear if the Microsoft Office installation type is different from that of *PowerFactory*. If Microsoft Office 32bit is used, Microsoft only installs the 32bit ODBC driver by default. *PowerFactory* (which is 64bit) needs the 64bit ODBC driver.

We recommend “Microsoft Access Database Engine 2016 Redistributable”.

---

## 24.10 UCTE-DEF Interface

In *PowerFactory*, both export and import of **UCTE-DEF** (Union for the Co-ordination of Transmission of Electricity - Data Exchange Format) is supported. The UCTE interface is currently intended for importing/exporting grid data of a country belonging to the former UCTE community.

The data contained in these files correspond basically to load flow and short circuit (3 phase) type data. Furthermore, it only considers specific UCTE voltage levels according to voltage level codes, as well as UCTE specific country codes, such as DK for Denmark, P for Portugal, etc. However, the export of non-compliant grids is possible, as explained in Section 24.10.2 below.

Important to note here is that from 1<sup>st</sup> of July 2009, **ENTSO-E** (European Network of Transmission System Operators for Electricity) took over all operational tasks of the 6 existing TSO associations in Europe, including the Union for the Coordination of Transmission of Electricity (UCTE).

For more information related to the UCTE format, refer to the ENTSOE website: <https://www.entsoe.eu>

## 24.10.1 Importing UCTE-DEF Data

To import data via the UCTE interface, the general procedure is as follows:

- From the main menu go to *File → Import → UCTE...* which opens the UCTE-Import dialog.
- Specify the required options and click on the **Execute** button.

Once the import process has been executed, the project (new or existing) is left activated upon completion.

The following section describes each of the UCTE import dialog options.

### 24.10.1.1 General Settings

#### Import into

**New Project** By choosing this option, a project will be created where all the UCTE data will be stored. The user will have the option of specifying a specific name and location (other than the default).

**Existing Project** By choosing this option, the UCTE data will be imported into an already existing project.

#### File Type

**Add UCTE Files** Location of the UCTE files. Two types of files are available: \*.uct and \*.ucte.

#### Options

**Import for DACF process** With this setting the user has the option to import the Day Ahead Forecast.

**Convert negative loads to generators** With this option enabled, negative loads defined in the UCTE file will be converted to generators in the *PowerFactory* model.

**Convert transformer equivalent to common impedance** With this option enabled, transformer equivalents defined in the UCTE file will be converted to common impedances in the *PowerFactory* model.

**Ignore reactive power limits for generators** With this option enabled, the reactive power limits of the generators defined in the UCTE file will be ignored .

#### Additional Parameters

This field is specified for internal use only. No extra information is required by the user.

## 24.10.2 Exporting UCTE-DEF Data

As in the other export interfaces, the *UCTE Export* is based on the **active** project at the moment the export takes place. To export data via the UCTE interface, the general procedure is as follows:

- Activate the project to be exported, considering the which *Study Case*, *Scenario* and *Variations* should be active.
- From the main menu go to *File → Export... → UCTE...* which opens the UCTE-Export dialog.
- Specify the required options, and click on the **Execute** button.

The following section describe each of these options.

### 24.10.2.1 General Settings

#### File Type

**UCTE Data** Location where the UCTE files are to be stored. Two types of files are available: \*.uct and \*.ucte.

#### Grids

Selection of which grids to export. The second column in this panel indicates for each grid whether it is UTCE-compliant or not.

#### Options

**Export UCTE voltage >=** Only the elements having a voltage greater than the UCTE voltage specified are exported.

**Export branch as single equivalent line** By enabling this option the export will convert the *PowerFactory* branch definitions into single equivalent lines.

**Use first character of characteristic name as branch order code** If checked, the characteristic name (first character) is used in the branch order code of the exported UCTE file.

**Allow export of non-compliant grids** If this option is selected, the export of one or more non-compliant grids is possible; they will be exported as one grid. A country code and country code node may be specified.

**Additional Parameters** This field is specified for internal use only. No extra information is required by the user.

## 24.11 CIM Interface

In *PowerFactory*, both export and import of **CIM** (**C**ommon **I**nformation **M**odel) is supported. The CIM interface is currently intended for importing/exporting the following profile:

- CGMES
- ENTSO-E 2009
- CIM LTDS

CIM is defined in IEC-61970, and its purpose is to allow the exchange of information related to the configuration and status of an electrical system.

For more information relating to CGMES, please see Section [24.12](#) below.

### 24.11.1 Importing CIM Data

To import data via the CIM interface, the general procedure is as follows:

- From the main menu go to *File* → *Import* → *CIM...* which opens the CIM-Import dialog.
- Specify the required options and click on the **Execute** button.

Once the import process has been executed, the project (new or existing) is left activated upon completion.

The following section describes each of the CIM import dialog options.

### 24.11.1.1 General Page

#### Import from

**Profile** Currently the profiles ENTSO-E 2009, CGMES 2.4.15 and CGMES 3.0.0, and CIM LTDS are supported.

**CIM File** Location of the CIM files. Two types of files are supported: \*.zip and \*.xml.

**Separated Files** With this setting the user has the option to import the equipment, topology and solved state files separately. This option is only available for ENTSO-E 2009.

#### Import into

**New Project** By choosing this option, a project will be created where all the CIM data will be stored. The user will have the option of specifying a specific name and location (other than the default).

**Active Project** By choosing this option, the CIM data will be imported into the active project.

#### Additional Parameters

This field is specified for internal use only. No extra information is required by the user.

## 24.11.2 Exporting CIM Data

As in the other export interfaces, the *CIM Export* is based on the **active** project at the moment the export takes place. To export data via the CIM interface, the general procedure is as follows:

- Activate the project to be exported, considering which *Study Case*, *Scenario* and *Variations* should be active.
- From the main menu go to *File* → *Export...* → *CIM...* which opens the CIM-Export dialog.
- Specify the required options, and click on the **Execute** button.

The following sections describe each of these options.

### 24.11.2.1 General Page

#### Export to

**Profile** Currently the profiles ENTSO-E 2009, CGMES 2.4.15 and CGMES 3.0.0, and CIM LTDS are supported.

**CIM file** Location on the hard disk where the CIM files will be stored. Two types of files are supported: \*.zip and \*.xml. Note that \*.xml files are only supported for ENTSO-E 2009 format.

### 24.11.2.2 Advanced Options

**Grids** Selection of which grids to export.

Following options are just available for ENTSO-E 2009 export. To perform a node reduction for CGMES export, the Grid to CIM command needs to be configured accordingly (see Section 24.12.4).

- Perform node reduction: Internal nodes and switches are reduced to create a bus-branch model.
- Convert motors to CIM SynchronousMachine

- Convert voltage controlled SVS to CIM SynchronousMachine

## 24.12 CGMES Tools

The CGMES Tools provide an additional interface to CIM. These tools are accessible via the main menu in *PowerFactory* under *Tools* → *CGMES Tools*.

The import and export of a CIM file is also accessible via *File* → *Import* → *CIM...* or *File* → *Export* → *CIM...*

This section describes these tools and uses the following naming conventions:

- **Model folder** stores all CIM data.
- **Archive** contains all corresponding CIM Models.
- **CIM Model** stores all data contained in a single instance file (mainly CIM Objects and namespaces).
- **CIM Object** stores all data contained in a single object.

The CGMES Tools separates each action (i.e. the import and export of CIM data) into two steps. To import data from a CIM file (XML or ZIP format) into *PowerFactory*, the first step is to import the CIM data into CIM objects (*CIM Data Import*). This offers the user the possibility to directly interact with the CIM data from within *PowerFactory*. The second step is to convert the CIM objects into a grid model (*CIM to Grid Conversion*). To export grids from *PowerFactory*, they must first be converted to CIM objects (*Grid to CIM Conversion*). These may still be modified if required, and also directly reimported. The newly-created CIM objects can then be exported as a ZIP or XML file in conformity with the CIM data structure (*CIM Data Export*).

### 24.12.1 CIM Data Import

CIM data can either be imported via the File or the Tools menu. The import via the File menu is explained in Section [24.11.1](#).

#### Import via CGMES Tools

The CIM Data Import is accessible in *PowerFactory* under *Tools* → *CGMES Tools* → *CIM Data Import*. The following options for the import location are available:

- **New archive in new project** creates a new project with the given name and imports the data into an archive with the same name.
- **New archive in active project** imports the data into a new archive with the given name, within the currently active project.
- **Existing archive in active project** imports the data into the given archive; data models that are already contained in the archive will not be modified.

#### Import of instance data belonging to a profile (XML file)

By selecting “**New archive in new project**”, the name of the project and the file to import can be specified. Upon clicking the **Execute** button, the content of the XML file will be imported. This step results in an active project containing the “CIM Model” folder, which itself has a single archive. This archive contains the model representations from the XML file.

### Import of multiple profiles instance data (ZIP file)

By selecting “**New archive in new project**”, the name of the project and the file to import can be specified. Upon clicking the **Execute** button, the content of the ZIP file will be temporarily extracted and imported. This step results in an active project containing the “CIM Model” folder, which has a single archive. This archive contains the model representations from the ZIP file.

### Import of multiple files

To import several files in a row, the destination **Existing archive in active project** must be used. The archive created by the first step must be selected as the “Path” for all subsequent files. This will import the data into the archive provided.

## 24.12.2 CIM Data Export

The CIM Data Export is accessible via one of the following options:

- *File → Export → CIM...*
- *Tools → CGMES Tools → CIM Data Export*
- *right-click → CIM Data Export* on a CIM Archive

The archive selected as “Source data” will be exported. This can be fine-tuned by an option for each available profile instance.

The option “Create archive for each CIM model” can be used to match the DACF and D2CF requirements by ENTSO-E to upload each individual profile within a separated zip file.

In all cases, the naming rules can be defined per profile. As these rules might be different for various processes, the naming rules can be configured on the “Advanced Options” page according to the individual needs for each profile.

## 24.12.3 CIM to Grid Conversion

The CIM to Grid Conversion is accessible under *Tools → CGMES Tools → CIM to Grid Conversion* or *right-click → CIM to Grid Conversion* on a CIM Archive.

To convert all data contained in an imported archive, select “CIM archives to convert” followed by **Execute**. This will additionally consider any valid difference models contained in the archive. To import a base model only, the difference models must be deleted before conversion. If only specific profile information should be exported, select the profiles that should be considered. It should be noted that the resulting subset of profiles still needs to be complete in terms of dependencies.

Additional information (e.g. SSH data) can be added to already converted models (i.e. EQ/TP); these must be selected as “CIM archives with additional data”.

To convert an archive including models that have dependencies on other models not included in the archive (e.g. the boundary grid), the other models must be specified as “CIM archives with additional data”.

For both selections “CIM archives with additional data” as well as “CIM archives to convert” the user may either select a single or multiple CIM archives in order to convert multiple archives at once or reference to multiple ones (e.g. the Boundary and the used EQ file) accordingly.

On the *Advanced option* page, all available profiles in the selected archive will be shown on per MAS basis in the “Modelling Authority Sets” table. These MAS will be linked to existing grid elements if a

matching rdfID is found, but can be adjusted manually if needed. Additionally the user can define how to deal with the models per MAS. Therefore, the options convert, link and ignore are available, where link means that the model is not converted, but only linked (e.g. an old version of a boundary is used). In case of conversion of an already existing (referenced) model, this will be updated in terms of new elements are added.

---

**Note:** A regular task in CGMES is to update the currently used boundary file. This can be achieved by referencing the old boundary grid in the CIM to Grid conversion and converting it. This way a merged boundary will be created (nodes that are used in the original Boundary that are not existing anymore in the new one will remain).

---

#### 24.12.4 Grid to CIM Conversion

The Grid to CIM Conversion is accessible under *Tools → CGMES Tools→ Grid to CIM Conversion*.

The following options for the destination are available:

- **New archive** converts the networks into a new archive with the given name.
- **Existing archive** imports the data into the archive specified in “Target Archive”; if the archive already contains models for the selected networks, the original models will be preserved.
- **CIM Archives with additional data** is used when references to other archives are required.

For “CIM Archives with additional data” the user may either select a single or multiple CIM archives for example to manage dependencies on the correct Boundary grid.

**CIM profile:** A drop-down menu allows the user to select which of the supported profiles should be used. In addition to the standard CGMES profiles, a CIM LTDS profile is available, to support the Long Term Development Statement process in Great Britain.

A model for each profile selected will be created per network in the target archive. By default all profiles are selected. Boundary grids will only be exported with EQ and TP profiles (the respective boundary versions).

##### 24.12.4.1 Advanced Options

Further information for the model to be converted can be altered under the *Advanced Options* page:

###### Convert network selection

The *PowerFactory* networks to be converted can be selected by ticking the checkbox. Only activated networks can be converted and each network to be converted must have a Modelling Authority Set (MAS) URI. If one of the networks is to be treated as a boundary grid, double-clicking on the cell in the Boundary column will bring up the dialog for the relevant *ElmNet*, where the option “Fictitious border grid” should be selected.

- Two or more networks can be associated to a common Modelling Authority Set. In such a case, all networks and data associated to a MAS will be exported into the same model set.
- If the selection contains two or more Modelling Authority Sets (apart from the Boundary network), and SV and/or DL profiles are selected for export, these instance data will be exported into an “Assembled” model set. Otherwise, if only one MAS is converted (apart from the Boundary network) SV and DL data will be exported into the same model set as EQ and TP models.

The **Model type** drop-down menu allows a selection between Full and Difference. Difference models can only be created if “CIM archives with additional data” has been set on the Basic Options page.

The **Topology Representation** drop-down menu offers the choice between Detailed (node-breaker) and Reduced (bus-branch). If the latter is selected, an internal reduction from a node-breaker model to a bus-branch model is done automatically and the resulting CIM archive will be a bus-branch model according to CGMES.

In addition, the following information can be optionally set:

- **Version** is an optional parameter to define the model version
  - **Description** is an optional parameter to describe the model
  - **Additional Parameters** are inputs causing specific behaviour and do not need to be used.
- 

**Note:** CGMES is very strict in defining the hierarchy of the network elements. Therefore the network selected for conversion needs to fulfil some requirements. Further information can be found in the relevant tutorial under *Help → Tutorial...*

---

## 24.12.5 CIM Data Validation

The CIM Data Validation is accessible under *Tools → CGMES Tools → CIM Data Validation* or *right-click → CIM Data Validation* on a CIM Archive.

This data validation is based on the UML profile information and can be used on CIM archives to validate their CGMES profile compliance. The archives used for this validation can be selected via “CIM Archives or Models”. Therefore a multiselection is possible.

---

**Note:** This build-in validator can be used to validate archives of third parties or in case there are issues in the conversion (e.g. missing dependencies). The validator is not officially supported by the ENTSO-E.

---

## 24.12.6 Import and Export of the EIC as additional parameter

### 24.12.6.1 Grid to CIM Conversion

The “IdentifiedObject.energylIdentCodeEic” attribute is not applicable to *PowerFactory* data-model. However, it is possible to assign an “EIC” to a *PowerFactory* element, by adding the following “User Attribute” entry at the end of the description field of *PowerFactory* elements:

```
<Attribute Name="EIC" Type="string">the code</>
```

For *PowerFactory* elements where no “EIC” is provided, no “IdentifiedObject.energylIdentCodeEic” is set in the corresponding CIM object.

### 24.12.6.2 CIM to Grid Conversion

The “IdentifiedObject.energylIdentCodeEic” attribute is not applicable to *PowerFactory* data-model, thus is converted as a “User Attribute” entry at the end of the description field in *PowerFactory* elements as follows:

<Attribute Name="EIC" Type="string">the code</>

If no “IdentifiedObject.energyIdentCodeEic” is set in a CIM object, the entry will not be created in the corresponding *PowerFactory* element. For CIM object classes which have no representation in *PowerFactory*, the “EIC” code is not converted, thus not visible in *PowerFactory* data-model (e.g. RegulatingControl, GeneratingUnit etc.).

## 24.13 Functional Mock-Up Interface

For a detailed description on the FMI refer to Section [30.14.1](#): Functional Mock-Up Interface.

## 24.14 OPC Interface

*PowerFactory*'s **OPC** interface is an asynchronous communication and data exchange mechanism used in process interaction and is widely applied in SCADA and control systems. This OPC implementation assumes that the *PowerFactory* software is executed as an OPC Client while the OPC Server is controlled via the external source. OPC server libraries are available from various manufacturers. An example of a freeware OPC Server is that available from Matrikon (“MatrikonOPC Simulation Server”).

*PowerFactory* supports both OPC DA (data access) and OPC UA (unified architecture) standards.

Figure [24.14.1](#) illustrates the integration of a SCADA system with *PowerFactory* via the OPC interface. In this OPC implementation, *PowerFactory* can be used either in GUI-less or normal mode. Some further characteristics of this integration include:

- OPC Client/Server exchange of any *PowerFactory* object parameter as well as any signal (bi-directional Data Exchange).
- *PowerFactory* listening mode to receive any data or signal from a registered OPC Server.
- *PowerFactory* sending mode to write back any data or signal to a registered OPC Server.

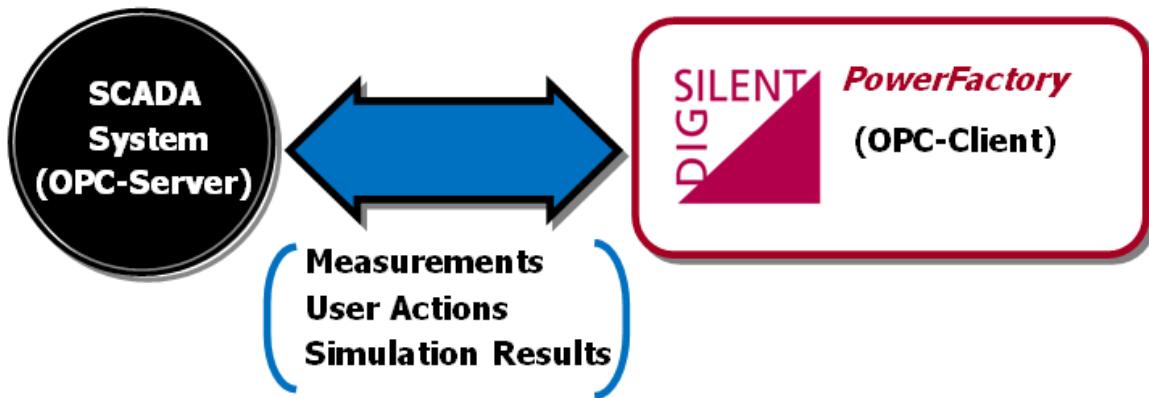


Figure 24.14.1: SCADA -*PowerFactory* integration via the OPC interface.

The OPC interface can be configured in two different modes:

- **Offline**
  - The bi-directional data exchange is carried out through an explicit command given by the user in *PowerFactory*. For example, by pressing a button predefined by the user in *PowerFactory*.

- **Online**
  - The bi-directional data exchange is automatically carried out at a certain frequency rate; where the frequency rate is determined by the user.

### 24.14.1 OPC Interface Typical Applications

Some typical applications of the OPC Interface are the following:

- **SCADA Online State Estimation**
- **SCADA Simulation Mode**, for example dispatcher load flow, switching validation.
- **SCADA Training Simulator**
- **Importing to PowerFactory**
  - in order to update the operational data.
  - in order to reflect the Operator actions, such as breaker status and tap positions.
  - in order to perform state estimation based on the measured network data.
- **Exporting from PowerFactory**
  - in order to update the SCADA interface with the calculated results.

## 24.15 StationWare Interface

This chapter describes the *StationWare* interface. An introduction into *StationWare* is provided in Section [24.15.1](#).

The following two sections describe the overall *StationWare* architecture (Section [24.15.2](#)) and the conceptual differences between *PowerFactory* and *StationWare* (Section [24.15.3](#)).

Both *PowerFactory* and *StationWare* have to be configured before they can be used together (Section [24.15.4](#)).

The *Getting Started* section (Section [24.15.5](#)) provides an introduction to the most important features. The complete documentation can be found in the section 'Description of the Menu and Dialogs' (Section [24.15.6](#)).

The terms *StationWare* and **PSMS** are used synonymously throughout this chapter. **PSMS** stands for Protection Settings Management System, and stresses the more internal and technical part of *StationWare*.

### 24.15.1 About StationWare

DIGSILENT *StationWare* is a centralised asset management system for primary and secondary equipment. It provides a reliable central protection settings database and management system for the complete power system data, both to manage the various control parameters and to centrally store power system related information and data, based on the latest .NET technology.

*StationWare* stores and records all settings in a central database, allows modelling of all relevant work flow sequences, provides quick access to device manuals, interfaces with manufacturer-specific relay settings software, and integrates with *PowerFactory* software, allowing powerful and easy-to-use settings co-ordination studies.

Modern numerical relays have a large number of settings that are determined, stored and communicated by proprietary software solutions (these may be suitable for only one particular manufacturer or

only one series or type of relay). This results in a fragmented and distributed settings “database”. *DIGSILENT StationWare* provides a single system that incorporates all different device protocols, thereby providing one manageable software data storage system, based on modern IT techniques, facilitating data interfacing and exchange in a transparent and straightforward manner.

*PowerFactory*’s data exchange facility allows it to access the settings stored in *StationWare*, such that these may be used as input to the powerful *PowerFactory* system simulation and protection settings tools. Settings that are calculated by using these tools may then be transferred back to *StationWare*.

## 24.15.2 Component Architecture

*DIGSILENT StationWare* is a so-called *Client-Server Application*: the functionality is distributed over at least two computers: client and server. Figure 24.15.1 gives an overview of the components.

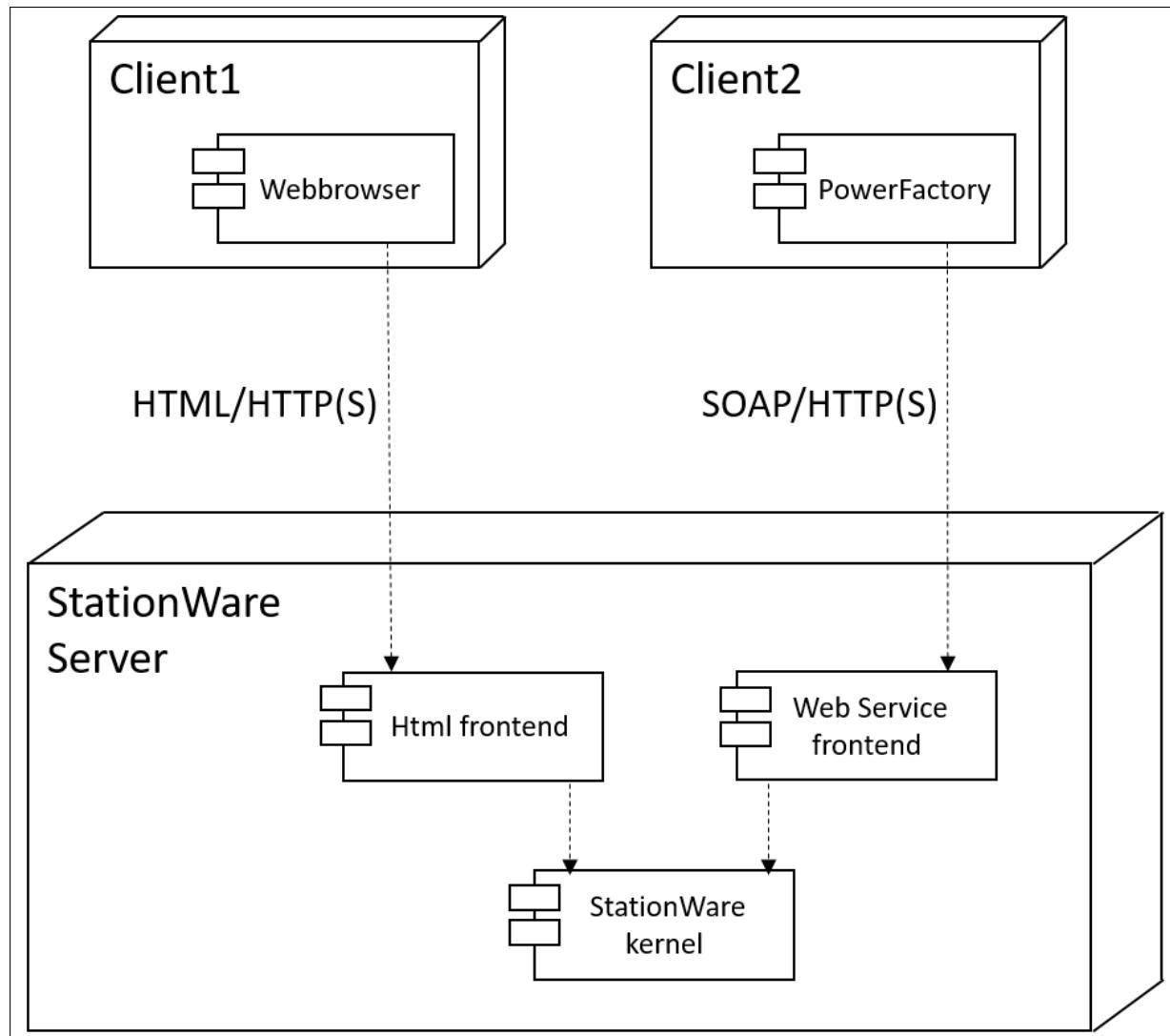


Figure 24.15.1: Architecture overview

There are usually several clients. One main advantage of this architecture is that the data is stored in one central database on the server. One client connects to the server and fetches the data from there, modifies it, and then stores it back to the server. These changes are visible on other clients.

*DIGSILENT StationWare* server provides two interfaces to access from client machines:

- Visualisation by means of a standard web browser. The HTML interface can be used with an usual web browser (e.g. Microsoft Internet Explorer or Mozilla Firefox).  
The browser displays HTML pages which are created by *StationWare*'s HTML front end. The HTML pages are transferred using the HTTP(S) protocol on top of the TCP/IP internet protocol. HTML allows to present all kind of data e.g. plain text, tables or images.  
Additionally HTML provides concepts to achieve interactivity: by submitting HTML forms or pressing on hyperlinks data is sent to the server. The server interprets such requests and creates new HTML pages which are displayed by the browser again.
- The web service interface, similar to the HTML interface uses the HTTP(S) protocol to communicate with the web service frontend, though no HTML pages are transferred but lower-level data (SOAP/XML encoded). The web service client application is responsible to present this data conveniently.  
*PowerFactory* is able to play the role of a web service client. It integrates parts of *StationWare*'s data and concepts smoothly into its own world.

---

**Note:** The default *StationWare* configuration requires SSL for the *StationWare* applications (web GUI and web services). Please use HTTP instead of HTTPS, if SSL is not enabled for your *StationWare* applications. In the following, the expression HTTP(S) is used.

---

The functionality of the HTML interface is covered in the *StationWare* manual. The remainder of this chapter focuses on *PowerFactory* as client.

### 24.15.3 Fundamental Concepts

Although *StationWare* and *PowerFactory* store data and settings associated with primary devices such as lines, transformers, ... and secondary devices, i.e. relays, CTs, VTs and circuit breakers, the two systems utilise different concepts to deal with this data.

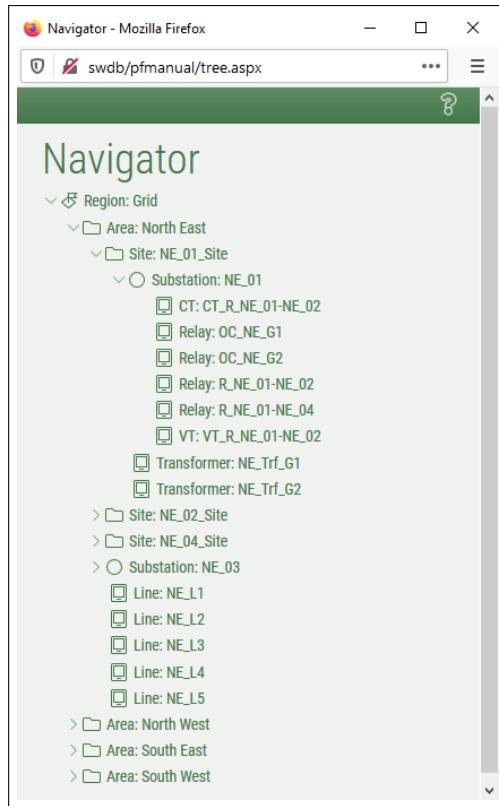
In *StationWare* it is possible to model a location hierarchy and associate the devices to nodes in this hierarchy (e.g. substations). This has no equivalent in *PowerFactory*, where the devices are stored inside the parent grid (*ElmNet*) object.

Conversely, *PowerFactory* allows to the creation of a topological representation of networks which is not supported in *StationWare*.

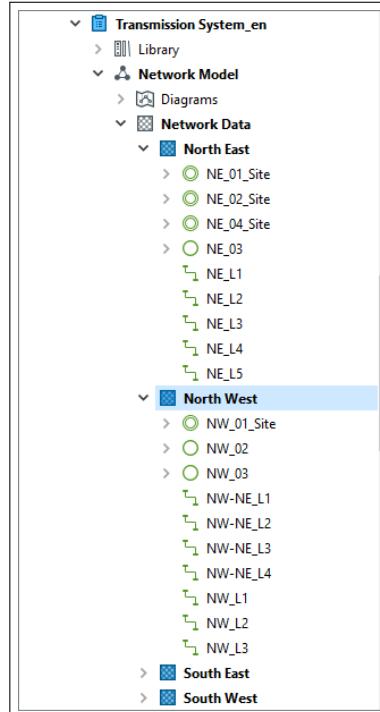
This section describes the concept mismatch between *PowerFactory* and *StationWare*. In order to use the *StationWare* interface, it is important to understand the differences between both applications.

#### 24.15.3.1 Location

In *StationWare* each device belongs to exactly one location. There are different location types e.g. *Region*, *Area*, *Site*, *Substation*, or *Bay*. The locations are organised in a hierarchy tree as shown in Figure 24.15.2.

Figure 24.15.2: *StationWare* locations

In *PowerFactory* the data is organised in projects (*IntPrj*). A project may have one or more grids (*ElmNet*) which in turn contain net elements e.g. terminals, cubicles, and relays (*ElmRelay*). See Figure 24.15.3 for a typical *PowerFactory* project.

Figure 24.15.3: *PowerFactory* project

*StationWare*'s location concept and *PowerFactory*'s project/grid concept hardly fit together. That's the reason why the data mapping between *PowerFactory* and *StationWare* begins at the device level which is the subject of the next sections.

### 24.15.3.2 Device

*StationWare* manages a set of devices e.g. relays, CTs, VTs, circuit-breakers, .... Each device is associated with a device type e.g. *ABB DPU2000R* or *SEL421 003*. In addition, each device has an unique ID: the *device ID*.

In *PowerFactory* a relay is represented by an *ElmRelay* object which references exactly one *TypRelay* object. The *ElmRelay* object contains several sub-components e.g. the *I>* component (a *RelToc* object), the Logic component (*RelLogic*), or the *Ios* component (*RelMeasure*). See Figure 24.15.4 for an example. The device ID is used to link one *StationWare* device to one *PowerFactory* device. The *PowerFactory* device e.g. an *ElmRelay* object stores the *StationWare* device ID as foreign key.

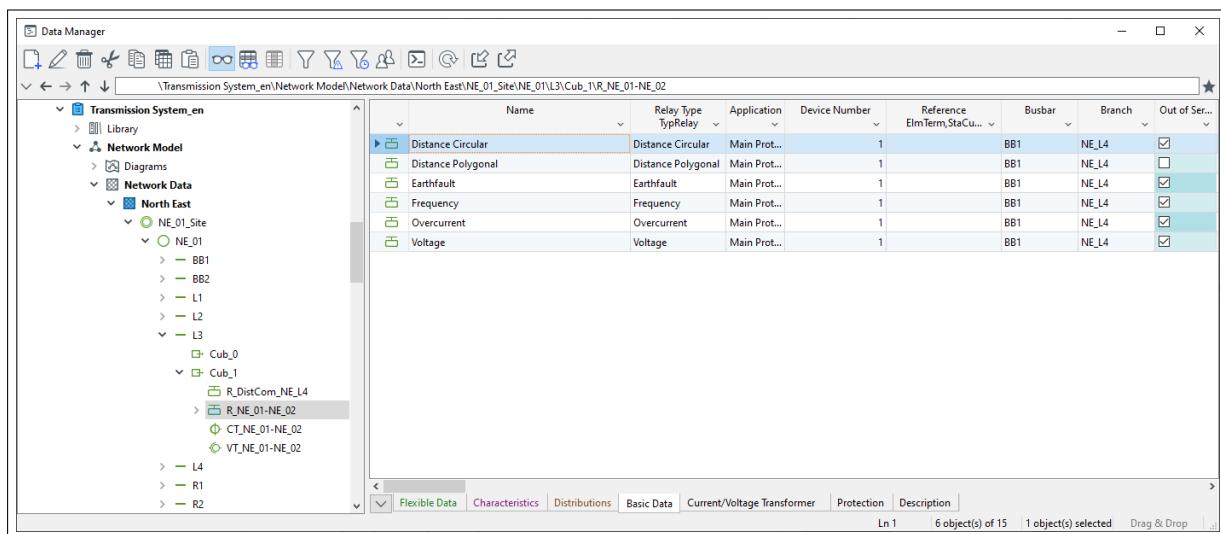


Figure 24.15.4: *PowerFactory* relay

### 24.15.3.3 Device State

A device's state is in *StationWare* called setting. A setting is a list of parameters, and describes the state of one device completely. An parameter is a tuple of

- parameter *name*,
- parameter *type* which can be an arbitrary integer or floating point number, optionally with a range restriction, or a string, or a enumeration type.,
- a *default* value,
- an optional *unit*.

A complex relay may have thousands of parameters. In *StationWare* the setting parameters are organised in so-called setting groups. A setting group groups the parameters together which belong somehow together. It's often defined by the device manufacturer. Each parameter belongs to exactly one setting group. Inside a group the parameter name is unique.

The device type defines which parameters and groups characterise a device. Table 24.15.1 shows an example of a possible device type. There are two setting groups G and H. Group G has the parameters a, b, and c, group H has the parameters d and e.

Group	Name	Type	Default	Unit
G	a	integer in [0,10]	0	A
	b	float	-0.32	l/s
	c	float in [0.03, 4.65]	1.0	
H	d	string	'DEFAULT'	
	e	enum 'yes', 'no', 'maybe'	'yes'	

Table 24.15.1: Settings Definition

According to this parameter definition a device can have settings as shown in tables [24.15.2](#) or [24.15.3](#).

Group, Name	Value
G,a	7
G,b	23.43
G,c	1.1
H,d	'abc'
H,e	'maybe'

Table 24.15.2: Settings Example 1

Group, Name	Value
G,a	8
G,b	0
G,c	1.1
H,d	'abcdef'
H,e	'yes'

Table 24.15.3: Settings Example 2

On the *PowerFactory* side there are neither settings nor groups. There is the *ElmRelay* object and its sub-objects. These objects can have parameters. See Table [24.15.4](#) for a definition and Table [24.15.5](#) for an example. The *TypRelay* type defines components and parameters.

*StationWare* parameters are mapped to *PowerFactory* parameters and vice versa. The mapping is non-trivial since only a small subset of the parameters (the calculation-relevant data) is modelled in *PowerFactory* and vice versa. Additionally there is no one-to-one relationship between the *StationWare* and *PowerFactory* parameters; i.e. a *PowerFactory* parameter may be calculated from several *StationWare* parameters.

Component	Parameter	Type
i>	o	integer
Logic	p	string
	q	enum 'enabled','disabled'
los	r	float
	s	float

Table 24.15.4: Parameter Definition

Some relays support *multiple setting groups* (MSG) also called *parameter sets*. Such relays have the same group many times (c.f. Table [24.15.5](#)). The groups H1, H2, and H3 have the same set of parameters (c and d). The relay models in *PowerFactory* do not support this concept. Instead of modelling all MSGs, only one instance of the H groups is provided.

In this case a group index parameter defines which of the MSGs actually is transferred from *StationWare* to *PowerFactory*.

#### 24.15.3.4 Lifecycle Phase

In *StationWare* each setting has one lifecycle phase e.g. *Planning* or *Applied*. At each point in time a device can have a set of settings e.g. three *Planning* settings, one *Applied* setting and 12 *Historic* settings.

Component Parameter	Value
i>:o	8
Logic:p	'HIGH'
Logic:q	'enabled'
los:r	18,5
los:s	19,5

Table 24.15.5: Parameter Example

Group	Name	Type	Default	Unit
G	a	integer in [0,10]	0	A
	b	float	-0.32	l/s
H1	c	string	'DEFAULT'	
	d	float in [0.03,1.65]	1.0	
H2	c	string	'DEFAULT'	
	d	float in [0.03,1.65]	1.0	
H3	c	string	'DEFAULT'	
	d	float in [0.03,1.65]	1.0	

Table 24.15.6: Multiple Setting Group Definition

In *PowerFactory* a device has exactly one state (or setting). Therefore when data is transferred between *PowerFactory* and *StationWare* always a concrete device setting in *StationWare* must be specified.

For *PowerFactory* purposes a special *PowerFactory* planning phase is introduced. The transfer directions are specified as follows:

- Imports from *StationWare* into *PowerFactory* are restricted to *Applied* and *PowerFactory* settings. *Applied* denotes the current applied setting (*Applied*) or a previous applied (*Historic*) setting.
- Exports from *PowerFactory* to *StationWare* are restricted to the *PowerFactory* setting. (*Applied* and *Historic* settings are read-only and can never be changed).

(Actually *PowerFactory*'s sophisticated variant management is similar to the phase concept, but there is no obvious way how to bring them together.)

#### 24.15.4 Configuration

In order to transfer data between *PowerFactory* and *StationWare* both systems must be configured.

##### 24.15.4.1 *StationWare* Server

An arbitrary *StationWare* user account can be used for the *StationWare* interface in *PowerFactory*. The user must have enough access rights to perform operations e.g. for the export from *PowerFactory* to *StationWare* write-rights must be granted.

The bi-directional transfer of settings is restricted to lifecycle phases

1. of the phase type PLANNING or REVIEW and
2. with a cardinality constraint of 1 i.e. there may exist one or no such setting for one device.

Ensure that at least one phase fulfils these requirements, and there exists a setting of this phase.

#### 24.15.4.2 PowerFactory Client

The client operating system must allow connections to the server (network and firewall settings etc.).

Nothing has to be done in the *PowerFactory* configuration itself. The *TypRelay* in the Library must of course support *StationWare - PowerFactory* mapping.

### 24.15.5 Getting Started

The mapping between *PowerFactory* object attributes and calculation results with *StationWare* device settings or process attributes, or additional attributes of devices, is done via flexible DPL scripts. These scripts have access not only to data in *PowerFactory* objects themselves, but also to other related objects e.g a relay type object or relay sub-blocks.

To be able to transfer data from *PowerFactory* to *StationWare* and vice versa, suitable DPL scripts have to be created and placed in an appropriate location in the project library folder.

```

Project.IntPrj
+- Library.IntPrjfilder
  +- Equipment Type Library.IntPrjfilder
    +- Relay Type X.IntFolder
      | +- Relay Type X.TypRelay
      |   +- PsmsExport.ComDpl
      |   +- PamsImport.ComDpl
      |
  +- Operational Library.IntPrjfilder
  +- Scripts.IntPrjfilder
  +- StationWare.IntPrjfilder
    +- Attributes.IntFolder
      |
      | +- ElmLne.IntFolder
      |   | +- PsmsExport.ComDpl
      |   | +- PsmsImport.ComDpl
      |   +- ElmTr2.IntFolder
      |     +- PsmsExport.ComDpl
      |     +- PsmsImport.ComDpl
      |
  +- Results.IntFolder
    |
    | +- arcflash.IntFolder
    |   | +- ElmTerm.IntFolder
    |   |   | +- PsmsExport.ComDpl
    |   |   +- ElmLne.IntFolder
    |   |     +- PsmsExport.ComDpl
    |   |
    |   +- shc.IntFolder
    |     +- ElmTerm.IntFolder
    |       | +- PsmsExport.ComDpl
    |     +- ElmLne.IntFolder
    |       +- PsmsExport.ComDpl
    |
  +- Settings.IntFolder
    |
    | +- ElmLne.IntFolder
    |   | +- PsmsExport.ComDpl
    |   | +- PsmsImport.ComDpl
    |   +- ElmTr2.IntFolder
    |     +- PsmsExport.ComDpl
    |     +- PsmsImport.ComDpl
  
```

Figure 24.15.5: Structure of the project library folder

The scripts for importing/exporting device settings should be located in the sub-folder “Settings” of the

*StationWare* folder inside the Project in *PowerFactory*.  
Project\Library\StationWare\Settings.

For importing/exporting additional attributes from *StationWare* DPL scripts should be located in the sub-folder “Attributes”. To be able to export results from *PowerFactory* to *StationWare* the corresponding script should be located in the sub-folder “Results”. None of these folders are by default in project library folder and must therefore be created.

**Important:** DPL scripts for import/export relay settings must be saved in the same folder as the relay model (as contents of a *TypRelay* object).

In difference to the data exchange of device settings, additional attributes and results, the DPL import/export scripts for relay settings can refer to mapping tables which simplify the mapping of individual parameters and the implementation of dependencies between parameters. Therefore, there are two different ways of exchanging relay settings. Either the mapping of the parameters in the DPL script code or the parameter mapping in mapping tables. Depending on which variant is selected, the basic DPL scripts differ. More information about the different mapping possibilities can be found in the documentation for [Protection Devices Library](#).

#### 24.15.5.1 Import/Export of Relay Settings

This section is a simple walk-through and covers the most essential *StationWare* interface functionality.

By using a basic *PowerFactory* project and basic *StationWare* substation, it describes

1. how relays in *StationWare* and *PowerFactory* are created,
2. how these relays are linked,
3. how settings can be exported from *PowerFactory* to *StationWare*
4. how settings can be imported again into *PowerFactory*.

All (especially the more advanced) options and features are described in the section ‘Description of the Menu and Dialogs’ (see Section [24.15.6](#)).

#### Prepare substation in *StationWare*

We begin with the *StationWare* side. We create a substation and two relays within:

- Start the web browser,
- log on to the *StationWare* system,
- create a new substation titled *Getting Started*,
- create two relays named *Getting Started Relay 1* and *Getting Started Relay 2* in the *Getting Started* substation.

In the HTML interface the station detail page should look as shown in Figure [24.15.6](#).

- Go to the detail page of the *Getting Started Relay 1* (Figure [24.15.7](#)).

Since we have just created the device it has no settings, yet. Later it will contain a *PowerFactory* setting which reflects the relay state on the *PowerFactory* side.

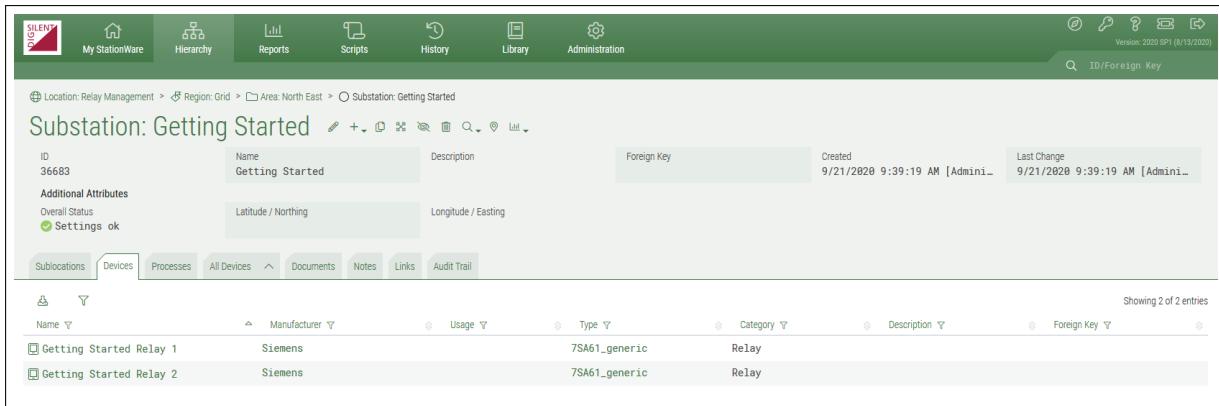


Figure 24.15.6: Substation

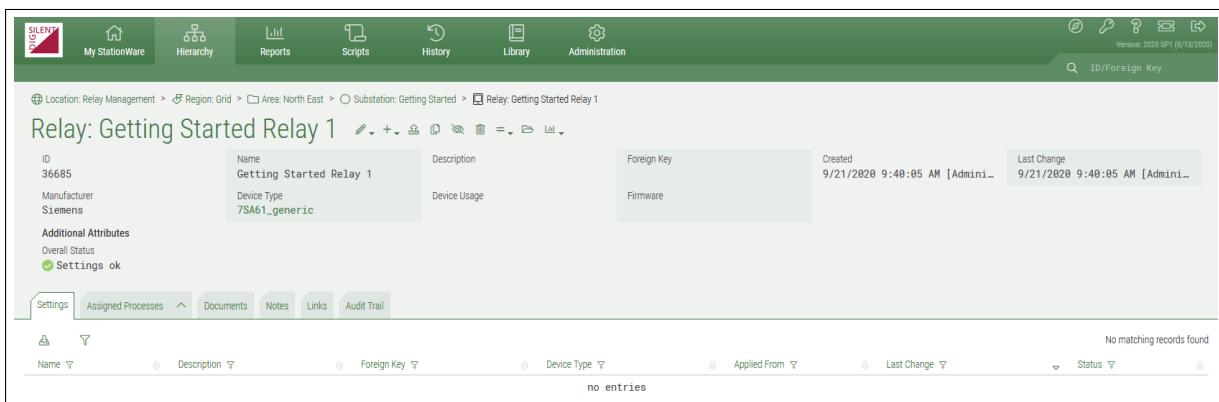


Figure 24.15.7: Device

### Prepare project in *PowerFactory*

Create a new *PowerFactory* project and create a simple grid within:

- Start *PowerFactory*,
- create a new project titled *GettingStarted*,
- draw a simple grid with two terminals (*ElmTerm*) connected by a line (*ElmLine*) as shown in Figure 24.15.8.

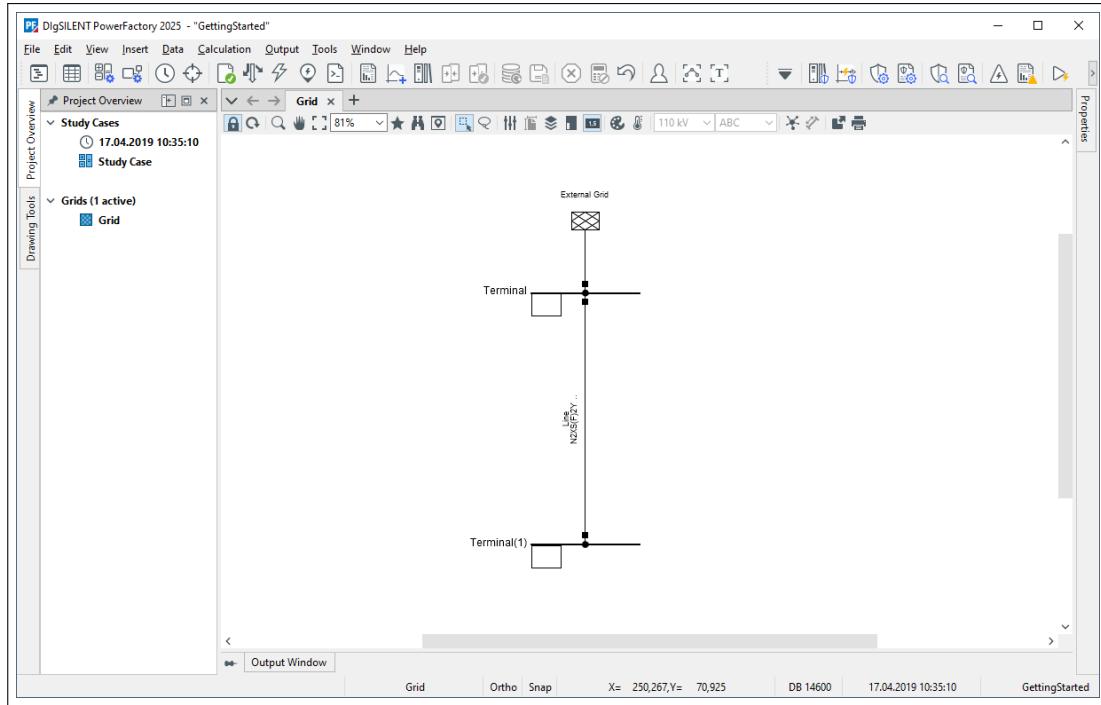


Figure 24.15.8: Grid

Now add a relay to the upper terminal:

- Right-click the cubicle quadrangle with the mouse. A context menu pops up.
- Select *Devices* → *Relay* → *New...* as shown in Figure 24.15.9.

A dialog pops up that allows you to specify the settings of the new relay (*ElmRelay*).

- Insert *Getting Started Relay 1* as Name,
- select an appropriate *Relay Type* which supports *StationWare* import/export (see Figure 24.15.10),
- press **OK**,
- in the same way add a relay *Getting Started Relay 2* to the second terminal.

*PowerFactory's object filter mechanism gives an overview over all devices inside the current project.*

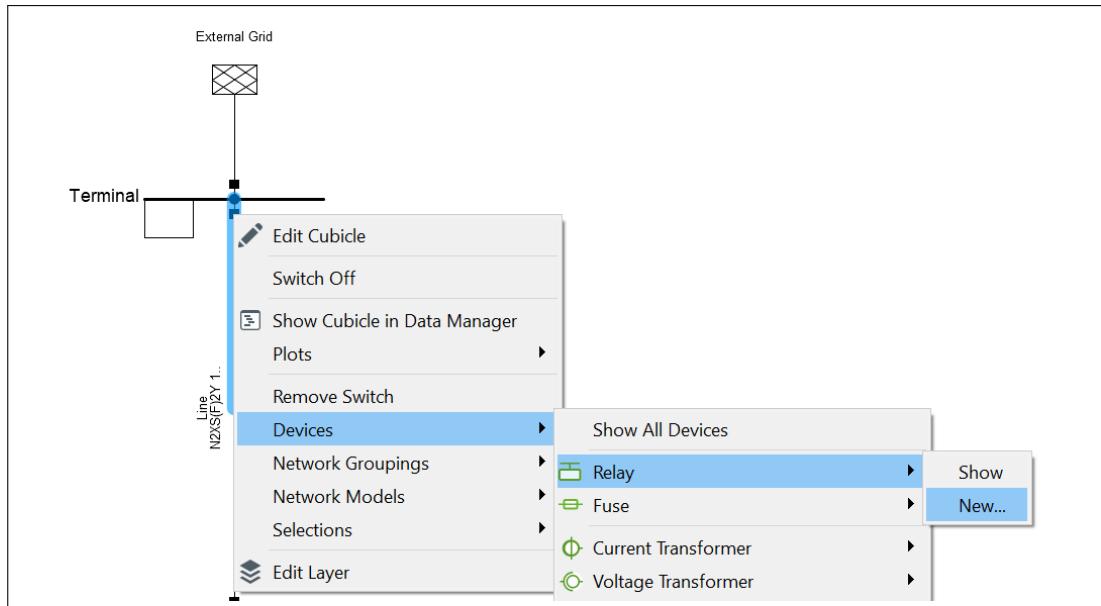


Figure 24.15.9: Cubicle context menu

- Press the icon (*Open Network Model Manager...*) in the toolbar and select the class (*ElmRelay*) to filter out all non-relay objects.

All calculation relevant relays (actually there only the two we created above) are displayed in a table (see Figure 24.15.11).

#### Link Relays and establish a Connection

Now the *PowerFactory* relays must get linked to the *StationWare* relays. To be able to make a connection:

- Ensure that the DPL Import/Export scripts are saved in the same folder as the relay model. If mapping tables are used, ensure that the path and the name of the mapping tables is set in the DPL Import/Export scripts.
- Mark both relay icons with the mouse.
- Press the right mouse button.

A context menu pops up as shown in Figure 24.15.12.

- Select the *StationWare* menu item,
- select the *Select Device ID* item.

A Log on to *StationWare* server dialog pops up. Since this is the first time *PowerFactory* connects to the *StationWare* server some connection settings must be entered.

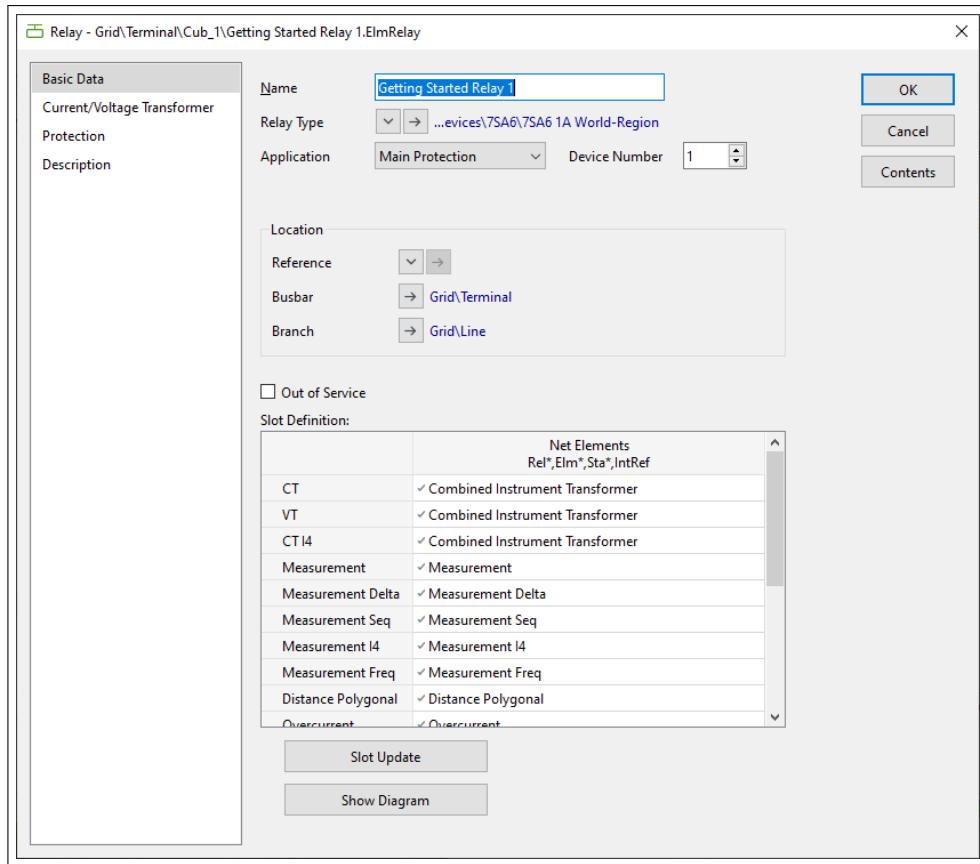


Figure 24.15.10: Relay dialog

- Enter the Server Endpoint URL of the *StationWare* server. The URL should have a format similar to  
`http(s)://192.168.1.53/pssmsws/PSMSService.asmx.`
- Enter *Username* and *Password* of a valid *StationWare* user account.

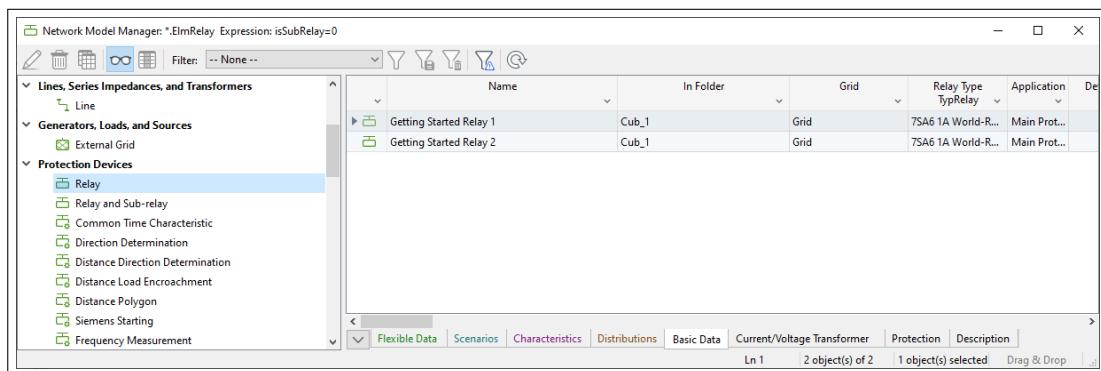


Figure 24.15.11: Relay display

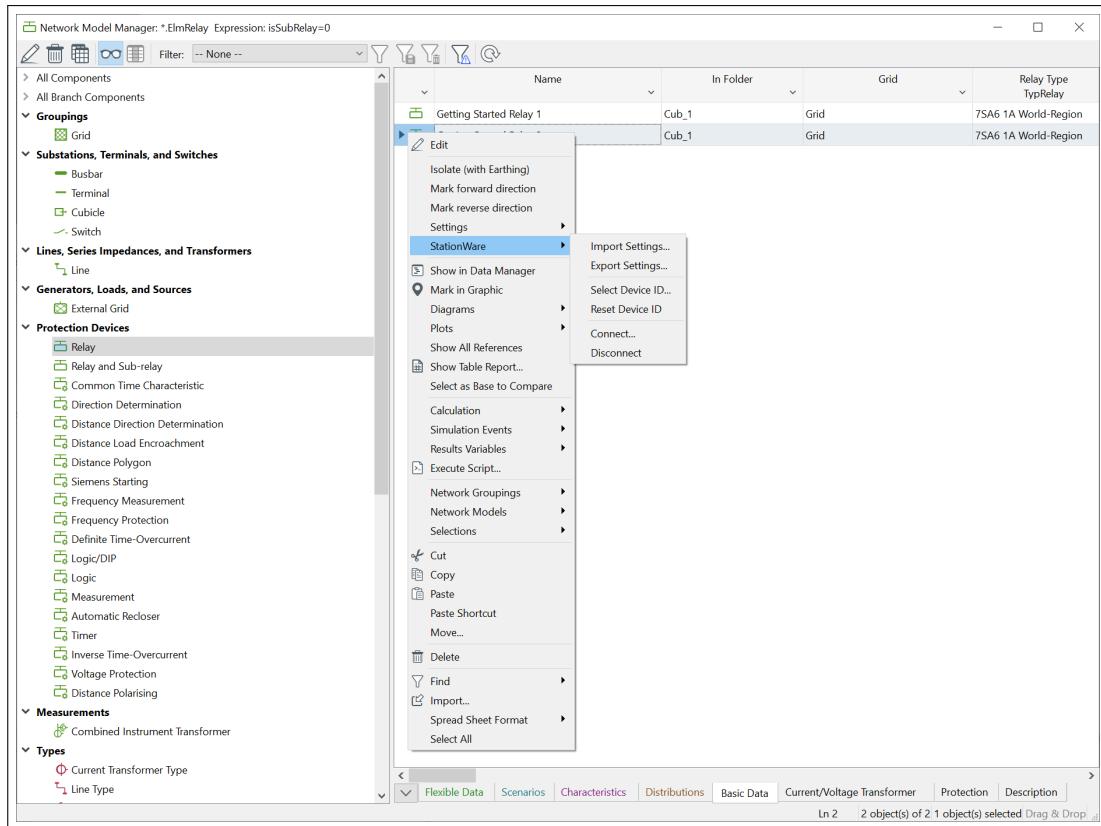


Figure 24.15.12: Device context menu

Figure 24.15.21 shows the dialog settings.

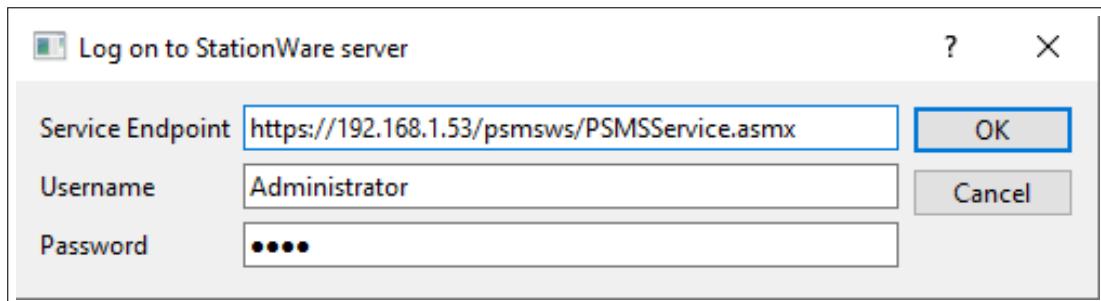


Figure 24.15.13: Log on dialog

- Press **OK**.

The connection procedure may take some seconds. If the server could be accessed and the user could be authenticated a success message is printed into the output window

```
Established connection to StationWare server
' https://192.168.1.53/pSMSws/PSMSService.asmx' (version 18.2.6981) as user
' Administrator'
```

Otherwise an error dialog pops up. Correct the connection settings until the connection is successfully created. The section 'Description of the Menu and Dialogs' (Section 24.15.6) explains the connection options in detail.

Having established a connection to the server, a browser dialog pops up which displays the location hierarchy as known from the *StationWare* HTML interface. The dialog is shown in Figure 24.15.14.

- Navigate to the *Getting Started* substation,
- select the *Getting Started Relay 1* device,
- press **OK**.

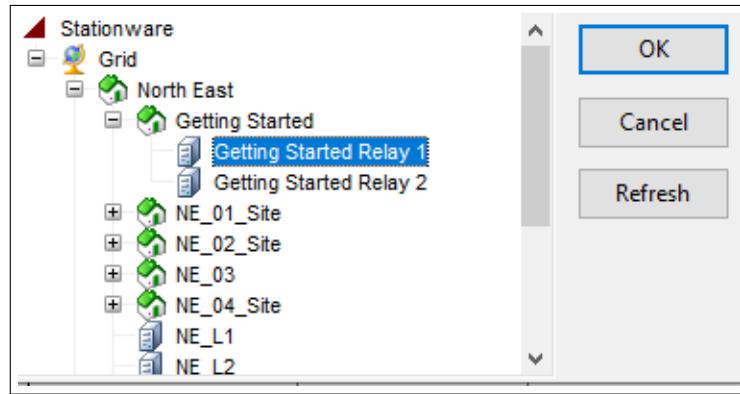


Figure 24.15.14: Browser dialog

Now the *PowerFactory* relay is “connected” to the *StationWare* device.

- In the same way select *Getting Started Relay 2* for the second *PowerFactory* relay.

### Export and Import Settings

Having linked *PowerFactory* to *StationWare* devices, the transfer between both systems can be started.

- Mark the relays with the mouse and right-click to get the relay context menu as shown in Figure 24.15.12.
- Select the *Export Settings...* in the *StationWare* menu entry.

A *ComStationware* dialog is shown which allows to specify the export options. See section ‘Export and Import Settings’ in the Chapter 24.15.6 for all export options.

- Select *PowerFactory* as lifecycle phase,
- press **Execute**.

After a few seconds the relay settings are transferred to the server, and the output window contains the message

```
Exported 2 of 2 device settings successfully
```

The result can now be observed in the *StationWare* HTML interface.

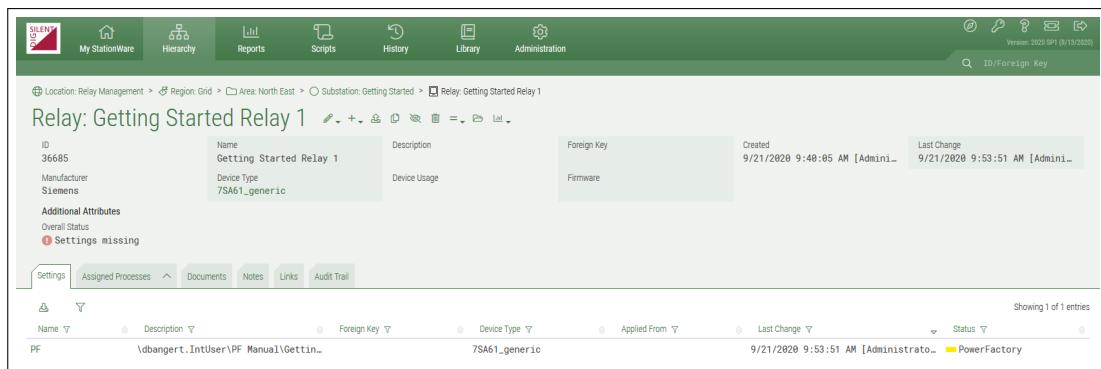


Figure 24.15.15: Device detail page

- Navigate to the relay detail view of the *Getting Started Relay 1* relay (see Fig. 24.15.15)

Observe the new created PF setting. The phase of this setting is *PowerFactory*.

- Switch to the settings detail page of the new *PowerFactory* setting (see Fig. 24.15.16).

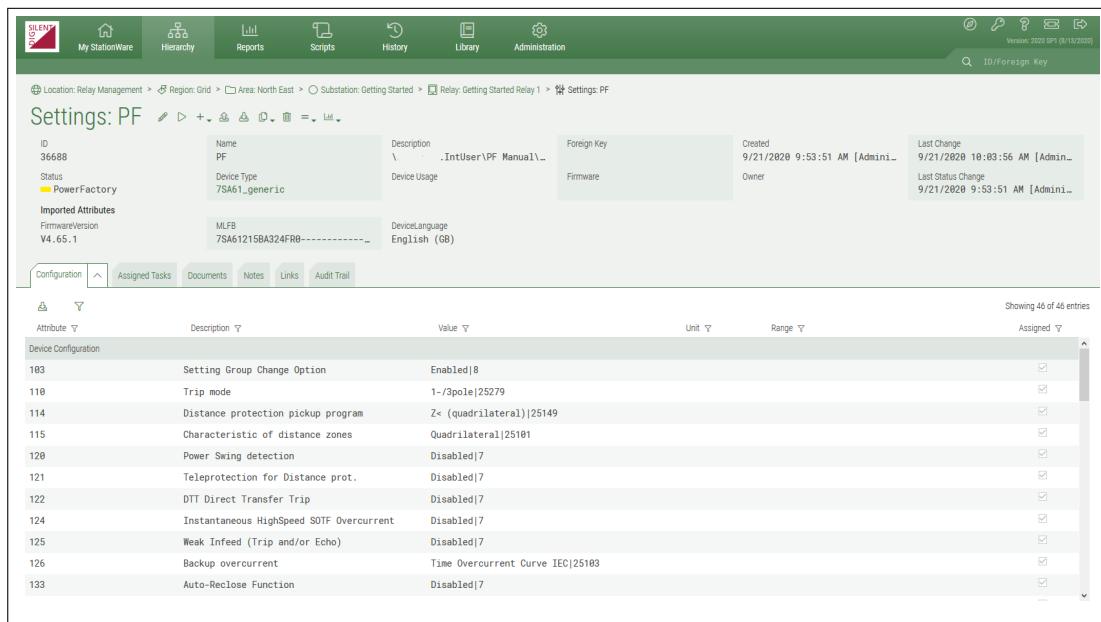


Figure 24.15.16: Setting detail page

The setting values should correspond to the relay state in *PowerFactory*. In the same way the *Getting Started Relay 2* relay has a new PF setting.

Now try the opposite direction and import a setting from *StationWare* into *PowerFactory*.

- Modify the PF settings in *StationWare* by entering some other values.
- In *PowerFactory* mark the relays with the mouse and right-click to get the relay context menu as shown in Figure 24.15.12.
- Select the *Import Settings...* in the *StationWare* menu entry.

Again the *ComStationware* dialog pops up as known from the export.

- Leave the default settings,
- press **Execute**.

Again the result of the settings transfer is reflected in the output window:

Imported 2 of 2 device settings successfully

- find *ElmRelay* object parameters changed according to the changes on the *StationWare* side

All import options are described in detail in the Section [24.15.6: Import/Export Options](#).

#### 24.15.5.2 Import/Export of the Additional Attributes

Additional attributes represent additional information which users may find useful for a location, device or settings within a device. These are not directly part of a settings record but are user-defined. For example, a common additional attribute that is useful for a feeder or substation location is the nominal voltage level in kV. Primary elements such as lines do not possess settings but instead parameters. Parameters such as length or impedance are then presented by the use of additional attributes.

The screenshot shows the SILENT PowerFactory interface with the following details:

- Header:** My StationWare, Hierarchy, Reports, Scripts, History, Library, Administration, Version: 2020 SP1 (8/13/2020), ID/Foreign Key.
- Breadcrumbs:** Location: Relay Management > Region: Grid > Area: North East > Line: NE\_L1.
- Line Details:** NE\_L1, ID: 36261, Manufacturer: \*, Device Type: Line.
- Additional Attributes Table:**

Overall Status	Terminal i	Terminal j	Length	Laying Ground	Substation i
Settings ok	BB2	BB1	1		NE_02
Substation j	NE_03				
- Tab Bar:** Type Data (highlighted with a red box), Settings, Assigned Processes, Documents, Notes, Links, Audit Trail.
- Table View:** Shows 13 entries of type data for the line NE\_L1, including columns for Name, Value, and Unit.

Figure 24.15.17: Additional attributes on the 'Device' page

The following information for additional attributes can be imported/exported:

- Name of the attribute
- Description
- Unit
- Value (Bool, String, Integer, Real, Enumeration, Data Time)
- Type (Attribute, Propagate, Overall Status, Revision Number)

Import/export of additional attributes also requires that the DPL script be saved in the appropriate place (see Section [24.15.5](#)). All actions are similar to those described for settings (see Section [24.15.5.1](#)).

#### 24.15.5.3 Export of the calculation results

Calculation results data exchange is only possible in one direction: from *PowerFactory* to *StationWare*. It is important to know that *PowerFactory* stores calculation results in attributes of temporary so-called "calculation objects".

This data will be exchanged between “calculation objects” and *StationWare* process objects.

### Preparation of *StationWare* for importing result data

Inside a *StationWare* project, define the process lifecycle, category and type. This process object should be configured to be capable of result data storage and presentation (e.g. “ArcFlashLabel Type” see Figure 24.15.18).

**Important:** Process lifecycle must posses a phase named “*PowerFactory*” of type “Planning”.

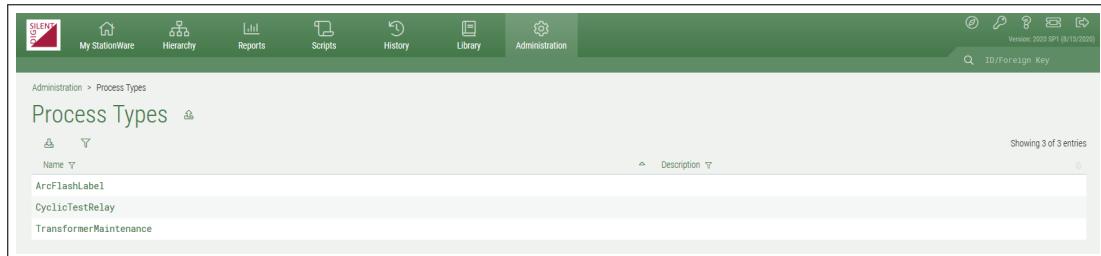


Figure 24.15.18: Process types page in *StationWare*

After being defined, the process should be created and have a device assigned to it.

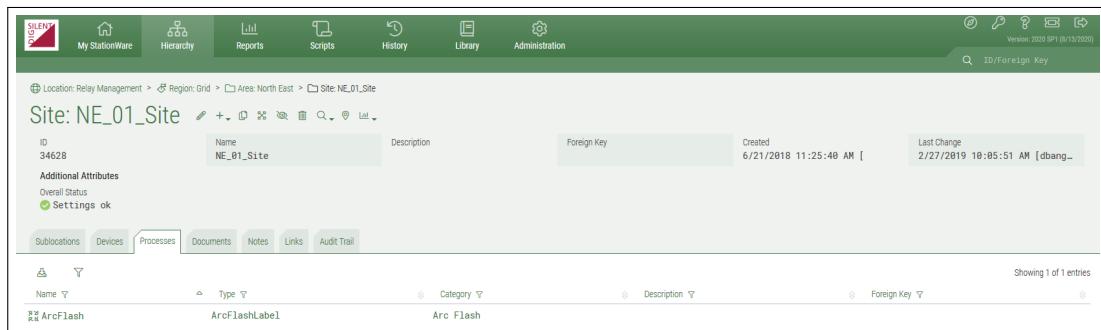


Figure 24.15.19: Location page where process is created

### Preparing *PowerFactory* for export of result data

In *PowerFactory* it is important to have the DPL transfer script created and saved in a proper place inside the project library folder (see Section 24.15.5). It is necessary to use separate scripts for each calculation type and for each *PowerFactory* object class.

### Connection of *PowerFactory* and *StationWare*

Refer to Section 24.15.5.1.

### Export of results

Refer to similar Section 24.15.5.1.

## 24.15.6 Description of the Menu and Dialogs

This section describes all options and features concerning the *StationWare* interface.

## The Device Context Menu

Almost all functionality can be accessed by the device context menu. Mark one or more objects which supports the *StationWare* transfer e.g. *ElmRelay*

- in the object filter (Figure 24.15.12)
- in the Data Manager as shown in Figure 24.15.20.

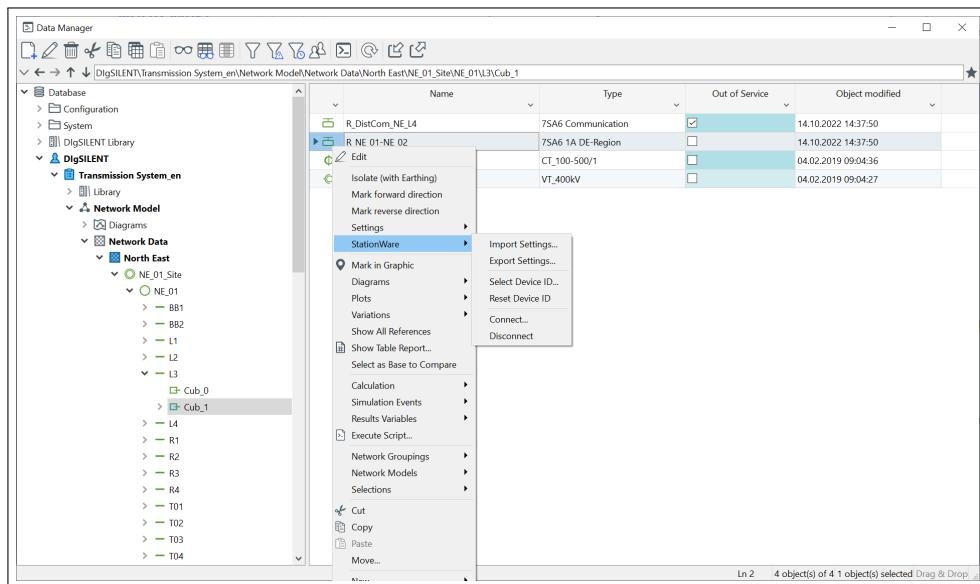


Figure 24.15.20: Device context menu

The *StationWare* submenu contains the entries as follows:

**Import X...** opens the *ComStationware* dialog and sets the device selection according to the above selected device objects. The *ComStationware* dialog settings are explained in detail in Section 24.15.6: The *ComStationware* Object.

**Export X...** does the same for the export direction.

**Select Device ID...** starts the Browser dialog (Figure 24.15.24) to link this device to a *StationWare* device. The dialog is subject of Section 24.15.6 : The Browser dialog.

**Reset Device ID** resets the device ID.

**Connect...** terminates the current *StationWare* session if it's already existing. Shows a Log On dialog. The connection settings are covered by Section 24.15.6. This may be useful when you are using several *StationWare* accounts and want to switch between them.

**Disconnect** terminates the *StationWare* session

## Connection

Similar to the HTML interface the *StationWare* interface in *PowerFactory* is session - oriented: when a user logs on to the system by specifying a valid *StationWare* account (username and password) a new session is created. Only inside such a session *StationWare* can be used. The account privileges restrict the application functionality e.g. an administrator account is more powerful than a usual user account.

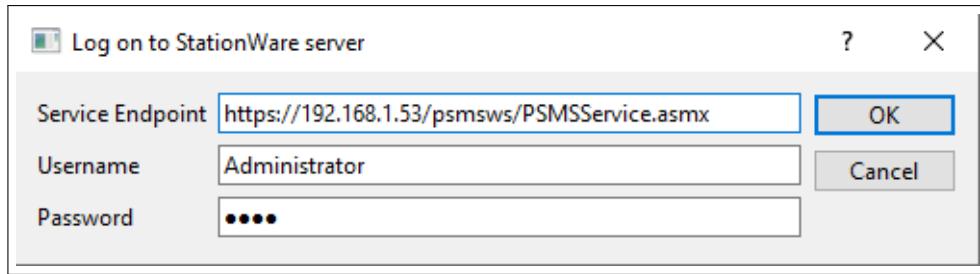


Figure 24.15.21: Log on dialog

Working with *PowerFactory* for the first time, the *StationWare* server is required, and the log on dialog is as shown in Figure 24.15.21.

The *StationWare* connection options are stored in the user settings (Figure 24.15.22). After each successful logon the user settings are updated.

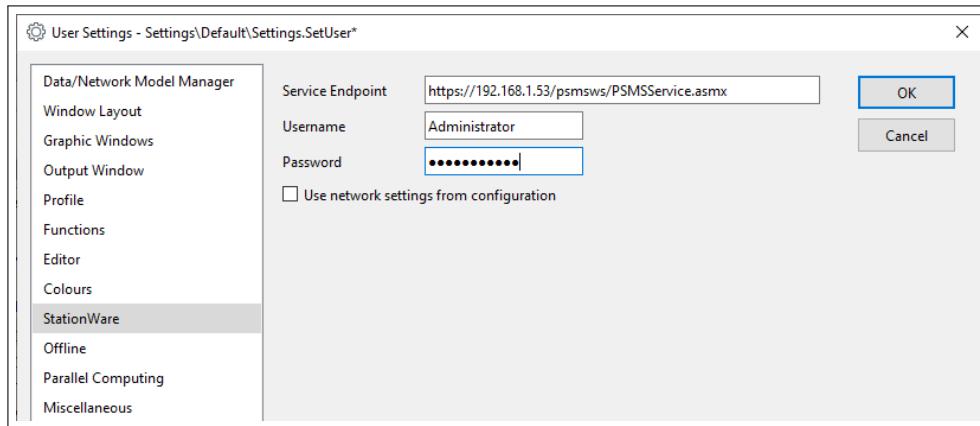


Figure 24.15.22: Log on dialog

As mentioned in the Architecture section (Section 24.15.2) *StationWare* is a client-server application. The *StationWare* server component is located on a server machine in the internet. The client component is the *PowerFactory* application which is running on a client machine.

The technology *PowerFactory* and *StationWare* use to communicate is called web services and is standardised like many other internet technologies (HTML, HTTP(S)). The server computer (or more exactly the *StationWare* service application on the server computer) has a 'name' by which it can be accessed. This 'name' is called service endpoint and resembles a web page URL:

`https://the.server.name/psmsws/PSMSService.asmx`

or

`https://192.168.1.53/psmsws/PSMSService.asmx`

http(s) denotes the protocol, the.server.name is the computer name (or DNS) of the server computer and psmsws/PSMSService.asmx is the name of the *StationWare* application.

The connection options are as follows:

**Service Endpoint** The Service Endpoint denotes the *StationWare* server 'name' as described above

**Username/Password** Username and Password have to be valid user account in *StationWare*. A *StationWare* user account has nothing to do with the *PowerFactory* user account.

The very same *StationWare* account can be used by two different *PowerFactory* users. The privileges of the *StationWare* account actually restrict the functionality. For device import the user requires read-access rights. For exporting additionally write-access rights are required.

### The Browser Dialog

As mentioned in the Concept description (see Section 24.15.3.2: Device) the *StationWare* device ID is stored as Foreign Key in the e.g. *ElmRelay* object dialog (Description page) as shown in Figure 24.15.23.

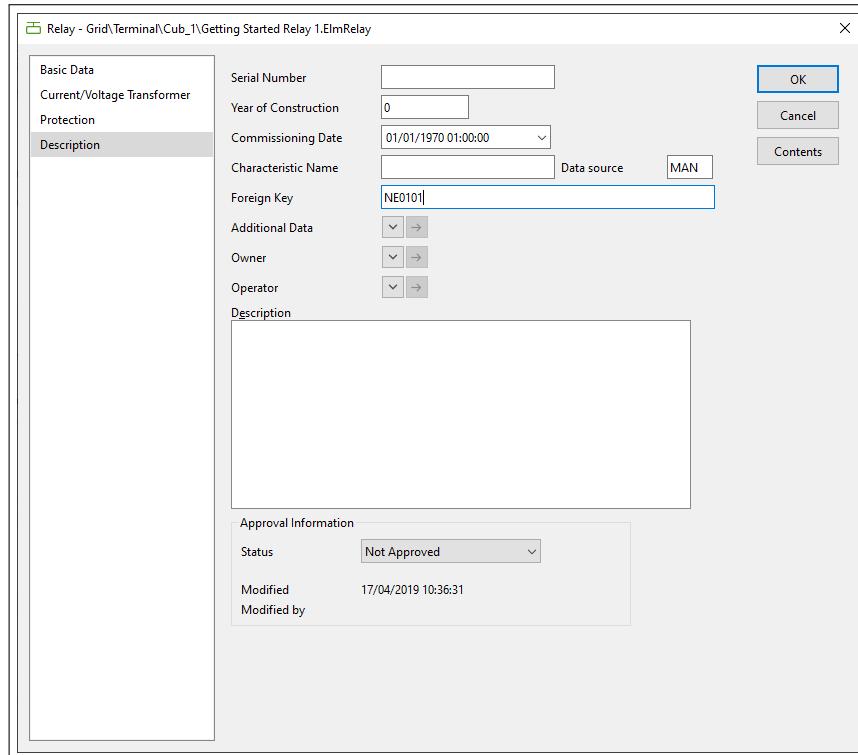


Figure 24.15.23: *ElmRelay* dialog

A more convenient way is to use the Browser dialog shown in Figure 24.15.24. The dialog allows to browse through the *StationWare* location hierarchy and select a device. The hierarchy data is cached to minimise network accesses. Due this caching it's possible that there may exist newly created locations or devices which are not displayed in the browser dialog. The Refresh button empties the cache and enforces *PowerFactory* to re-fetch the correct data from the server.

### The ComStationware Object

In *PowerFactory* almost everything is an object: relays are *ElmRelay* objects, users are *IntUser* objects, and grids are *ElmNet* objects, ...

What may be on the first sight confusing is the fact that actions are objects as well: for a short-circuit calculation a *ComShc* object is created. The calculation can be performed with several options e.g. 3-Phase, single phase, or 3 Phase to Neutral.

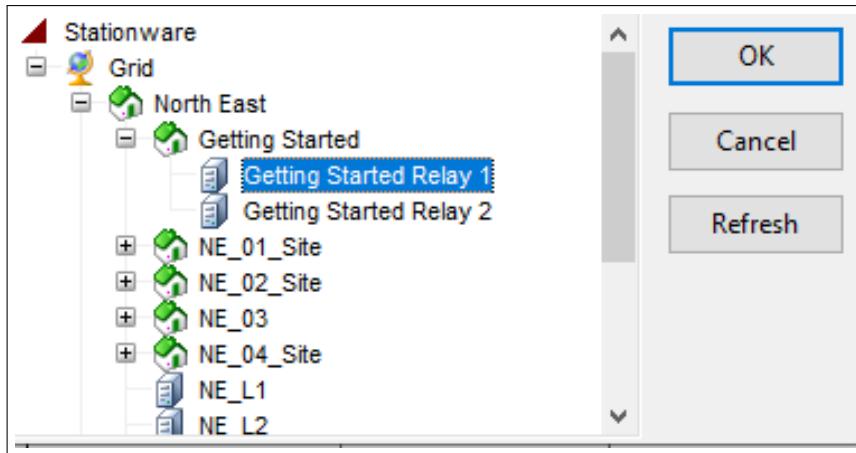


Figure 24.15.24: Browser dialog

You can even specify the fault location. All these calculation options are stored in the *ComShc* object. Every action object has an **Execute** button which starts the action. In fact there is a large number of parameterised actions like load flow calculation (*ComLdf*), simulation (*ComSim*), there is even a *ComExit* object that shuts down *PowerFactory*. All objects which can 'do' something have the Com prefix.

Since the *StationWare* interface is actually 'doing' something (it does import data, it does export data) it is implemented as a *ComStationware* object.

The *ComStationware* object is used both for the import and the export. It is located in the project's study case according to *PowerFactory* convention.

By default the study case of a new project contains no *ComStationware* object. It is automatically created when it is first needed, as well as the *ComShc* object is instantiated at the time when the first short-circuit calculation is performed.

### Import/Export Options

The *ComStationware* dialog provides import/export options as follows:

**Transfer Mode** select Import/Export from *StationWare* as Transfer Mode

**Transfer Data** select Import/Export Data from *StationWare* (Attributes, Settings, Results of last calculation)

**Check only Plausibility** if the Check only Plausibility flag is enabled the import is only simulated but not really executed.

**Lifecycle Phase/Time stamp** A list of available lifecycle phases is shown.

- *PowerFactory* selects the current setting with *PowerFactory* phase as source setting.
- If Applied is selected the current Applied setting is transferred. If additionally a Timestamp value is entered the setting that was applied at this time is transferred which may either be Applied or Historic.

The Timestamp format is in ISO format: e.g. 2005-02-28 22:27:16

The time part may be omitted. Then 00:00:00 AM is assumed.

**All Devices** If All Devices is enabled, all calculation-relevant devices are imported/exported.

**Device Selection** Unless All Devices is enabled, the Device Selection provides a more subtle

way to specify which devices are to be transferred.

The Device Selection is automatically set if the Device Context Menu mechanism (Section 24.15.6: The Device Context Menu) is used.

**All Settings Groups/Group Index** This parameter specifies how multiple settings groups (MSG) are handled.

The import/export transfer is started by pressing **Execute**.

## 24.16 API (Application Programming Interface)

For a detailed description on the API, a reference document is available via the main menu *Help → Additional Packages→ Programming Interface (API)*

## **Part IV**

# **Power System Analysis Functions**

# Chapter 25

## Load Flow Analysis

### 25.1 Introduction

Whenever evaluating the operation and control of power systems, the electrical engineer is typically encountered with questions such as:

- Are the voltages of every busbar in the power system acceptable?
- What is the loading of the different equipment in the power system? (transformers, transmission lines, generators, etc.)
- How can I achieve the best operation of the power system?
- Does the power system have a weakness (or weaknesses)? If so, where are they located and how can I countermeasure them?

Although we may consider that the above questioning would arise only when analysing the behaviour of “existing” power systems; the same interrogations can be formulated when the task relates to the analysis of “future” systems or “expansion stages” of an already existing power system; such as evaluating the impact of commissioning a transmission line or a power plant, or the impact of refurbishment or decommissioning of equipment (for example shutting down a power plant because it has reached its life expectancy).

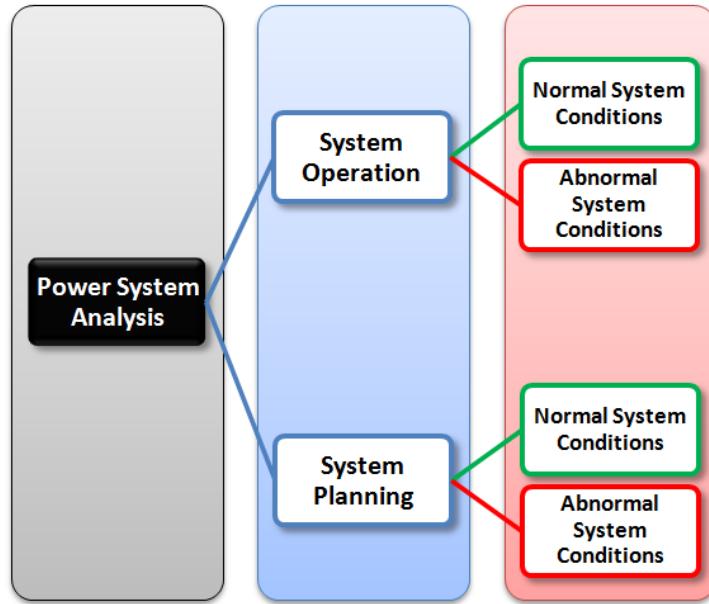


Figure 25.1.1: Power System Analysis: System Operation and System Planning

Taking into account these two aspects: 1) Present operation and 2) Future operation, is how power should be analysed. From one side, an operation or control engineer requires relevant information to be available to him almost immediately, meaning he must be able to obtain somehow the behaviour of the power system under different configurations that can occur (for example by opening or closing breakers in a substation); on the other side, a planning engineer requires obtaining the behaviour of the system reflecting reinforcements that have not yet been built while considering the corresponding yearly and/or monthly load increase. Regardless of the perspective, the engineer must be able to determine beforehand the behaviour of the power system in order to establish, for example, the most suitable operation configuration or to detect possible weakness and suggest solutions and alternatives. Figures 25.1.2 and 25.1.3 illustrate the system operation and planning aspects.

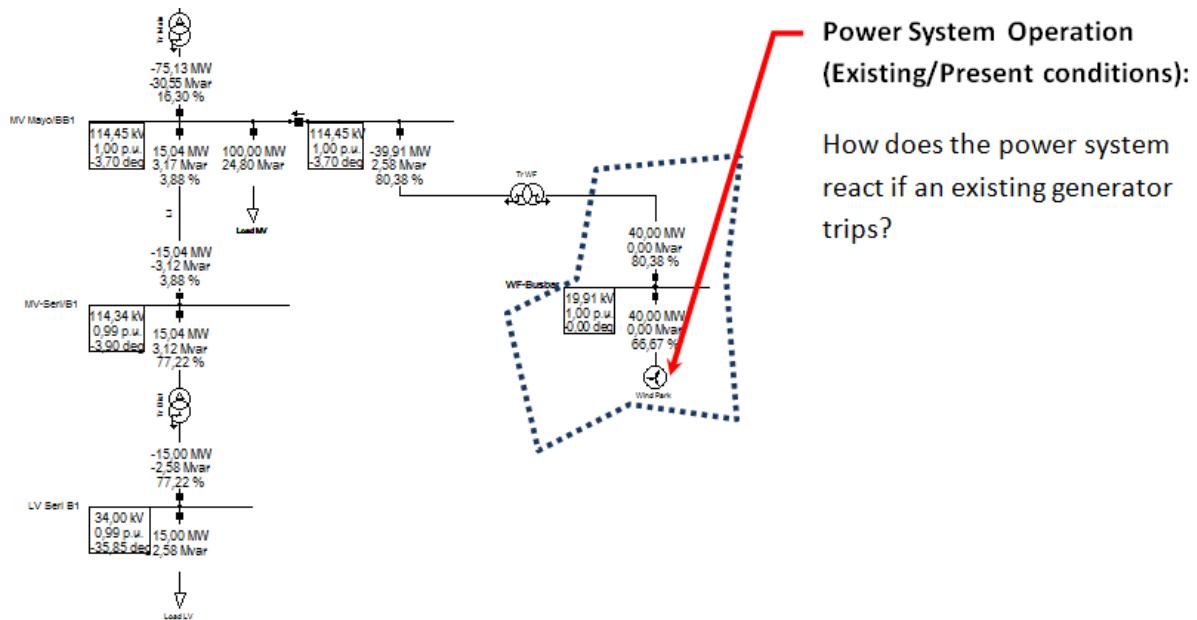


Figure 25.1.2: Power system operation example

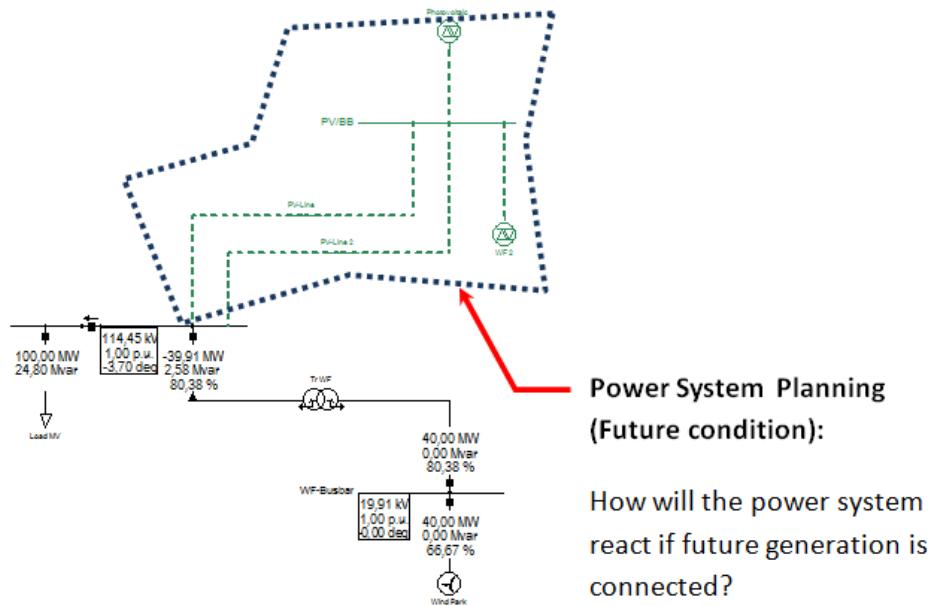


Figure 25.1.3: Power system planning example

## 25.2 Technical Background

Load flow calculations are used to analyse power systems under steady-state non-faulted (short-circuit-free) conditions. Where **steady-state** is defined as a condition in which all the variables and parameters are assumed to be constant during the period of observation. We can think of this as “taking a picture” of the power system at a given point in time. To achieve a better understanding let us refer to Figure 25.2.1. Here a 24 hour load demand profile is depicted. The user can imagine this varying demand to be the demand of a specific area or region, or the demand of a whole network. In this particular case the load is seen as increasing from early in the morning until it reaches its maximum at around 18:00 hrs. After this point in time, the total load then begins to decrease. A load flow calculation is stated to be a **steady-state** analysis because it reflects the system conditions for a certain point in time, such as for instance at 18:00 hrs (maximum demand). As an example, if we require determining the behaviour of the system for every hour of the day, then 24 load flows need to be performed; if the behaviour for every second is required then the number of load flow calculations needed would amount to 86 400. In *PowerFactory*, the active power (and/or reactive power) of the loads can be set with a *Characteristic* so they follow a certain profile (daily, weekly, monthly, etc.). By doing so, the active power will change automatically according to the date and time specified. For more information refer to Chapter 18 (Parameter Characteristics, Load States, and Tariffs).

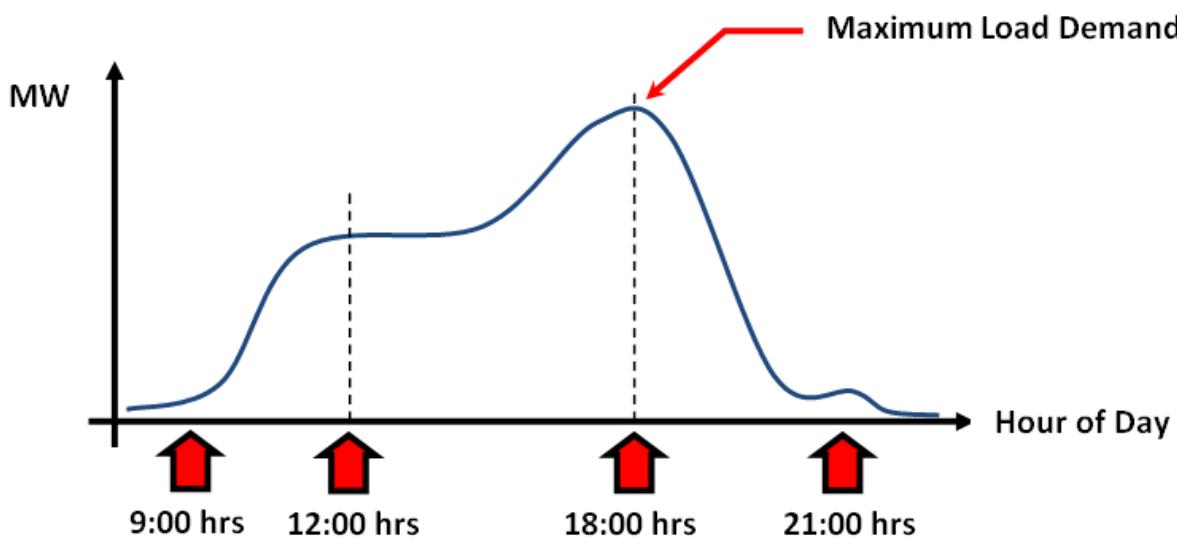


Figure 25.2.1: Example of a Load Demand Curve

A load flow calculation will determine the active and reactive power flows for all branches, and the voltage magnitude and phase for all nodes.

The main areas for the application of load flow calculations can be divided in normal and abnormal (Contingency) system conditions as follows:

### Normal System Conditions

- Calculation of branch loadings, system losses and voltage profiles.
- Optimisation tasks, such as minimising system losses, minimising generation costs, open tie optimisation in distributed networks, etc.
- Calculation of steady-state initial conditions for stability simulations or short-circuit calculations using the complete superposition method.

### Abnormal System Conditions

- Calculation of branch loadings, system losses and voltage profiles.
- Contingency analysis, network security assessment.
- Optimisation tasks, such as minimising system losses, minimising generation costs, open tie optimisation in distributed networks, etc.
- Verification of system conditions during reliability calculations.
- Automatic determination of optimal system resupplying strategies.
- Optimisation of load-shedding.
- Calculation of steady-state initial conditions for stability simulations or short-circuit calculations using the complete superposition method (special cases).

Regarding the above definitions of "normal" and "abnormal" system conditions, a distinction should be made in terms of the manner simulations should be performed:

**Simulation of normal operating conditions:** Here, the generators dispatch as well as the loads are known, and it is therefore sufficient for the load flow calculation to represent these generators dispatch and to provide the active and reactive power of all loads. The results of the load flow

calculation should represent a system condition in which none of the branch or generator limits are exceeded.

**Simulation of abnormal operating conditions:** Here a higher degree of accuracy from the models is needed. It can no longer be assumed that the entire system is operating within limits. The models must be able to correctly simulate conditions which deviate from the normal operating point. Hence the reactive power limits of generators or the voltage dependency of loads must be modelled. Additionally, in many applications, the active power balance cannot be established with a single slack bus (or machine). Instead, a more realistic representation of the active and reactive power control mechanisms have to be considered to determine the correct sharing of the active and reactive power generation.

Besides the considerations regarding abnormal conditions presented above, the assumption of balanced systems may be inappropriate for certain distribution networks. State of the art computational tools for power systems analysis must be therefore able to represent unbalanced networks for load flow calculations as well.

The calculation methods and the options provided by *PowerFactory*'s load flow analysis function allow the accurate representation of any combination of meshed 1-, 2-, and 3-phase AC and/or DC systems. The load flow tool accurately represents unbalanced loads, generation, grids with variable neutral potentials, HVDC systems, DC loads, adjustable speed drives, SVSs, and FACTS devices, etc., for all AC and DC voltage levels. With a more realistic representation of the active and reactive power balance mechanisms, the traditional requirement of a slack generator is left optional to the user.

The most considerable effect of the resistance of transmission lines and cables is the generation of losses. The conductor resistance will at the same time depend on the conductor operating temperature, which is practically linear over the normal range of operation. In order to carry out such type of analysis, *PowerFactory* offers a *Temperature Dependency* option, so that the conductor resistance is corrected according to the specified temperature value.

For very fast and reliable analysis of complex transmission networks, where only the flow of active power through the branches is considered, *PowerFactory* offers an additional load flow method, namely "DC load flow (linear)", which determines the active power flows and the voltage angles within the network.

The following sections introduce the calculation methods and the options provided with *PowerFactory*'s load flow tool. This information is a guide to the configuration of the *PowerFactory* load flow analysis command . Additional information about special options are given in [25.2](#). Further technical details related to the models (Network Components) implemented in *PowerFactory* for load flow calculations are provided in the [Technical References Document](#).

### 25.2.1 Network Representation and Calculation Methods

A load flow calculation determines the voltage magnitude ( $V$ ) and the voltage angle ( $\vartheta$ ) of the nodes, as well as the active ( $P$ ) and reactive ( $Q$ ) power flow on branches. Usually, the network nodes are represented by specifying two of these four quantities. Depending on the quantities specified, nodes can be classified as:

- **PV nodes:** here the active power and voltage magnitude are specified. This type of node is used to represent generators and synchronous condensers whose active power and voltage magnitude are controlled (synchronous condensers  $P=0$ ). In order to consider equipment limits under abnormal conditions (as mentioned in the previous section), reactive power limits for the corresponding network components are also used as input information.
- **PQ nodes:** here the active and reactive power are specified. This type of node is used to represent loads and machines with fixed values. Loads can also be set to change (from their original  $P_0$  and  $Q_0$  values at nominal voltage) as a function of the voltage of the node to which the load itself is connected. Elements specified as PQ (for example synchronous machines, static generator's, PWM converters or SVS's) can be "forced" by the algorithm so that the P and Q resulting from the load flow are always within limits.

- **Slack node:** here the voltage magnitude and angle are fixed. In traditional load flow calculations the slack node (associated with a synchronous generator or an external grid) carries out the balancing of power in the system.
- **Device nodes:** special nodes used to represent devices such as HVDC converters, SVSs, etc., with specific control conditions (for example the control of active power flow at a certain MW threshold in a HVDC converter, or the control of the voltage of a busbar by an SVS).

---

**Note:** In traditional load flow calculations, asynchronous machines are represented by PQ nodes, assuming that the machine operates at a certain power factor, independent of the busbar voltage. Besides this traditional representation, *PowerFactory* offers a more accurate “slip iteration” (AS) representation based on the model equivalent circuit diagrams. For further information refer to the [Technical References Document](#)

---

In contrast to other power system calculation programs, *PowerFactory* does not directly define the node characteristic of each busbar. Instead, more realistic control conditions for the network elements connected to these nodes are defined (see the *Load Flow* page of each element's dialog). For example, synchronous machines are modelled by defining one of the following control characteristics:

- Controlled power factor ( $\cos(\varphi)$ ), constant active and reactive power (**PQ**);
- Constant voltage, constant active power (**PV**) on the connected bus;
- Secondary (frequency) controller (**slack**, **SL**).

It is also important to note that in *PowerFactory* the active and reactive power balance of the analysed networks is not only possible through a slack generator (or external grid). The load flow calculation tool allows the definition of more realistic mechanisms to control both active and reactive power. For further information refer to Section [25.4.1](#).

## Phase Technology

*PowerFactory* offers the possibility to model single-, bi- and three-phase AC networks with and without the neutral conductor and DC networks. The system type (AC, DC or AC/BI) and the number of phases are defined in the nodes.

- **ABC** corresponds to a three phase system with a phase shift of  $120^\circ$  between the phases.
- **BI** represents a dual phase system with a  $180^\circ$  phase shift between both phases.
- **2PH** is used if only two of the three phases of an ABC-system are connected.
- **1PH** is the choice if only a single phase has to be modelled.
- **xxx-N** considers an additional neutral conductor for the xxx phase technology.

Edge elements with 1, 2 or 3 phases can be connected to nodes with a corresponding phase technology. For e.g. lines or transformers, the phase technology is set in their types, whereas e.g. the balanced or unbalanced power demand of loads can be configured directly in the element.

This leads to a high flexibility in modelling any desired network with varying phase technologies in the same or different projects and calculating the resulting power flows.

## AC Load Flow Method

In *PowerFactory* the nodal equations used to represent the analysed networks are implemented using two different formulations:

- Newton-Raphson (Current Equations).

- Newton-Raphson (Power Equations, classical).

In both formulations, the resulting non-linear equation systems must be solved by an iterative method. *PowerFactory* uses the Newton-Raphson method as its non-linear equation solver. The selection of the method used to formulate the nodal equations is user-defined, and should be selected based on the type of network to be calculated. For large transmission systems, especially when heavily loaded, the standard Newton-Raphson algorithm using the “Power Equations” formulation usually converges best. Distribution systems, especially unbalanced distribution systems, usually converge better using the “Current Equations” formulation.

In addition to the Newton-Raphson iterations, which solve the network nodal equations, *PowerFactory* applies an outer loop when the control characteristic of automatic transformer tap changers and/or switchable shunts is considered. Once the Newton-Raphson iterations converge to a solution within the defined tolerance (without considering the setpoint values of load flow quantities defined in the control characteristic of the tap changers/switchable shunts), the outer loop is applied in order to reach these target values. The actions taken by the outer iterative loop are:

- Increasing/decreasing discrete taps;
- Increasing/decreasing switchable shunts; and
- Limiting/releasing synchronous machines to/from max/min reactive power limits.

Once the above-listed actions are taken, a new Newton-Raphson load flow iteration takes place in order to determine the new network operating point.

In the classical load flow calculation approach, the unbalance between phases are neglected. For the analysis of transmission networks this assumption is generally admissible. In distribution networks this assumption may be inappropriate depending on the characteristics of the network. *PowerFactory* allows the calculation of both balanced (*AC Load Flow, balanced positive sequence*) and unbalanced (*AC Load Flow Unbalanced, 3-phase (ABC)*) load flows according to the descriptions above.

### DC Load Flow Method

In addition to the “AC” load flow calculations presented in this section, *PowerFactory* offers a so-called “DC” load flow calculation method. The DC load flow should not be interpreted as a method to be used in case of DC systems given that it basically applies to AC systems.

Some occasions we may require performing fast analysis in complex transmission networks where only a reasonable approximation of the active power flow of the system is needed. For such situations the DC load flow can be used. Other applications of the DC load flow method include situations where the AC load flow has trouble converging (see Section [25.6: Troubleshooting Load Flow Calculation Problems](#)).

In this particular method, the non-linear system resulting from the nodal equations is simplified due to the dominant relation that exists between voltage angle and active power flow in high voltage networks. By doing so a set of linear equations is thereby obtained, where the voltage angles of the buses are directly related to the active power flow through the reactance of the individual components. The DC load flow does not require an iterative process and the calculation speed is therefore considerably increased. Only active power flow without losses is considered. Summarising, the DC load flow method has the following characteristics:

- The calculation requires the solving of a set of linear equations.
- No iterations required, therefore fast, and also no convergence problems.
- Approximate solution:
  - All node voltage magnitudes fixed at 1.0 per unit.
  - Only active power and voltage angles calculated.
  - Losses are neglected.

## 25.3 Executing Load Flow Calculations

A load flow calculation may be initiated by:

- Pressing the  icon on the main toolbar;
- Selecting the *Calculation → Load Flow ...* option from the main menu.

The following pages explain the load flow command options. Following this, some hints are given regarding what to do if your load flow cannot be solved.

The following pages describe the different load flow command (*ComLdf*) options. For a more detailed technical background regarding the options presented here, refer to Section 25.4. If the execution of the Load Flow Calculation leads to errors, refer to Section 25.6 for troubleshooting. The analysis of the results is however described in Section 27.5.

### 25.3.1 Basic Options

#### 25.3.1.1 Calculation Method

**AC Load Flow, balanced, positive sequence:** performs load flow calculations for a single-phase, positive sequence network representation, valid for balanced symmetrical networks. A balanced representation of unbalanced objects is used (for further details refer to Section 25.2.1).

**AC Load Flow, unbalanced, 3 Phase (ABC):** performs load flow calculations for a multi-phase network representation. It can be used for analysing unbalances of 3-phase systems, e.g. introduced by unbalanced loads or non-transposed lines, or for analysing all kinds of unbalanced system technologies, such as single-phase- or two-phase systems (with or without neutral return). For further details refer to Section 25.2.1. Unbalance specific results are described in Section 25.5.6.

**DC Load Flow (linear):** performs a DC load flow based on a set of linear equations, where the voltage angles of the buses are strongly related to the active power flow through the reactance of the individual components (for further details refer to Section 25.2.1).

#### 25.3.1.2 Active Power Regulation

**Automatic tap adjustment of phase shifters:** this option allows the automatic tapping of phase shifters (quadrature boosters) which have automatic tapping enabled. It will be effective both for DC and AC load flow calculations.

**Consider active power limits:** active power limits for models (as defined on the element's *Load Flow* tab) participating in active power balance, will be applied. If this option is disabled, the active power output limits may be violated, in which case a warning is issued. Note that it is possible to be selective about which machine models are considered in this respect; Section 25.3.3 describes how the models can be selected.

#### 25.3.1.3 Voltage and Reactive Power Regulation

This option is available only for AC load flow calculations.

**Automatic tap adjustment of transformers:** adjusts the taps of all transformers which have the option Automatic Tap Changing enabled on the Load Flow page of their element dialogs. The tap adjustment is carried out according to the control settings defined in the transformer element's dialog (for further information refer to the [Technical References Document](#)).

**Automatic tap adjustment of shunts:** adjusts the steps of all switchable shunts that have the option *Switchable* enabled on the Load Flow page of the shunt's element dialog (for further information refer to the [Technical References Document](#)).

**Consider reactive power limits:** considers the reactive power limits of models. If the load flow cannot be solved without exceeding the specified limits, a convergence error is generated. If this option is not enabled, *PowerFactory* will print a warning message if any of the specified limits are exceeded. Note that it is possible to be selective about which machine models are considered in this respect; Section [25.3.3](#) describes how the models can be selected.

#### 25.3.1.4 Temperature Dependency: Line/Cable Resistances

...at 20°C: the resistance of each line, conductor and cable will be according to the value stated in the Basic Data page of their corresponding type (at 20°C).

...at Maximum operating temperature: the resistance of each line, conductor and cable will be adjusted according to the equation (25.26) described in Section [25.4.4](#) and the Temperature Dependency option stated in its corresponding type (*TypLne*, *TypCon*, *TypCab*).

...at Operating temperature: when this option is selected, the specified individual operating temperature for each line and cable is used (specified on the *Load Flow* page of the element).

...at Temperature: when this option is selected a global operating temperature can be specified in the load flow command that is applied to all lines and cables.

#### 25.3.1.5 Load Options

**Consider Voltage Dependency of Loads:** the voltage dependency of loads with defined voltage dependency factors (*Load Flow* page of the general- and complex load types) will be considered.

**Feeder Load Scaling:** scales loads with the option *Adjusted by Feeder Load Scaling* enabled on the *Load Flow* page of their element dialog according to the *Scaling Factors* specified in the *Load Scaling* section of the feeder element. In this case, the *Scaling Factor* specified on the *Load Flow* page of load element dialog is disregarded. For the purpose of unbalanced load flows, if the option *Phasewise scaling* is selected on the *Load Flow* page of the feeder elements, different set points can be specified for each phase.

Details of the scaling process can be found in Section [25.4.3](#).

### 25.3.2 Active Power Control

#### 25.3.2.1 Active Power Control

As explained in Section [25.4.1](#), *PowerFactory*'s load flow calculation offers several options for maintaining power balance within the system under analysis. These options are:

**as Dispatched:** if this option is selected and no busbar is assigned to the *Reference Busbar* (*Reference Bus and Balancing* section of the *Active Power Control* tab), the total power balance is established by one reference generator/external grid ("slack"-generator). The slack generator can be directly defined by the user on the *Load Flow* page of the target element. The program automatically sets a slack if one has not been already defined by the user.

**according to Secondary Control:** power balance is established by all generators which are considered by a "Secondary Controller" as explained in Section [25.4.1](#). Active power contribution is according to the secondary controller participation factors.

**according to Primary Control:** power balance is established by all generators having a  $K_{pf}$ -setting defined (on the *Load Flow* page of a synchronous machine element dialog), as explained in Section 25.4.1. Active power contribution is according to the droop of every generator.

**according to Inertias:** power balance is established by all generators, and the contribution of each is according to the inertia (acceleration time constant) as explained in Section 25.4.1.

### 25.3.2.2 Balancing

If *as Dispatched* is selected in the *Active Power Control* section of the tab, further options regarding the power balancing method are available:

**by reference machine:** for each isolated area, the reference machine will balance the active power.

**by load at reference bus:** this option is valid only when the reference busbar has been defined. The load with highest active power injection at the reference bus will be selected as the slack (such as to balance the losses).

**by static generator at reference bus:** as in the case of *Balancing by Load*, this option is valid only when the reference busbar has been defined. The static generator with the highest rated apparent power at the reference bus will be selected as the slack (i.e. to balance the losses).

**Distributed slack by loads:** when this option is selected, only the loads which have the option Adjusted by Load Scaling enabled in the isolated area will contribute to the balancing. The distribution factor calculated for a load is determined by the following equation:

$$K_i = \frac{P_{ini,i}}{\sum_{j=1}^n P_{ini,j}} \quad (25.1)$$

where,

$P_{ini}$  is the initial active power of the load.

**Distributed slack by synchronous generators:** all the synchronous generators in the isolated area will contribute to the balancing. As in the *Distributed slack by loads* option, the distribution factor calculated for a generator is determined by the following equation:

$$K_i = \frac{P_{ini,i}}{\sum_{j=1}^n P_{ini,j}} \quad (25.2)$$

where,

$P_{ini}$  is the initial dispatched power of the generator.

**Distributed slack by synchronous generators and static generators:** same as *Distributed slack by synchronous generators*, but taking into account also static generators.

### 25.3.2.3 Reference Bus

**Reference Busbar:** a different busbar to the one connecting the slack machine (or network) can be selected as a reference for the voltage angle. In this case the user must specify the value of the voltage angle at this selected reference bus, which will be remotely controlled by the assigned slack machine (or network).

**Angle:** user-defined voltage angle for the selected reference busbar. The value will be remotely controlled by the slack machine (external grid). Only available if a *Reference Busbar* has been selected.

---

**Note:** Only one reference node can be set. Networks with multiple isolated areas automatically select a reference bus according to the *by reference machine* setting except for the isolated area with the user defined reference node. The *balancing by load at reference bus* and *by static generator at reference bus* are only available for one isolated area.

---

### 25.3.2.4 Interchange Schedule

This option is available only when the *Distributed slack by loads*, *Distributed slack by synchronous generators* or *Distributed slack by synchronous generators and static generators* is selected. It allows the loads or generation in a region to be scaled up or down to control the interchange of this region. The regions can be defined by Grids, Boundaries, Zones or Areas.

In the load flow page of the grid, boundary, zone and area elements, the following operational parameters are available:

- **Consider Interchange Schedule:** enables or disables the *Interchange Schedule* for this region. By default this option is not selected.
- **Scheduled active power interchange:** for setting the expected active power interchange.

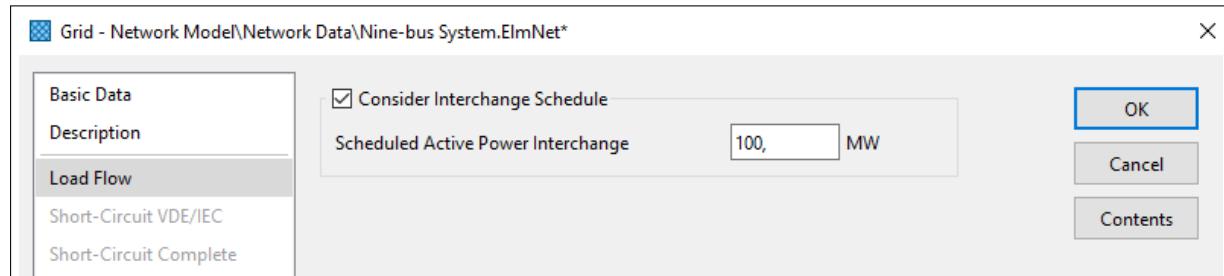


Figure 25.3.1: *Consider Interchange Schedule* option in the Load Flow page of the Grid element (*ElmNet*)

---

**Note:** Each boundary with a considered interchange schedule must define an interior and an exterior region. Furthermore each network element can only be associated to one boundary with an interchange schedule. The respective boundaries must therefore be disjointed. Detailed interchange schedules can be implemented with power frequency controllers (*ElmSecctr*).

---

Prior to version 2017 of *PowerFactory*, it was not possible to execute contingency analysis with this option to consider interchange schedule selected, but from version 2017 onwards, Contingency Analysis supports the use of interchange schedules for contingencies, provided that they are also considered in the base case.

### 25.3.3 Advanced Options

#### 25.3.3.1 Tap Adjustment

##### Method

- **direct:** this method will include the tap controller models in the load flow calculation (i.e. in the internal loop involving the Newton-Raphson iterations). The new tap positions will then be calculated directly as a variable and are therefore known following a single load flow calculation.
- **stepped:** this method will calculate a load flow with fixed tap positions, after which the required tap changes are calculated from the observed voltage deviations and the tap controller time constants. The load flow calculation is then repeated with the new tap positions, until no further changes are required. These tap adjustments take place in the outer loop of the calculation.

Further information on these methods can be found in the Technical References Document of corresponding elements of interest.

##### Min. Controller Relaxation Factor

The tap controller time constants are used in the automatic tap changer calculations to determine the relative speed of the various tap controllers during the load flow iterations. The relaxation factor can be used to slow down the overall controller speeds (in case of convergence problems, set a factor of less than 1.0), or to speed them up (for a faster load flow, set a factor of greater than 1.0). Reducing the relaxation factor results in an increased number of iterations, but yields greater numerical robustness.

##### Automatic detection of tap hunting

Occasionally, users executing a load flow calculation can find that it fails to converge in the outer loop just because a transformer is toggling between one tap position and another. To avoid this situation the load flow algorithm will detect such behaviour and stop the affected transformer from tapping further. By default, the corrective measure will be taken after three transitions, but this number is configurable by the user.

##### Automatic detection of repeated reactive power limitations

A similar preventative measure to the detection of transformer tap hunting is applied to generators, which can also toggle between being at a reactive power limit and being released from that limit. By default, the corrective measure will be taken after the generator has been released from its limit three successive times, but this number is configurable by the user.

#### 25.3.3.2 Operational Limits

**Consider operational limits for tap changer:** this option globally enables or disables the operational tap limits of transformers. If this option is disabled the limits from the transformer type are used.

Here it is also possible to specify which element classes should have their active and/or reactive power limits observed. These selections come into play if the options Consider active power limits and/or Consider reactive power limits are selected on the Basic Options page of the Load Flow Command dialog, as described in Section 29.3.1.

### Considered Models for Active Power Limits

The following element classes can be independently selected:

- Synchronous machine (*ElmSym*)
- Static generator (*ElmGenstat*)
- Asynchronous machine (e.g. *ElmAsm*)
- PWM converter (e.g. *ElmVsc*)

An additional flag *Limit active power also for const. P machines*, which by default is selected, can be deselected if the user does not want limits to be considered for constant P machines.

### Consider reactive power limits scaling factor

This option is only available if *Consider reactive power limits* is enabled. If selected, the reactive power limits of generators are scaled by the relaxation factors: *Scaling factor (min)* and *Scaling factor (max)* which are set on the *Load Flow* page of the generator element's dialog. Note that the reactive power limits of generators are also defined on the *Load Flow* page of the generator element's dialog by one of the following: maximum/minimum values, or according to the generator's assigned type.

### Considered Models for Reactive Power Limits

Here it is possible to specify which element classes should have their reactive power limits observed. The following element classes can be independently selected:

- Synchronous machine (*ElmSym*)
- Static generator (*ElmGenstat*)
- Asynchronous machine (e.g. *ElmAsm*)
- PWM converter (e.g. *ElmVsc*)
- Static Var system (*ElmSvs*)
- External grid (*ElmXnet*)
- Reference machine

The last option makes it possible to ignore the reactive limits on the reference machine even if they are observed for other machines of the same class. This option is *only* relevant if the flag for the corresponding element class has been checked, and the flag for the reference machine is not checked. (If the flag for the corresponding class has not been checked, then limits will not be observed for the reference machine whatever the setting.)

An additional flag *Limit reactive power also for const. Q machines*, which by default is selected, can be deselected if the user does not want limits to be considered for constant Q machines.

#### 25.3.3.3 Simulation Options

This tab is not only important for load flow but also for other calculation functions such as transient simulation. Utilising the options on this page can result in improved performance; i.e. the speed of a transient simulation may improve when protection devices are neglected in the calculation.

### Consider Protection Devices

Calculates the tripping times for all modelled relays and fuses. This will also show the load currents in the overcurrent plots and/or the measured impedance in the R-X diagrams. Disabling this option will speed up the calculations.

### Ignore Composite Elements

Disables all controller models. The panes *Models Considered* and *Models Ignored* are used to disable specific groups of controller models. Model names can be moved between these panes by either double-clicking on them or by selecting them and using the arrow buttons. Enabling this option may result in faster convergence, or an increased likelihood of convergence for systems which are otherwise difficult to solve.

#### 25.3.3.4 Advanced

##### Station Controller

Available on Advanced tab of the *Advanced Options* page. The options presented in this field determine the reactive power flow from generators participating in station controllers (*ElmStactrl*). Refer to section Load Flow Controllers of the [Technical References Document](#) for information on station controllers and their control modes.

##### Modelling Method of Towers

- **with in/output signals:** the equations of the lines are modelled in the tower. It should be noted that selecting this option will result in slower performance.
- **ignore couplings:** inter-circuit couplings are ignored.
- **equations in lines:** the constant impedance and admittance matrices are calculated by the tower and used to develop the equations of the lines. The equations involving coupling are modelled in the lines; consequently, using this option results in faster performance than using option *with in/output signals*.

##### Use this load flow for initialisation of OPF

The results of this load flow calculation are used to initialise the OPF calculation.

##### Calculate max. current at busbars

This option calculates the maximum current that will be seen along a busbar. In reality, the current along the different sections of a busbar will be dependent on the layout of the substation. Since the busbar has no length in *PowerFactory*, the connection points of circuits connected to the busbar can be defined using Bay objects (*ElmBay*). If a substation is created using Bay objects, the numbering of the Bays allows the load flow calculation to calculate the current flows on the busbar and hence the maximum current seen (*I<sub>max</sub>*). To configure a project such that new substations are created with Bays, the user should go to *Project Settings* → *Graphic* → *Insert substations with bays*.

##### Train simulation

This option enables a performance-optimised consideration of changing locations of trains (train elements), which are moved along lines within the network model. The option is only relevant for load flow calculations that are part of a train simulation. The lines which represent railway tracks with overhead contact lines (catenary) or conductor rails have to be marked accordingly as line for train simulation. Single-phase AC lines (AC, AC/BI) as well as DC lines are supported for train simulation. Only if the two aforementioned options are selected (*Train simulation* and *Line for train simulation*), are trains considered with their locations on lines. Otherwise a train element can only be connected to a terminal (node), as any other network element with fixed location, or is (without connection to a terminal) neglected in load flow calculation. For further information on train simulation, refer to Section [25.4.6](#).

### Consider coincidence of low-voltage loads

---

**Note:** The usage of coincidence curves is handled in the more sophisticated Low Voltage Load Flow Calculation (*ComLvldf*) that is part of the *PowerFactory* Distribution Network Tools, and described in Section 42.5. However, a simple approach with a pre-defined coincidence function is possible in the normal Load Flow, as described in the following.

---

If this option is selected, a 'low voltage load flow' is calculated, where load coincidence factors are considered, to produce maximum branch currents and maximum voltage drops. Since coincidence factors are used, the results will not obey Kirchhoff's current law. After the load flow has been successfully executed, maximum currents (*I<sub>max</sub>*), maximum voltage drops (*d<sub>umax</sub>*) and minimum voltages (*U<sub>min</sub>*, *U<sub>min</sub>*) are displayed in every branch element and at every busbar. Losses are calculated based on average values, and maximum branch loading is calculated using maximum currents.

For this calculation, low voltage loads (*ElmLodlv* and *ElmLodlp*) are modelled with a "Base Load" and an "Additional Load". While the "Base Load" is fixed, the "Additional Load" is influenced by the coincidence factor. It is defined by the LV load type (*TypLodlv*) and the "Utilisation Factor" in the LV load in *Load Flow* → *Additional Load* → *Load per customer*. The maximum value of the "Additional Load" (which is dependent upon the number of customers, *n*) is described by the following formula:

$$S_{max}(n) = n \cdot g(n) \cdot S_{max} \quad (25.3)$$

Where *S<sub>max</sub>* is the maximum "Additional Load" per connection (customer) and the function *g(n)* describes the maximum coincidence of loads, dependent upon the number of connections, *n*. If a Gaussian distribution is assumed, the coincidence function is:

$$g(n) = g_\infty + \frac{1 - g_\infty}{\sqrt{n}} \quad (25.4)$$

The average value of the "Additional Load" is calculated with  $g_\infty \cdot S_{max}$

### Voltage Drop Analysis

This option appears if *Consider coincidence of low-voltage loads* is selected. For the consideration of the stochastic nature of loads, *PowerFactory* offers two calculation methods:

- **Stochastic Evaluation:** more theoretical approach, and can also be applied to meshed network topologies.
- **Maximum Current Estimation:** applies stochastic rules only for the estimation of maximum branch flows. Based on the maximum current flow in each branch element, maximum voltage drops are calculated and added along the feeder. Obviously, this method has its limitations in case of meshed LV networks.

### 25.3.4 Calculation Settings

#### 25.3.4.1 Algorithm

##### Load Flow Method

As explained in Section 25.2.1, the nodal equations used to represent the analysed networks are implemented using two different formulations:

- Newton-Raphson (Current Equations)
- Newton-Raphson (Power Equations, classical)

In both formulations, the resulting non-linear equation systems must be solved using an iterative method. *PowerFactory* uses the Newton-Raphson method as its non-linear equation solver. The selection of the method used to formulate the nodal equations is user-defined, and should be selected based on the type of network to be calculated. For large transmission systems, especially when heavily loaded, the classical Newton-Raphson algorithm using the *Power Equations* formulation usually converges best. Distribution systems, especially unbalanced distribution systems, usually converge better using the *Current Equations* formulation.

#### 25.3.4.2 Iteration Control

The options on this tab relate to the non-linear equation solver and are therefore only available for *PowerFactory*'s AC load flow calculation methods.

##### Max. Number of Iterations for

The load flow calculation comprises an inner loop involving the Newton-Raphson method (see Section 25.2.1), and an outer loop to determine changes to tap settings and to consider generator reactive power limits. Default values for the maximum number of iterations for these two loops are 25 iterations for the inner loop, and 20 iterations for the outer loop.

- **Newton-Raphson Iteration:** the inner loop of the load flow involves the Newton-Raphson iterations. This parameter defines the maximum number of iterations (typically 25).
- **Outer Loop:** the outer loop of the load flow calculation will determine changes to the tap changer (depending on the tap adjustment method selected), and considers reactive power limits of generators, etc. These are adjusted in the outer loop and then a new iteration of the inner loop is started again (see Section 25.2.1). The maximum number of outer loop iterations (typically 20) is set by this parameter.
- **Number of Steps:** problematic load flows with slow or no convergence may be improved by starting a load flow calculation for a low load level, then increasing the load level gradually in the given number of steps. This is achieved by setting the *Number of Steps* to a value greater than one. For example, *nsteps* = 3 begins a load flow at a load/generation level of 1/3 and then increases the power to 100 % over two further steps.

##### Max. Acceptable Load Flow Error for

A higher precision or a faster calculation can be obtained by changing the maximum allowable error (i.e. tolerance). The values of the calculated absolute error for nodes, or the calculated relative errors in the model equations, e.g. voltage error of voltage controlled generators, are specified here.

- **Nodes:** maximum Iteration Error of Nodal Equations (typical value (Default): 1 kVA). The thresholds can be distinguished and entered for different voltage levels, where the voltage levels itself are definable as well in the project settings:
  - **Bus Equations (HV):** Default  $U_{HV} > 66 \text{ kV}$
  - **Bus Equations (MV):** Default  $U_{MV} > 1 \text{ kV}$
  - **Bus Equations (LV):**  $U_{LV} > 0 \text{ kV}$
- **Model Equations:** maximum Error of Model Equations (typical value: 0.1 %).

### Iteration step size

- **automatic adaptation:** default option.
- **fixed relaxation:** when this option is selected, a **Relaxation Factor** can be entered. A Newton-Raphson relaxation factor smaller than 1.0 will slow down the convergence speed of the load flow calculation, but may result in an increased likelihood of convergence for systems which are otherwise difficult to solve.

This following option can be used to reduce the time taken by a load flow that is not reaching a convergent solution.

- **Trim unreasonable Newton-Raphson steps:** if this option is selected, the user specifies a limit to the number of iterations to be carried out when convergence is not being reached.

### Automatic Model Adaptation for Convergency

The *PowerFactory* load flow calculation will always first try to find a solution using non-linear mathematical power system models. If a solution cannot be found, and this option is enabled, an adaptive algorithm will change these models slightly to make them more linear, until a solution is found. Any model adaptations are reported in the output window.

Iteratively, starting from Level 1 up to Level 4, some types of models are adjusted in order to find a solution. The adaptations of the models for each level are the following:

- **Level 1**
  - Loads: All voltage dependency factors are set to minimum 0.5
  - Generators and external grids: Reactive power limits are disabled
  - Transformers: tap control is disabled
  - Motors: The rotor resistance is not allowed to vary
- **Level 2**
  - Loads: All voltage dependency factors are set to minimum 0.8
  - Generators and external grids: Reactive power limits are disabled
  - Transformers: tap control is disabled
  - Motors: The rotor resistance is not allowed to vary
- **Level 3**
  - Loads: All voltage dependency factors are set to minimum 2
  - Generators and external grids: Reactive power limits are disabled
  - Transformers: tap control is disabled
  - Motors: The rotor resistance is not allowed to vary
- **Level 4**
  - Loads: All voltage dependency factors are set to minimum 2
  - Generators and external grids: Reactive power limits are disabled and voltage equation are linearised
  - Transformers: tap control is disabled
  - Motors: The rotor resistance is not allowed to vary

The models are not only linearised but also simplified. If Level 4 is reached, the user should consider switching to the DC load flow method.

### 25.3.4.3 Initialisation

This page allows the user to influence the way in which the load flow calculation is initialised. The default settings are suitable in most cases.

---

**Note:** Not all the options listed below are available for DC load flows. In particular, the options to save results and use these as a starting point for subsequent load flows are only applicable for AC load flow calculations.

---

#### No Topology Rebuild

Will speed up large sets of consecutive load flow calculations. Enabling this option means that the topology of the system will not be rebuilt when calculating the next load flow. If no topological changes will be made to the system between these consecutive load flow calculations, then this option may be enabled.

#### Max. transformer phase shift

When a network contains phase-shift transformers whose tap settings cause a large voltage change, the initial condition of the load flow can be far from the solution, which can result in non-convergence. To avoid this problem, the load flow calculation can introduce the phase shift more gradually. For any phase-shift transformer where it is detected that the total phase shift will be greater than the specified threshold (the default is  $20^\circ$ ), it will be broken down into two or more steps, in successive outer loops.

#### Voltage magnitude initialisation

When initialising the voltages for a load flow calculation, the starting point for the voltage magnitude is always the voltage setpoint of the reference busbar. By default, the starting voltages of other terminals are then determined taking into account their nominal voltage and the rated voltage of branch elements connecting them when tracing through the network. This means that - for example - if the rated voltages of a transformer (defined by its type) do not quite align with the nominal voltages of the terminals to which it is connected, this is taken into account when calculating the initial voltage magnitudes. If the rated voltage of a line does not align with the nominal voltage of a terminal it is connected to, this will also be taken into account. The options below allow the user to modify this default approach if required.

- **Use voltage setpoint at reference busbar:** terminal voltage magnitudes are simply initialised at the same p.u. value as the reference busbar
- **Consider ratio of rated to nominal voltage (Default setting):** take into account the voltage ratios calculated for transformers and other branch elements when determining the initial voltage magnitudes
- **Consider ratio of rated to nominal voltage for all branches except transformers:** as with the default setting, but not considering transformers
- **Consider ratio of rated to nominal voltage for transformers only:** as with the default setting, but only considering transformers

#### Starting point

This panel contains options relating to the initialisation of the load flow calculation. The load flow can be initialised from a “flat start” (which is the default), or starting from the last calculated results, or from saved results. More details about this functionality can be found in Section [25.4.5](#)

- **Start from last calculated results if available:** If previous results have not been reset, these will be used as the starting point for the load flow. This will be the case even if the *Start from saved results* flag is checked.

- **Start from saved results:** Use previously saved results.
  - Source: the user selects the source of the saved results, which can either be in memory or in a file. If “from a file” is selected, the user must supply the file name and location.

The **Save results** button becomes active once a successful load flow calculation has been run. Clicking on the button brings up a new dialog with the following options:

- **To memory:** Results to be stored in memory
- **To a file:** Results to be stored in a file, whose name and location can be specified by the user (but otherwise will default to the workspace)
- **Adapt starting point settings in the Load Flow command:** Ensures that the Load Flow command is set up correctly to use these stored results.

If results are to be saved to a file, the user must choose whether objects should be identified by object ID or full path name. See Section [25.4.5.2](#) for more information about these options.

### 25.3.5 Outputs

#### Show Outer Loop messages

Will print a report concerning the outer loop iterations, which may be used to solve convergence problems.

#### Show Convergence Progress Report

Will print a detailed report throughout the load flow calculation. When enabling this option the *Number of reported buses/models per iteration* can be stated. As a result, the required number of buses and models with the largest error will be reported (e.g. by stating 3, the 3 buses and models with the largest error will be printed out in the output window). As in the case of *Outer Loop* messages, this information can be useful in solving convergence problems.

#### Maintain load flow results if calculation fails

The results of the last iteration will be displayed. These may be used to identify and solve convergence problems.

#### Check Control Conditions

This option is selected by default and lists in the output window all elements whose control conditions (e.g. reactive power limits, etc.) have not been fulfilled. The arrow button → opens the edit dialog of the *Check Control Conditions* command (*ComCheckctrl*). On the *Basic Options* page of the command the following options can be selected:

- **Elements to be reported:** to choose whether to report only the elements where the violation is control-relevant, or all the elements. If the first option is selected, violations of limits will only be reported if the violation is caused by an automated control in load flow calculation. As an example, violations of reactive power limits would be reported for a generator that is set to control voltage, and so has a reactive power output that can vary in the load flow, but not for a generator that is set to *Const. Q* and whose reactive power set-point has been set outside the specified limits.
- **Report violations of limits:** to select the limits for which violations are to be reported (i.e. active power, reactive power and/or voltage) and set tolerances.
- **Report variables at limits:** to select whether to report elements at limits and which limits to check (i.e. active power, reactive power and/or voltage). This option is relevant for limits that are not selected (in the *Load Flow Calculation* command) to be considered.

On the page *Element Type* the control devices to be checked can be selected:

- Generator
- Transformer
- Shunt and SVC
- Station Controller
- Tap Controller
- Others

### 25.3.6 Load/Generation Scaling

On this page it is possible to define global scaling factors for loads, generators, motors and night storage heaters that will be used only during the execution of the Load Flow Calculation.

The following categories are available:

#### Loads

The active/reactive power of the following elements is scaled by the Load Scaling Factor:

- General Load (*ElmLod*)
- MV Load, load part (*ElmLodmv*)
- LV Load (*ElmLodlv* and *ElmLodlvp*)

#### Generation

The active/reactive power of the following elements is scaled by the Generation Scaling Factor:

- Static Generator (*ElmGenstat*)
- Synchronous Generator (*ElmSym*), when connected as a Generator
- Asynchronous Generator (*ElmAsm*), when connected as a Generator
- DFIG Generator (*ElmAsmsc*), when connected as a Generator
- MV Load (*ElmLodmv*), generation part

#### Motors

The active/reactive power of the following models is scaled by the Motor Scaling Factor:

- Synchronous Motor (*ElmSym*), when connected as a Motor
- Asynchronous Motor (*ElmAsm*), when connected as a Motor
- DFIG Motor (*ElmAsmsc*), when connected as a Motor

#### Night storage heaters

In LV Loads (*ElmLodlv* and *ElmLodlvp*), the active power in *Load Flow* → *Additional Load* → *Night Storage Heater* is scaled by the Night storage heater Scaling Factor.

## Zone Scaling

The Load Scaling Factor of a zone (parameter curscale) can be applied either to all loads or just loads which have the Adjusted by Load Scaling parameter selected.

## 25.4 Detailed Description of Load Flow Calculation Options

The following sections describe the options available in the Load Flow Calculation command in detail.

### 25.4.1 Active and Reactive Power Control

#### 25.4.1.1 Active Power Control

Besides the traditional approach of using a slack generator to establish the power balance within the system, *PowerFactory*'s load flow calculation tool provides other active power balancing mechanisms which more closely represent the reality of transmission networks (see selection in the *Active Power Control* page of the load flow command). These mechanisms are implemented in the steady-state according to the control processes that follow the loss of large power stations:

**As Dispatched:** as mentioned at the beginning of this section, the conventional approach in load flow calculations consists of assigning a slack generator, which will establish the power balance within the system. Besides this traditional approach, *PowerFactory* offers the option of balancing by means of a single or a group of loads (*Distributed Slack by Loads*). Under such assumptions, the active power of the selected group of loads will be modified so that the power balance is once again met; while leaving the scheduled active power of each generator unchanged. Other methods of balancing include considering the participation of all synchronous generators according to their scheduled active power (*Distributed Slack by Generation*).

**According to Secondary Control:** if an unbalance occurs between the scheduled active power values of each generation unit and the loads plus losses, primary control will adapt (increase/decrease) the active power production of each unit, leading to an over- or under-frequency situation. The secondary frequency control will then bring the frequency back to its nominal value, re-establishing cost-efficient generation delivered by each unit. Secondary control is represented in *PowerFactory*'s load flow calculations by network components called *Power Frequency Controllers* (*ElmSecctrl*). If the *Active Power Control* option *According to Secondary Control* is selected, the generators considered by the *Power Frequency Controller* establish the active power balance according to their assigned participation factors. It is also possible, within the group of controlled generators, to implement a merit order priority; if this option is selected in the *Power Frequency Controller*, generators with the highest merit order priority will be dispatched first, as far as their operational limits allow, then the generators with successively lower merit order priorities as required.

In cases where the total required MW change in generation exceeds the total available on generators with a non-zero merit-order, the remainder will be distributed between generators with a zero merit-order, according to the Primary Frequency Bias (Kpf) of these generators, those with a higher Kpf being used first.

(For further information, refer to the [Technical References Document](#)).

**According to Primary Control:** shortly following a disturbance, the governors of the units participating in primary control will increase/decrease their turbine power and drive the frequency close to its nominal value. The change in the generator power is proportional to the frequency deviation and is divided among participating units according to the gain ( $K_{pf}$ ) of their primary controllers and which is depicted in Figure 25.4.1. If the Active Power Control option According to Primary Control is selected in *PowerFactory*'s load flow command, the power balance is established by all generators (synchronous generators, static generators and external grids) having a primary controller gain value different than

zero (parameter Prim. Frequency Bias in the Load Flow page - Figure 25.4.2). The modified active power of each generator is then calculated according to the following equation:

$$P_i = P_{i-dispatch} + \Delta P_i \quad (25.5)$$

where

$P_i$  is the modified active power of generator  $i$ ,  
 $P_{i-dispatch}$  is the initial active power dispatch of generator  $i$  and  
 $\Delta P_i$  is the active power change in generator  $i$ .

The active power change of each generator ( $\Delta P_i$ ) will be determined by its corresponding primary controller gain value ( $K_{pf-i}$ ) and the total frequency deviation.

$$\Delta P_i = K_{pf-i} \cdot \Delta f \quad (25.6)$$

where

$K_{pf-i}$  is the primary controller gain parameter of generator  $i$  and  
 $\Delta f$  is the total frequency deviation.

The total frequency deviation ( $\Delta f$ ) can be obtained according to:

$$\Delta f = \frac{\Delta P_{Tot}}{\sum K_{pf}} \quad (25.7)$$

where  $\Delta P_{Tot}$  corresponds to the active power change sum of every generator:

$$\Delta P_{Tot} = \sum_{j=1}^n \Delta P_j \quad (25.8)$$

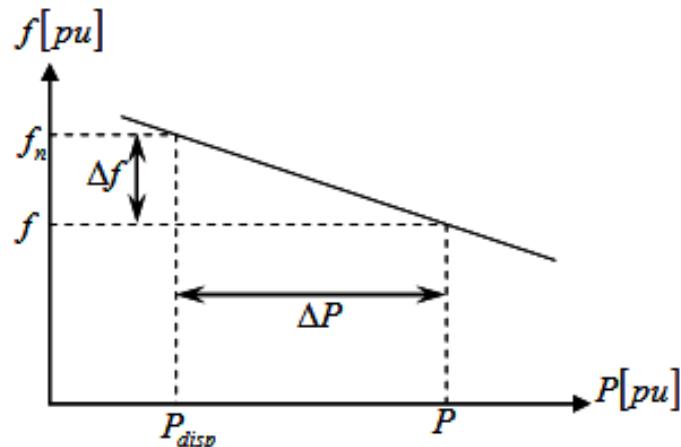


Figure 25.4.1: Primary Frequency Bias

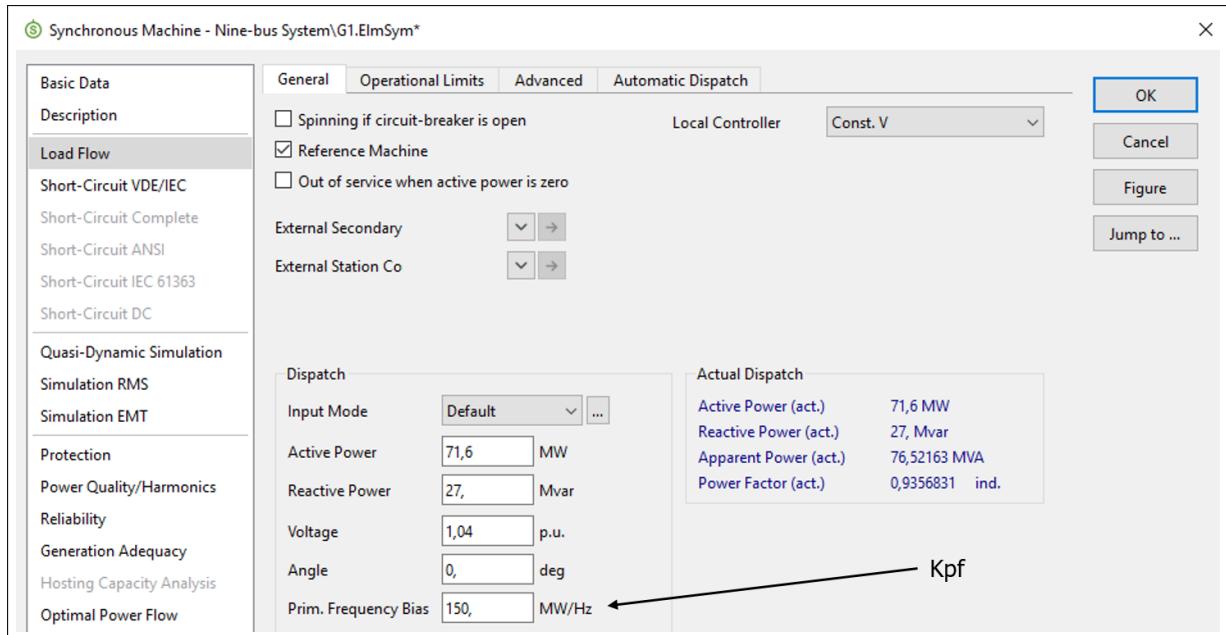


Figure 25.4.2: Primary Frequency Bias ( $K_{pf}$ ) Setting in the Load Flow Page of the Synchronous Machine Element (*ElmSym*)

Consider the following example:

Three generators supply a load.

- G1, has a set dispatch of 2 MW and a primary controller gain of 2 MW/Hz.
- G2, has a set dispatch of 2 MW and a primary controller gain of 2 MW/Hz.
- G3, has a set dispatch of 3 MW and a primary controller gain of 1 MW/Hz.
- G3 is set as the reference machine.

An *as dispatched* active power control indicates that a total active power output of 5 MW is required from the 3 generators to supply the connected load:

G1 and G2 each supply 2 MW corresponding with their dispatch setpoints.

As the reference machine, G3 supplies the final 1 MW.

The mismatch between the supplied power and the dispatched power is therefore.

$$\Delta P_{Tot} = 5 \text{ MW} - 7 \text{ MW} = -2 \text{ MW} \quad (25.9)$$

The total frequency deviation is therefore:

$$\Delta f = \frac{-2 \text{ MW}}{5 \text{ MW/Hz}} = -0.4 \text{ Hz} \quad (25.10)$$

The active power deviation between the dispatch setpoint of each generator and its primary controlled output is therefore:

$$\Delta P_{G1} = 2 \text{ MW/Hz} \cdot -0.4 \text{ Hz} = -0.8 \text{ MW} \quad (25.11)$$

$$\Delta P_{G2} = 2 \text{ MW}/\text{Hz} \cdot -0.4 \text{ Hz} = -0.8 \text{ MW} \quad (25.12)$$

$$\Delta P_{G3} = 1 \text{ MW}/\text{Hz} \cdot -0.4 \text{ Hz} = -0.4 \text{ MW} \quad (25.13)$$

Finally, The primary controlled output of each generator is therefore:

$$P_{G1} = 2 \text{ MW} - 0.8 \text{ MW} = 1.2 \text{ MW} \quad (25.14)$$

$$P_{G2} = 2 \text{ MW} - 0.8 \text{ MW} = 1.2 \text{ MW} \quad (25.15)$$

$$P_{G3} = 3 \text{ MW} - 0.4 \text{ MW} = 2.6 \text{ MW} \quad (25.16)$$

**According to Inertias:** immediately following a disturbance, the missing/excess power is delivered from the kinetic energy stored in the rotating mass of the turbines. This leads to a deceleration/acceleration and thus to a frequency decrease/increase. The power balance is established by all synchronous machines if the *Active Power Control* option *According to Inertias* is selected in the load flow command. Individual contributions to the balance are proportional to the inertia/acceleration time constant of each generator (defined on the *RMS-Simulation* page of the synchronous generator type's dialog and depicted in Figure 25.4.3). This relation can be mathematically described as follows:

$$P_i = P_{i-dispatch} + \Delta P_i \quad (25.17)$$

where

$P_i$  is the modified active power of generator  $i$ ,

$P_{i-dispatch}$  is the initial active power dispatch of generator  $i$  and

$\Delta P_i$  is the active power change in generator  $i$ .

The active power change of each generator  $\Delta P_i$  is determined by its inertia gain and the initial frequency response as follows:

$$\Delta P_i = J_i \cdot \omega_n \cdot 2\pi \cdot RoCoF_{CoI}|_{t=0+} \quad (25.18)$$

where

$RoCoF_{ini,CoI}$  is the initial Rate of Change of Frequency for the Center of Inertia and

$J_i$  is generator's the moment of inertia.

The respective load flow signal representing the  $RoCoF_{ini,CoI}$ , which is used to set the active power of the Synchronous Machines, is `s:dFin`. The inertia of each machine is described by

$$J_i = S_{n,i} \cdot \frac{T_{ag,i}}{\omega_n^2} = S_{n,i} \cdot \frac{2 \cdot H_i}{\omega_n^2} \quad (25.19)$$

where

$\omega_n$  is the rated angular velocity,

$S_{n,i}$  is the generator's rated apparent power),

$T_{ag,i}$  is the generator's acceleration time constant rated to  $S_{n,i}$  and

$H_i$  is the generator's inertia constant rated to  $S_{n,i}$  with  $T_{ag,i} = 2 \cdot H_i$ .

**Note:** The generator's rated apparent power  $S_{n,i}$  must be exchanged with the rated active power in all presented equations if  $H_i$  or  $T_{ag,i}$  are rated to the active power instead of the apparent power.

The  $RoCoF_{CoI}|_{t=0^+}$  is a simplified estimation of a system's frequency response to an active power imbalance and depends on the weighted sum of the inertia constant. It is calculated with

$$RoCoF_{CoI}|_{t=0^+} = \frac{f_n \cdot \Delta P_{sys}}{2 \cdot S_{n,sys} \cdot H_{sys}} \quad (25.20)$$

where

$f_n$  is the nominal system frequency,

$\Delta P_{sys}$  is the total active power imbalance of the system,

$S_{n,sys}$  is the sum the nominal apparent power of all generators  $k$  contributing to the inertia and

$H_{sys}$  is the weighted system inertia constant, which is calculated as

$$H_{sys} = \frac{\sum_i^k H_i \cdot S_i}{\sum_i^k S_i} \quad (25.21)$$

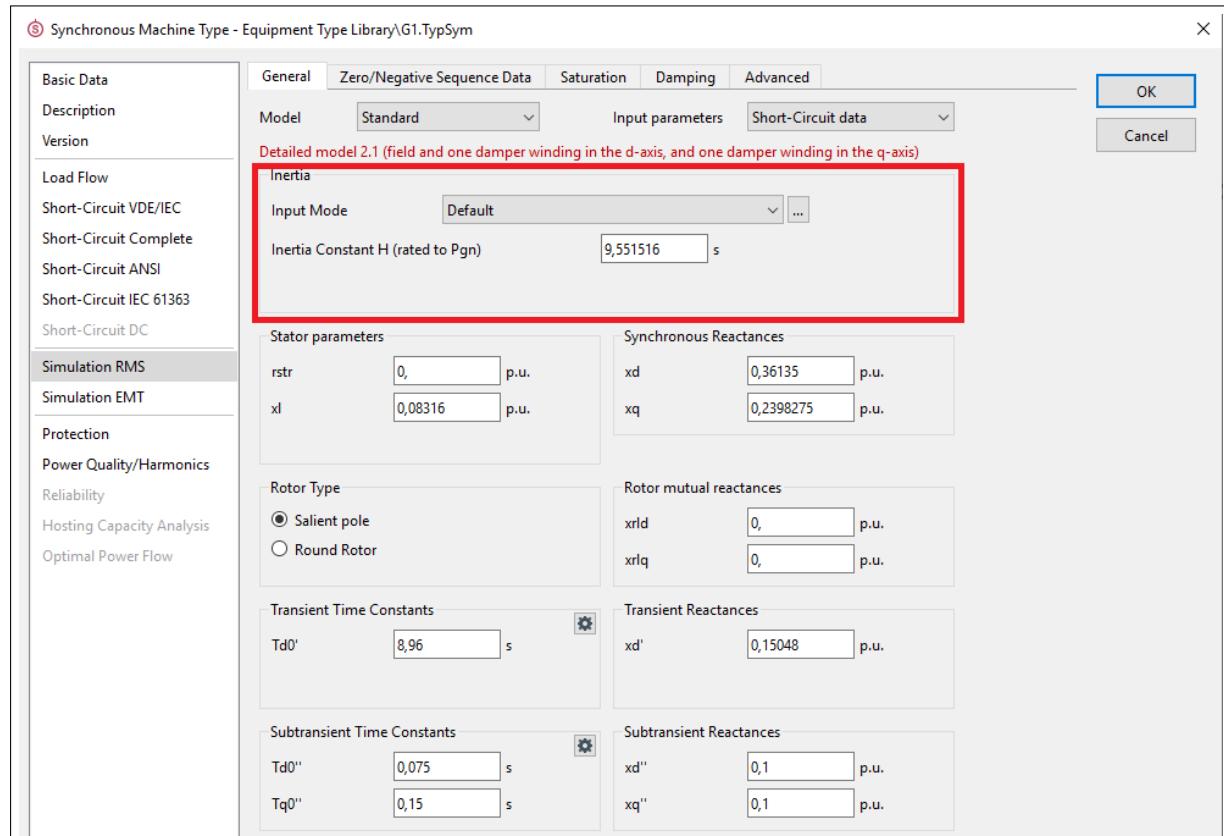


Figure 25.4.3: Inertia/Acceleration Time Constant Parameter of the Synchronous Machine Type (TypSym). *Simulation RMS* Page

Figure 25.4.4 illustrates the different types of active power control.

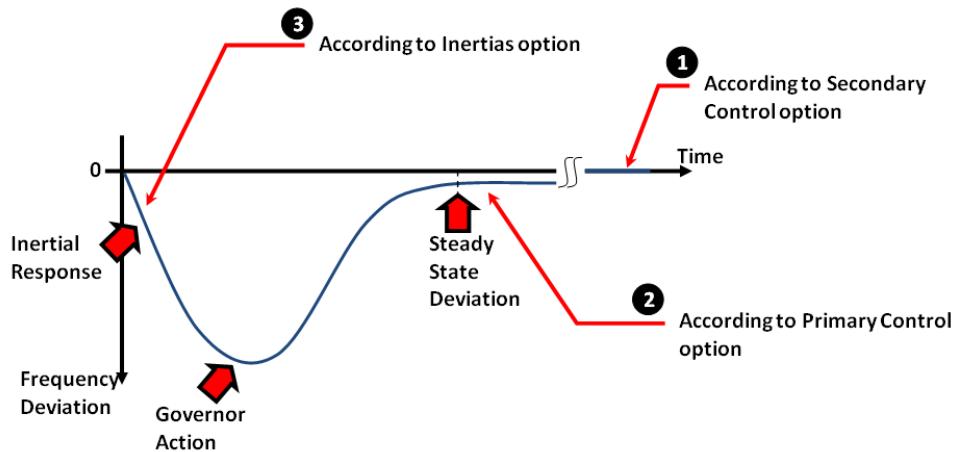


Figure 25.4.4: Frequency Deviation Following an Unbalance in Active Power

**Note:** The Secondary Control option will take into account the participation factors of the machines defined within a *Power-Frequency Controller* (*ElmSecctr*) in order to compensate for the frequency deviation. In such a case, the final steady state frequency is considered to be the nominal value (number 1 in Figure 25.4.4). The Primary Control option will take into account the frequency droop (MW/Hz) stated in every machine in order to determine the active power contribution. Depending on the power unbalance, the steady state frequency will deviate from the nominal value (number 2 in Figure 25.4.4). The According to Inertias option will take into account the inertia/acceleration time constant stated in every machine in order to determine its active power contribution. In this case, depending on the power unbalance, the steady state frequency will deviate from the nominal value (number 3 in Figure 25.4.4).

#### Forced voltage angle control:

This setting is not part of the load flow command but an option for synchronous machines and static generators, which are marked as reference machines. By default, the voltage angle of such machines, which have not been selected as the reference machine by *PowerFactory*, is automatically calculated during the load flow. However, some network elements, such as the coupling of a 50 Hz system to a 16 2/3 Hz system, may require a specific voltage angle to control the power flow. The voltage angle of an individual machine, which is set on the *Load Flow → General* page of a synchronous machine or a static generator, can be forced using the *Forced voltage angle control (const. phi)* option on the *Load Flow → Advanced* page. This option is only visible, if the element is set as a reference machine. It is also possible to use a QDSL model for more complex voltage angle controllers.

##### 25.4.1.2 Reference Machine Selection

The reference (slack) machine for a load flow calculation can be selected directly. This is done for synchronous and static generators (*ElmSym*) and *ElmGenstat*) via the checkbox *Reference Machine* (*ip\_ctrl*) on the *Load Flow* page. In addition, voltage sources (*ElmVac*), which are not Ward Equivalents, as well as external networks (*ElmXnet*) with the bus type slack (SL) are considered. However, if the reference machine is not clearly defined for a network or an isolated part of the network, *PowerFactory* selects an element to act as the reference machine. This is the case, if none or multiple network elements are selected. It should be noted that the automatic selection of a reference machine is subject to a project setting for Automatic Slack Assignment, described in Section 9.1.3.3.

**Automatic Determination of Reference Machine if none selected**

*PowerFactory* designates a reference machine if the reference machine is not defined and the network does not contain any other slack element. The applied logic is designed to ensure that a strong network element is selected. The priority of each viable network element with relevant settings is calculated using the factor according to the table below.

Element	Condition	Factor
External Grid <i>ElmXnet</i>	$bustp \neq SL$	$Snom$
Synchronous Machine <i>ElmSym</i>	$i\_spin = 1$ $ip\_ctrl = 0$	$Snom$
	$i\_spin = 0$ $ip\_ctrl = 0$	$0.01 \cdot Snom$
Static Generator <i>ElmGenstat, ElmPvsys</i>	$ctrlStruct = \text{Grid-forming}$ $ip\_ctrl = 0$	$Snom$
PWM Converter <i>ElmVsc, ElmVsmono</i>	$ctrlStruct = \text{Grid-forming}$ $i\_acdc = \text{PMW-phi}$	$0.001 \cdot Snom$
	$ctrlStruct = \text{Grid-forming}$ $i\_acdc \notin \{\text{Vac-phi}, \text{Vdc-phi}, \text{Vdc-Q}, \text{Vdc-Vac}, \text{Vdc-cos(phi)}\}$	$0.00001 \cdot Snom$
AC Voltage Source <i>ElmVac</i>	$itype = \text{Extended Ward Equivalent}$ $\text{Slack Priority (iPossibleSlack):}$ <ul style="list-style-type: none"><li>• None: <math>k_{prio} = 0</math></li><li>• Low: <math>k_{prio} = 0.001</math></li><li>• Medium : <math>k_{prio} = 1</math></li><li>• High: <math>k_{prio} = 10^8</math></li></ul>	$k_{prio} \cdot (Pgen + 0.001)$
	$itype = \text{Ward Equivalent}$ $\text{Slack Priority (iPossibleSlack):}$ <ul style="list-style-type: none"><li>• None: <math>k_{prio} = 0</math></li><li>• Low: <math>k_{prio} = 0.001</math></li><li>• Medium : <math>k_{prio} = 1</math></li><li>• High: <math>k_{prio} = 10^8</math></li></ul>	$0.1 \cdot k_{prio} \cdot (Pgen + 0.001)$

Table 25.4.1: Reference machine prioritisation factors (no element set to reference machine)

---

**Note:** In the case that method 2 is selected in the project setting for *Auto slack assignment*, synchronous machines without the attribute *Spinning if circuit breaker is open* ( $i\_spin=0$ ) are not considered unless they're actively set to be considered as a reference machine. Further details are described in Section [9.1.3.3](#).

---

**Automatic Determination of Reference Machine with more than one selected**

If multiple network elements are set as reference machines, a similar prioritisation as described above according to the element class and size takes place. The respective factors are listed in the table below.

Element	Condition	Factor
External Grid <i>ElmXnet</i>	$bustp = SL$	$10\ 000 \cdot Snom$
Static Generator <i>ElmGenstat, ElmPvsys</i>	$ip\_ctrl = 1$	$100 \cdot Snom$
Synchronous Machine <i>ElmSym</i>	$ip\_ctrl = 1$	$100 \cdot Snom$
PWM Converter <i>ElmVsc, ElmVscmono</i>	$i\_acdc = Vac\text{-}\phi$	$0.01 \cdot Snom$
AC Voltage Source <i>ElmVac</i>	$itype = \text{Voltage Source}$ Slack Priority ( <i>iPossibleSlack</i> ): • None: $k_{prio} = 0$ • Low: $k_{prio} = 0.001$ • Medium : $k_{prio} = 1$ • High: $k_{prio} = 10^8$	$0.01 \cdot k_{prio} \cdot Unom \cdot usetp$

Table 25.4.2: Reference machine prioritisation factors (several elements set to reference machine)

**Note:** For synchronous machines *ElmSym* and static generators *ElmGenstat*, the user has a further possibility to influence the choice of slack machine. Within the project settings, Advanced Calculation Parameters page, there are three options for Priority for Reference Machines. These are:

- Rated Power:  $Snom$  is used as the criterion, as in the table above
- Active Power Capability:  $P_{max}\text{-}P_{min}$  is used instead of  $Snom$
- Active Power Reserve:  $P_{max}\text{-}P_{gini}$  is used instead of  $Snom$

### 25.4.1.3 Reactive Power Control

The reactive power reserves of synchronous generators in transmission networks are used to control the voltages at specific nodes in the system and/or to control the reactive power exchange with neighbouring network zones. In *PowerFactory*'s load flow calculation, the voltage regulator of the generators has a voltage setpoint which can be set manually (defining a PV bus type as introduced in Section 25.2.1), or from an Automatic Station Controller (*ElmStactrl*). This Automatic Station Controller combines several sources of reactive power to control the voltage at a given bus. In this case the relative contribution of each reactive power source (such as generators and SVSs) is defined in the Station Controller dialog. For further details about the use and definition of Automatic Station Controllers refer to the [Technical References Document](#).

## 25.4.2 Voltage Dependency of Loads

All non-motor loads, as well as groups of non-motor loads that conform a sub-system, for example, a low-voltage system viewed from a medium voltage system, can be modelled as a “general load”.

Under “normal conditions” it is permissible to represent such loads as constant PQ loads. However under “abnormal conditions”, for example during voltage collapse situations the voltage-dependency of the loads should be taken into account.

Under such assumptions, *PowerFactory* uses a potential approach, as indicated by Equations (25.22) and (25.23). In these equations, the subscript 0 indicates the initial operating condition as defined in the input dialog box of the Load Type.

$$P = P_0 \left( aP \cdot \left( \frac{v}{v_0} \right)^{e_{-aP}} + bP \cdot \left( \frac{v}{v_0} \right)^{e_{-bP}} + (1 - aP - bP) \cdot \left( \frac{v}{v_0} \right)^{e_{-cP}} \right) \quad (25.22)$$

where,

$$cP = (1 - aP - bP)$$

$$Q = Q_0 \left( aQ \cdot \left( \frac{v}{v_0} \right)^{e_{-aQ}} + bQ \cdot \left( \frac{v}{v_0} \right)^{e_{-bQ}} + (1 - aQ - bQ) \cdot \left( \frac{v}{v_0} \right)^{e_{-cQ}} \right) \quad (25.23)$$

where,

$$cQ = (1 - aQ - bQ)$$

By specifying the particular exponents ( $e_{-aP}$ ,  $e_{-bP}$ ,  $e_{-cP}$  and  $e_{-aQ}$ ,  $e_{-bQ}$ ,  $e_{-cQ}$ ) the inherent load behaviour can be modelled. For example, in order to consider a constant power, constant current or constant impedance behaviour, the exponent value should be set to 0, 1 or 2 respectively. In addition, the relative proportion of each coefficient can be freely defined using the coefficients  $aP$ ,  $bP$ ,  $cP$  and  $aQ$ ,  $bQ$ ,  $cQ$ . For further information, refer to the *General Load* technical reference in the [Technical References Document](#).

---

**Note:** These factors are only considered if the “Consider Voltage Dependency of Loads” is checked in the Load-flow Command window. If no Load Type (*TypLod*) is assigned to a load, and the load flow is performed considering voltage dependency then the load will be considered as Constant Impedance.

---

### 25.4.3 Feeder Load Scaling

In radially operated distribution systems the problem often arises that very little is known about the actual loading of the loads connected at each substation. The only information sometimes available is the total power flowing into a radial feeder. To be able to still estimate the voltage profile along the feeder a load scaling tool is used. In the simplest case the distribution loads are scaled according to the rated power ratings of the transformers in the substations. Of course, more precise results are obtained by using an average daily, monthly or annual load.

This is illustrated in Figure 25.4.5. Here, the measured value at the beginning of the feeder is stated to be 50 MW. Throughout the feeder there are three loads defined, of which only for one of them the load is precisely known (20 MW). The other two loads are estimated to be at around 10 MW each. PowerFactory's load flow analysis tool offers a special *Feeder Load Scaling* option so that the selected groups of loads (scalable loads) are scaled accordingly in order to meet the measured value.

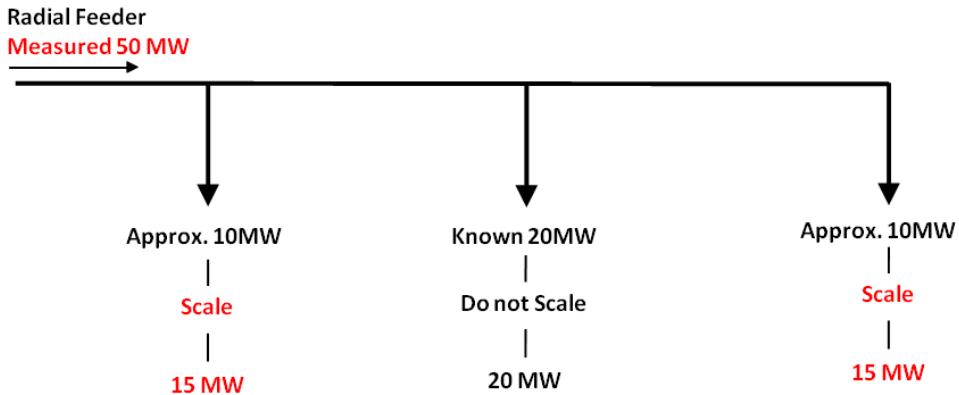


Figure 25.4.5: Radial Feeder. Feeder Load Scaling Option

In *PowerFactory* the following options for *Feeder Load Scaling* are available:

- No scaling.
- Scaling to measured apparent power.
- Scaling to active power.
- Scaling to measured current.
- Scaling Manually.
- Scaling to measured reactive power.
- Scaling to measured power factor.

Furthermore, the previous options can be combined; for example, scaling a selected groups of loads in order to meet a measured active power and power factor.

---

**Note:** Loads that are to be scaled must be marked as such (Adjusted by Load Scaling), also the load scaling must be enabled in the load flow command option (Feeder Load Scaling).

---

The feeder load scaling process also can take into account the different type of load behaviour represented. Figure 25.4.6 illustrates just this. Here, a radial feeder consisting of three different type of loads is depicted (constant power, constant current and constant impedance). Under such assumptions, performing a load flow calculation with the option *Consider Voltage Dependency of Loads* (see previous Section), will result in calculated base quantities according to the type of load specified; for example,  $I_{base}$  for the constant current load and  $Z_{base}$  for the constant impedance load. If in addition to the voltage dependency of loads, the *Feeder Load Scaling* option is enabled, the calculated scaling factor  $k$  is applied according to the type of load defined in the feeder.

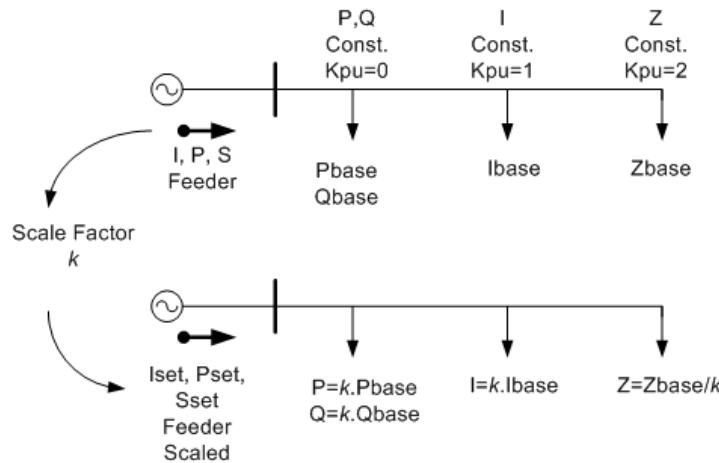


Figure 25.4.6: Feeder Load Scaling Factor Considering Different Behaviour of Loads

In *PowerFactory*, the number of *Feeder* definitions is not limited to the number of radial paths represented in the model. This means that the user can define more than one feeder element (*ElmFeeder*) along the same radial path, as indicated in Figure 25.4.7. In this particular example, both Feeder 1 and 2 have the same specified orientation ( $\rightarrow$  Branch). While Feeder 1 is defined from the beginning of the radial path, Feeder 2 is defined after load  $L_2$ . This particular type of feeder representation is termed as *Nested Feeders*. Since Feeder 1 is defined from the beginning of the radial path, every load ( $L_1, L_2, L_3$  and  $L_4$ ), as well as every feeder (Feeder 2) along this path will be considered as part of its definition. Since Feeder 2 is along the path defined for Feeder 1; Feeder 2 is nested in Feeder 1.

In such cases, executing the load flow (with the option *Feeder Load Scaling*) will treat the two feeders as independent. Although nested, Feeder 1 will only try to scale loads  $L_1$  and  $L_2$  according to its setting, while Feeder 2 will scale loads  $L_3$  and  $L_4$ . If Feeder 2 is placed *Out of Service*, then Feeder 1 will scale all the loads along the radial path ( $L_1, L_2, L_3$  and  $L_4$ ).

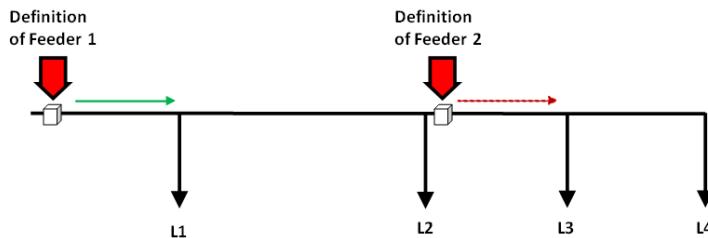


Figure 25.4.7: Nested Feeder Definition

**Note:** In order to consider a load in the feeder-load-scaling process, the option *Adjusted by Load Scaling* has to be enabled. In this case, the individual *Scaling Factor* of the load is not taken into account but overwritten by the feeder-scaling factor.

For further information on Feeder definitions refer to Chapter 15, Section 15.6 (Feeders).

Additionally, loads can be grouped in zones, areas or boundaries so the scaling factor can be easily edited. In case of zones, there will be an additional *Zone Scaling Factor*.

### Calculation process and Equations

This section explains how the scaling is carried out for the various scaling options. There are two ways of scaling: Manual scaling where a scaling factor is supplied, and scaling to setpoint, where the setpoint

is supplied and the scaling factors are calculated in order to meet this setpoint. In the equations below, the variables *scale* and *scalephi* are the quantities which are passed as signals for the load object scaling.

#### 25.4.3.1 Manual scaling, balanced Feeder Load Scaling

For balanced Feeder Load Scaling (flag *scalePhaseWise* inside *ElmFeeder* is off), manual scaling is enabled by selecting “Manually” in the drop-down menu. (*i\_scale* = 4 for the first set point, *i\_scale* = 3 for the second set-point.)

Manual scaling using the first control introduces an additional scaling factor *scale* for the specific feeder. This scaling factor *scale* is equal to the user-supplied factor *scale0*, and is applied to the apparent power of each scalable load in the feeder:

$$S_n = S_o * scale$$

where  $S_n$  is the ‘new’ apparent power of the load and  $S_o$  is the ‘old’ apparent power.

Manual scaling using the second control introduces an angle *scalephi* for the specific feeder. This scaling factor *scalephi* is equal to the user-supplied Power Factor *cospfiset*. Each scalable load inside the feeder is then set to the same power factor *scalephi*:

$$\begin{aligned} P_n &= P_o * \cos(scalephi) \\ Q_n &= Q_o * \sin(scalephi) \end{aligned}$$

where  $P_n, Q_n$  are the ‘new’ active and reactive power consumptions of the load, and  $P_o, Q_o$  are the ‘old’ active and reactive power consumptions.

#### 25.4.3.2 Manual scaling, phasewise Feeder Load Scaling

For phasewise Feeder Load Scaling (flag *scalePhaseWise* inside *ElmFeeder* is on), manual scaling is enabled by selecting “Manually” (*i\_scale* = 4) in the drop-down menu.

Only the first control may be set to Manual if phasewise scaling is being used.

The equations in this instance are complex and outside the scope of the User Manual.

#### 25.4.3.3 Scaling to setpoint: Balanced Feeder Load Scaling

For balanced Feeder Load Scaling (flag *scalePhaseWise* inside *ElmFeeder* is off), the loads inside a feeder may be scaled to meet a given setpoint:

Therefore:

- additional controls / signals are introduced: *scale* and / or *scalephi*
- additional setpoint equations are written at the feeding point

#### Feeding point equations:

When feeder set points have been specified, *scale* and *scalephi* have to be determined for each feeder so that when the loads are scaled, the specified set points are met. The requirements at the feeder points can be expressed as follows:

Given some setpoint *s* and some calculated value *v* at the feeding point (for example, *s* is some measured value of apparent power at the feeding point, and *v* is the calculated apparent power at

the feeding point), the equation reads as:

$$s = v.$$

#### *Equations first setpoint, balanced Load Flow*

Let  $cur$  be the current at the feeding point and  $u$  the voltage at the feeding point.

- Apparent power scaling:  $Sset = |u * \bar{cur}|$ .
- Current scaling:  $Iset = |cur|$ .
- Active power:  $Pset = \Re(u * \bar{cur})$ .

#### *Equations second setpoint, balanced Load Flow*

Let  $cur$  be the current at the feeding point and  $u$  the voltage at the feeding point.

- Reactive power scaling:  $Qset = \Im(u * \bar{cur})$ .
- Power factor scaling: Let  $angleset$  equal  $\text{acos}(\cos\phi_{set})$  in the case of inductive scaling, and equal  $-\text{acos}(\cos\phi_{set})$  in the case of capacitive scaling. Then the equation reads  $angleset = \text{Arg}(u * \bar{cur})$ .

#### *Equations, unbalanced Load Flow*

The equations for balanced Feeder Load Scaling with unbalanced Load Flow are similar to the equations in the case of the balanced load flow. The only difference is that the phasewise calculated values at the feeding point are summed up (sum of complex powers) in order to meet the setpoint.

#### **Load equations:**

When Feeder Load Scaling is enabled, additional controls *scale* and *scalephi* are introduced to control the power consumption of the loads.

These controls are applied to the power consumption of the loads depending on the setting *i\_scale*, *i\_scalepf* inside the feeder:

- All combinations of settings *i\_scale* and *i\_scalepf* in ElmFeeder, **except** Active Power control and Reactive power control:  
If *i\_scale* is not 0: Apparent power of the load is scaled:

$$S_n = S_o * scale \quad (25.24)$$

where  $S_n$  is the 'new' apparent power of the load, and  $S_o$  is the 'old' apparent power.

If *i\_scalepf* is not 0: Power factor of the load is adjusted to equal *scalephi*:

$$\begin{aligned} P_n &= S_o * \cos(scalephi) \\ Q_n &= S_o * \sin(scalephi), \end{aligned} \quad (25.25)$$

where  $P_n, Q_n$  are the 'new' active and reactive power consumptions of the load, and  $S_o$  is the 'old' apparent power.

- Active Power scaling and Reactive Power scaling: In this case a more involved scaling algorithm is used, in order to improve convergence. This is outside the scope of the User Manual.

#### **25.4.3.4 Scaling to setpoint: Phasewise Feeder Load Scaling**

For phasewise Feeder Load Scaling (flag *scalePhaseWise* inside ElmFeeder is on), the loads inside a feeder may be scaled per phase to meet a given setpoint:

Therefore:

- additional controls / signals are introduced: *scale1r*, *scale1s*, *scale1t* and *scale2r*, *scale2s*, *scale2t*
- additional setpoint equations are written at the feeding point

#### Feeding point equations:

The feeding point equations in this case are written per phase. They follow the same logic as for balanced Feeder Load Scaling.

#### Load equations:

In this case, a more involved scaling algorithm is used in order to support phasewise scaling. This is outside the scope of the User Manual.

---

**Note:** The phasewise load scaling is quite sensitive in terms of phase shifts. In case that a transformer (or other equipment) is located between the feeder begin and the scaled load, which causes a significant phase shift, the scaling algorithm might not find a solution.

---

#### 25.4.4 Temperature Dependency of Lines and Cables

The most important effect of the resistance of transmission line and cable conductors is the generation of losses ( $I^2R$ ). Resistance will also affect the voltage regulation of the line due to voltage drop (IR).

The resistance of a conductor is mainly affected by the operating temperature, and its variation can be considered practically linear over the normal range of operation (an increase in temperature causes an increase in resistance).

In *PowerFactory*, the load flow calculation has different options for considering the *Temperature Dependency* of resistance for lines and cables:

- **at 20° C:** When this option is selected, the load flow calculation uses the resistances (lines and cables) stated in the Basic Data page of the corresponding component (*TypLne*, *TypCon*, *TypCab*).
- **at Maximum operating temperature:** When this option is selected, the load flow calculation uses the corrected value of resistance with the maximum operating temperature of each line (Specified on the *Load Flow* page in the Type).
- **at Operating temperature:** When this option is selected, the specified individual operating temperature for each line and cable is used (specified on the *Load Flow* page of the element).
- **at Temperature:** When this option is selected a global operating temperature can be specified in the load flow command that is applied to all lines and cables.

The following equation is used to determine the temperature dependent resistance:

$$R = R_{20}[1 + \alpha(T - 20^\circ C)] \quad (25.26)$$

where,

$R_{20}$  is the resistance at temperature 20°C (Basic Data page of the corresponding type)

$\alpha$  is the temperature coefficient in  $K^{-1}$

$T$  is the operating temperature of the line. It can be the maximal operating temperature (line type), the individual operating temperature (element) or a global temperature (load flow command).

$R$  is the resistance at temperature  $T$

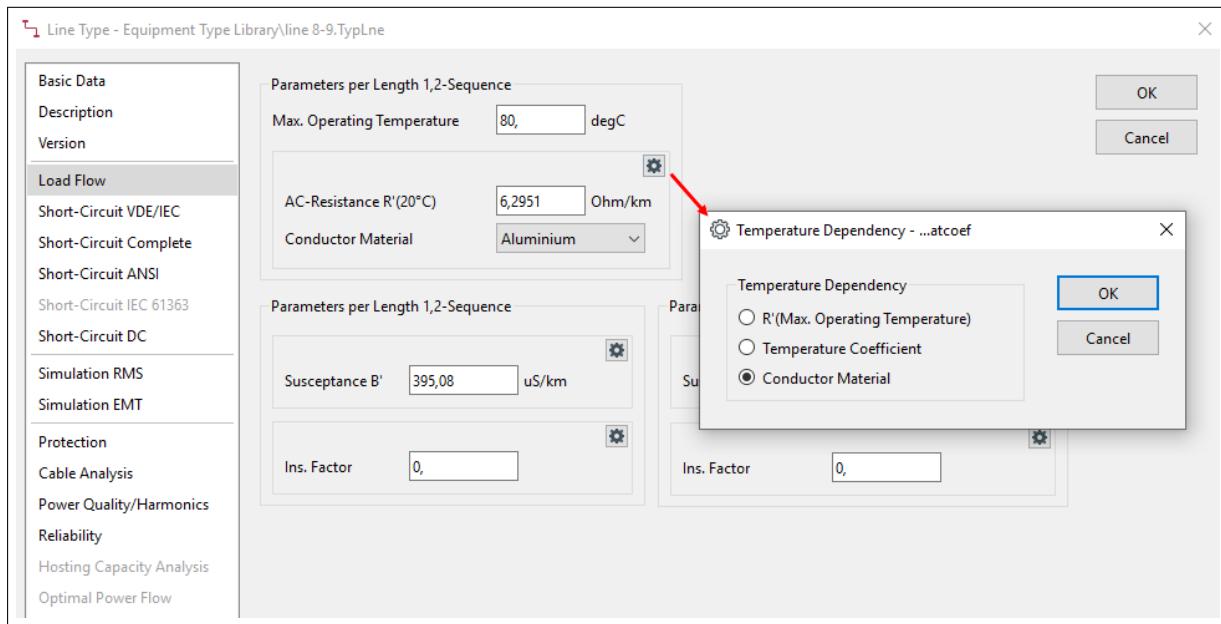


Figure 25.4.8: Temperature Dependency Option Setting in the Load Flow page of the line type (*TypLne*)

Additionally, the resistance temperature dependency can be defined by specifying either the resistance at maximum operating temperature, the temperature coefficient (1/K) or the conductor material (Aluminium, Copper or Aldrey).

Table 25.4.3 indicates the electrical resistivities and temperature coefficients of metals used in conductors and cables referred at 20°C/68°F (taken from IEC 60287-1 standard).

Material	Resistivity ( $\Omega \cdot \text{m}$ )	Temperature coefficient [ $\text{K}^{-1}$ ]
Aluminium	$2.8264 \cdot 10^{-8}$	$4.03 \cdot 10^{-3}$
Copper	$1.7241 \cdot 10^{-8}$	$3.93 \cdot 10^{-3}$

Table 25.4.3: Electrical Resistivities and Temperature coefficients of Aluminium and Copper

## 25.4.5 Load Flow initialisation and saving of results

By default, an AC load-flow calculation begins with a so-called flat start, where nodes are initialised based on the set point of the reference node and the voltage ratios along the paths through the network. There is often, however, an advantage in terms of performance if a load flow calculation can take as its starting point the results of a previous calculation.

On the Initialisation page of the Load Flow command, options are available for (a) saving load flow results for future use and (b) selecting the last available load flow results, or selecting saved results, to be used as a starting point for the calculation. The options on this page are listed in Section 25.3.4.3.

The option to *Start from last calculated results if available* can be selected for either DC or AC calculations. If the results from a successful load flow calculation are available (i.e. have not been reset), they will be used as the starting point for the next load flow calculation. It should be noted that this option will take precedence over the option to use saved results.

The option to *Start from saved results*, and associated *Save results* button are only available for AC load flow calculations. Starting from saved results is possible even when the network has changed in

the interim, though of course if the network has changed significantly, starting from saved results could make convergence harder rather than easier.

#### 25.4.5.1 Saving results to be used as a starting point in future load flow calculations

When calculation results are available, they can be saved using the **Save results** button. There are two options for saving results:

- **Save to memory:** This is the default and simplest option. Results stored in memory are available until the study case is deactivated, but only one set of results is held.
- **Save to a file:** The user can specify a file name and a location, although a default name and location are presented and can be used. The default location is the user's workspace directory. The user can overwrite existing results or provide a new file-name to enable multiple result files to be saved. The advantage of saving results into a file is that they remain available for use after *PowerFactory* has been restarted, or even if a project has been imported into a different *PowerFactory* installation.

If the *Adapt starting point settings in the Load Flow command* flag is selected, this will ensure that the Load Flow dialog is automatically configured to use these saved results as a starting point for future load flow calculations.

#### 25.4.5.2 Object Identification when saving results

When saving results to a file, the user must specify how objects are to be identified when the results are used as the starting point for future calculations, so that each result can be assigned to the correct element. Two options are offered:

- **Via object ID:** The target object for each result is identified by the unique object ID for the element. This has the advantage that it does not matter if a project has been moved around in a database since the results were saved. It is also the faster of the two options. However, it will not work if a project has been exported and reimported (because the objects will have new object IDs).
- **Via full path name:** This is slower than the first option, but has the advantage that it will still work if the project has been replaced (in the same location in the database) by another project of the same name and structure.

#### 25.4.6 Load flow calculation for train simulation

A train simulation combines a mechanical simulation of the forces (tractive effort, running resistance), acceleration/deceleration, speed and location of trains with the electrical load flow calculation of the moving trains in railway power supply networks. In the electrical load flow calculation, voltage dependencies of the power demand and power recuperation (regenerative braking) and dependencies of the power factor of trains are taken into account, along with the electrical characteristics of the railway power supply system. The results of the electrical load flow calculation have an impact on the mechanical calculation, especially if the power demanded by trains cannot be covered completely due to characteristics of the electric system.

*PowerFactory* supports the electrical part of the train simulation by a dedicated load flow calculation, in which train elements are considered, in a performance-optimised manner, to change their location and move along lines. To enable this dedicated type of load flow calculation, the option *Train simulation* has to be enabled, which is found in the *Advanced Options → Advanced* of the Load Flow Calculation command (see Section 25.3.3.4).

The electrical characteristics of the trains are defined and entered in train elements (\*.ElmTrain) and train types (\*.TypTrain). The location of a train (of a train element) within the network model during the

train simulation is defined as a connection to a line with the exact position on the line (or as a connection to a terminal node if a train is at the beginning or end of a line). The location is usually set via DPL or Python function *SetPosition*, which can be called by the mechanical simulation tool or an additional tool, which links *PowerFactory* with the mechanical simulation tool:

```
int train_object.SetPosition(object line or terminal,  
                           double position on line)
```

with “train\_object” being a train element (\*.*ElmTrain*).

Returns

- |   |   |
|---|---|
| 1 | An error occurred.                                  |
| 0 | Otherwise, i.e. train set successfully on position. |

A train simulation typically consists of a sequence of several thousand load flow calculations. Thus, any non-essential steps, for example automatically resetting the calculation results, should be avoided. To achieve optimal performance with respect to simulation speed, disabling the automatic reset of calculation results is recommended in Python scripts. Additional information on the Python function below is provided in the [Python Reference](#).

```
app.SetAutomaticCalculationResetEnabled(0)
```

---

**Note:** Usually, this is only necessary for Python scripts. The option to automatically reset calculation results is not enabled by default for DPL scripts. The respective function can be found in the [DPL Reference](#).

---

The power demand of the train model comprises traction power and auxiliary power. Generative braking (power recuperation) is supported as well. Reactive power or power factor is considered, with either constant power factor or power factor depending on the train’s active power and speed, or voltage at pantograph and speed. Several options are available to represent the voltage dependency of traction/recuperation power. As well as the voltage-dependent current limitation according to EN 50641:2020 (which includes current limitation according to EN 50388), flexible user-defined input of voltage-dependent limitation of active power or current magnitude via a look-up table, linear current reduction below a voltage threshold, and voltage-dependent power demand are supported.

The lines which represent railway tracks with overhead contact lines (catenary) or conductor rails have to be marked as *Line for train simulation* (dialog window of line element: *Basic Data* → *Advanced* → *Line for train simulation*). Single-phase AC lines (AC, AC/BI) as well as DC lines are supported for train simulation. The train positions on those lines are typically considered with a resolution of a few metres (depending on the line impedance per length, which represents the loop impedance per length of the track’s power supply lines and rails; resolution typically better than 20 m). The minimum resolution of train distances is displayed in line’s dialog window. Only if the option *Line for train simulation* together with the *Train simulation* option of the load flow calculation are selected, are trains considered with their locations on lines. Otherwise a train element can only be connected to a terminal (node), as any other network element with fixed location, or is (without connection to a terminal) neglected in load flow calculation.

A train simulation typically contains of a sequence of several thousands of load flow calculations. To achieve optimal performance with respect to simulation speed, enabling the following calculation settings is recommended:

- Load flow calculation option *Train simulation*: This option enables the possibility to place trains (train elements) on lines (single-phase line elements), as described above. The option is found in the *Advanced Options* → *Advanced* of the Load Flow Calculation command (see Section [25.3.3.4](#))

- Load Flow Method: *Newton-Raphson (Current Equations)*. This method usually provides a more robust convergence behaviour in the case of unsymmetrical networks. The load flow method is found in the *Calculation Settings* → *Algorithm* of the Load Flow Calculation command (see Section 25.3.4).
- Load flow initialisation setting *No Topology Rebuild*: This option suppresses building the topology of the power system again, when calculating the next load flow. The option is found in *Calculation Settings* → *Initialisation* of the Load Flow Calculation command (see Section 25.3.4.3).
- Load flow initialisation setting *Start from last calculated results if available*: By enabling this option, the results of the previous load flow calculation will be used as the starting point for the next load flow calculation. The option is found in *Calculation Settings* → *Initialisation* → *Starting point* of the Load Flow Calculation command (see Section 25.3.4.3).
- An *Operation Scenario* (see Chapter 16) should be active during the train simulation: This avoids permanent database updates during the simulation run, when changing operational data such as location or power of trains.

Result quantities available from the analysis include the voltage, current and resulting active and reactive power at the train pantographs, the current and power flows through individual components of the railway power supply network and equipment loading.

Coupling *PowerFactory* for calculation of the electrical quantities with an external tool that does the mechanical simulation of speed and position of the running trains based on time schedule, track layout, mechanical forces and power etc., establishes a simulation environment for closed-loop train simulation, as specified in EN 50641 (Railway applications - Fixed installations - Requirements for the validation of simulation tools used for the design of traction power supply systems). For simple train systems, e.g. small metro systems with independent train routes, implementation of the mechanical train simulation using Python or DPL scripts within *PowerFactory* may be sufficient and suitable as well.

## 25.5 Results Analysis

In *PowerFactory* the results can be displayed directly in the single line diagram, in tabular form or by using predefined report formats. Also available are several diagram colouring options in order to have a “quick” overview of the results.

### 25.5.1 Viewing Results in the Single Line Diagram

Once a load flow calculation has been successfully executed, the result boxes shown in the single-line diagram will be populated. There is a result box associated with each “side” of an element. So for example a load has one result box, a line two result boxes, and a three-winding transformer three result boxes. In *PowerFactory* these elements are collectively called edge elements. In addition, there are result boxes for nodes or buses.

The information shown inside a result box depends on the element to which it is associated. Several predefined formats can be selected, as described in Chapter 10: Network Graphics, Section 10.5. Result boxes can also be personalised as described in Chapter 19: Reporting and Visualising Results, Section 19.2.

### 25.5.2 Flexible Data Page

Once a load flow calculation has been successfully executed, pressing the *Open Network Model Manager...* button ( ) located on the main menu will open a browser window with a list of all classes on the left side of the window that are currently used in the calculation. Clicking any of the classes will

display all elements of that class that are currently used in the calculation in a table on the right side of the window. The left-most tab-page at the bottom of the browser is the Flexible Data tab page. Click on this tab page to show the flexible data. To change the columns in the flexible page, press the *Define Flexible Data* button (grid icon). This will bring a selection window where the set of variables can be edited. Section 19.3 in Chapter 19: Reporting and Visualising Results, describes the *Variable Selection* object which is used to define the variables to be presented.

### 25.5.3 Standard Reports

In addition to viewing the results on a diagram or in the Network Model Manager, the user can generate one or more reports, using the Report Generation (*ComReport*) command, which is accessed via the  icon on the main toolbar. Most of these are presented by default as PDF reports, using *PowerFactory*'s inbuilt PDF Viewer, although some "legacy" reports appear as tabular reports or as ASCII output in the output window. The Report Generation command will present all report options that are applicable to the *Load Flow Analysis* command that has been executed, and the selected reports can be generated as separate documents or combined into one. There is also the option to export reports from *PowerFactory* in various different formats.

For more information about reports and the Report Generation command, see Chapter 19: Reporting and Visualising Results, Section 19.5. It should be noted that reports which are viewed internally in *PowerFactory* are stored in the desktop of the study case, even after they have been closed, until they are actively deleted by the user.

A *Verification Report* can also be generated when the command is executed (see Section 51.4.6). This appears as an ASCII report in the output window.

### 25.5.4 Diagram Colouring

When performing load flow calculations, it is very useful to colour the single line-diagram in order to have a quick overview of the results, for example if elements have a loading above 90 % or if the voltages of the busbars are outside the specified limits. In *PowerFactory* there is the option of selecting different colouring modes according to the calculation performed.

The use of colouring in network diagrams is described in Section 10.3.10.1, and more detail about the use of colours and colour palettes can be found in Section 4.7.8.

If a specific calculation is valid, then the selected colouring for this calculation is displayed. As an example, if the user selects the colouring mode *Zones* for *No Calculation* and *Low and High Voltage/Loadings* for the load flow calculation, then the initial colouring will be according to *Zones*. However, as soon as the load flow is calculated, the diagram will be coloured according to *Low and High Voltage/Loadings*. If the load flow calculation is reset or invalid, the colouring mode switches back to *Zones*.

The Diagram Colouring has also a 3-priority level colouring scheme also implemented, allowing colouring elements according to the following criteria: 1<sup>st</sup> Energising status, 2<sup>nd</sup> Alarm and 3<sup>rd</sup> "Normal" (Other) colouring.

#### **Energising Status**

If this check box is enabled "De-energised" or "Out of Calculation" elements are coloured according to the settings in the "Project Colour Settings". The settings of the "De-energised" or "Out of Calculation" mode can be edited by clicking on the *Colour Settings* button.

### **Alarm**

If this check box is enabled a drop down list containing alarm modes will be available. It is important to note here that only alarm modes available for the current calculation page will be listed. If an alarm mode is selected, elements “exceeding” the corresponding limit are coloured. Limits and colours can be defined by clicking on the *Colour Settings* button.

### **“Normal” (Other) Colouring**

Here, two lists are displayed. The first list will contain all available colouring modes. The second list will contain all sub modes of the selected colouring mode. The settings of the different colouring modes can be edited by clicking on the *Colour Settings* button.

Every element can be coloured by one of the three previous criteria. Also, every criterion is optional and will be skipped if disabled. Regarding the priority, if the user enables all three criterions, the hierarchy taken into account will be the following:

“Energising Status” overrules the “Alarm” and “Normal Colouring” mode. The “Alarm” mode overrules the “Normal Colouring” mode.

## **25.5.5 Load Flow Sign Convention**

By default, *PowerFactory* has the following load flow sign convention (Mixed Mode):

### **Branches:**

Power Flow going out of the Busbar is positive while going into the busbar is negative.

### **Loads:**

Power Flow going out of the Busbar is positive while going into the busbar is negative. Here, the term load considers “General Loads”, “Low-Voltage Loads”, “Motors”, “Shunts/Filters” and “SVS”. A synchronous machine stated as a “Motor” will have also this sign convention.

### **Generation:**

Power Flow going out of the Busbar is negative while going into the busbar is positive. Here, the term Generation considers “Generators”, “External Grids”, “Static Generators” and “Current and Voltage Sources”. An asynchronous machine stated as a “Generator” will have also this sign convention.

## **25.5.6 Results for Unbalanced Load Flow Calculations**

One of *PowerFactory*’s strengths lays in the modelling and simulation of asymmetric networks with individual phases (also neutrals). The unbalanced load flow calculation leads to phase/conductor specific results. In addition unbalance factors for voltage, current and power are calculated.

The percentage unbalance factor (for voltage) is calculated for node elements as ratio of the negative and positive sequence voltage. The current unbalance factor is calculated similarly with negative and positive sequence currents. It is evaluated for each side of all branch elements, where in addition the power unbalance factor is calculated. It is defined as follows:

Let  $N$  be the number of phases, and let

$$\hat{S} = \frac{1}{N} \sum_{i=1}^N S_i$$

be the average complex power (at one end) of a branch element, where  $S_i, i = 1, \dots, N$  are the complex powers on phases  $1, \dots, N$ . Let

$$\bar{S} = \frac{1}{N} \sum_{i=1}^N |S_i|$$

be the average of the absolute values of the powers on the different phases. Then the power unbalance factor  $s_b$  for the branch element  $b$  is defined as

$$s_b := (1/\bar{S}) \max_{i=1, \dots, N} \{|S_i - \bar{S}| \}.$$

---

**Note:** The power unbalance factor is calculated for every side of all branch elements. Due to its definition, the voltage and current unbalance factor however is only calculated for three phase elements, where the transformation into symmetrical components is possible.

---

For feeder elements, the unbalance factors of all assigned elements are analysed, the maximum and average values identified and stored into dedicated variables. In addition the unbalance factors of the feeding point (the branch element, from which the feeder starts) are available.

### 25.5.7 Update Database

Input (data that has been entered by the user) and output (parameters that have been calculated) data is kept separate in *PowerFactory*. Output data, such as the new tap positions following an automatic tap adjustment calculation, does not overwrite the settings that are originally entered, unless the *Update Database* (*ComDbupd*)  command on the main toolbar is executed.

---

**Note:** The corresponding input parameters of the database will be overwritten by the calculated values.

---

#### Network components

- **Update:** apply the changes to either the whole system, the set from a custom selection (*SetSelect*) or filtered elements (*SetFilt*).

#### Apply update

This section contains the selection of elements and the respective attributes, that are updated by the command.

- **Transformers -Taps:** update either the tap position of all transformers, only phase shifting or only not phase shifting transformers.
- **Loads:** update either P and Q setpoints, the scale factor or all three values of the loads (*ElmLod*).
- **MV Loads - Distribution Transformer Taps:** update the tap position of the distribution transformers of MV load elements (*ElmLodmv*).
- **Shunts/Filters - Steps:** update the step position of Shunts and Filters (*ElmShnt*).
- **SVSs - TCR and Capacitive Steps:** update the TCR and the capacitive step position of SVSs (*ElmSvs*).
- **Asynchronous Machines, DFIGs:** update any combination of the P, Q and V setpoints of Asynchronous Machines (*ElmAsm*) and DFIGs (*ElmAsmsc*).

- **Synchronous Machines:** update any combination of the P, Q and V setpoints of synchronous machines (*ElmSym*).
- **Static Generators:** update any combination of the P, Q and V setpoints of static generators (*ElmGenstat*).
- **External Grids:** update any combination of the P, Q and V setpoints of external grids (*ElmXnet*).
- **Extended-/Ward Equivalents:** update any combination of the P and Q setpoints of (Extended) Ward Equivalents (*ElmVac*).
- **PWM Converters:** update any combination of the P and Q setpoints of PWM converters (e.g. *ElmVsc*).

#### Update reactive power (DC load flow or for option 'only P' or 'P and V')

This section is only available if a DC load flow is carried out and the 'only P' or the 'P and V' option is selected for any of the applicable elements mentioned above.

- **do not update:** the reactive power is not adjusted and the power factor of the element can change.
- **according to const. power factor:** the reactive power of the respective elements is adjusted to retain the previous constant power factor setpoint.

#### Example:

A Load Flow is calculated with the options *Automatic tap adjustment of transformers* and *Automatic tap adjustment of shunts* enabled. The calculated tap and shunt positions can be seen in the single line diagram, but the input data parameters in the element data dialog remain as originally entered. If the *Update Database* command  is carried out, the input parameters are now overwritten by the calculated values found on the single line diagram.

## 25.6 Troubleshooting Load Flow Calculation Problems

In general, if a solution can be found (in other words, the network is mathematically solvable), *PowerFactory* will find a solution. In some cases the user may have made an error which will not allow a solution to be found; such as a large load causing a voltage drop so large that a voltage collapse results. In a real-world power system the same problem would be found.

When creating a network for the first time it is best to enter the data for only a small part or 'path' of the network and solve the network by calculating a load flow. *PowerFactory* has a data verification process in which certain checks are performed, such as whether a line is connected between nodes of the same voltage; and the correct voltage orientation of transformers, etc.

Typical reasons for non-convergence in the load flow are:

- Data model problem.
- Too many inner loop iterations.
- Too many outer loop iterations.
- Excessive mismatch.
- Tap hunting.

Clearly this is not an exhaustive list of problems, but these are the main causes of non-convergence and that will be discussed in this section.

### 25.6.1 General Troubleshooting

The place to search for the causes of the non-convergence problem is in the *PowerFactory* output window. Here, there can be three different types of messages printed out, which are the following:

#### Info messages

Information detailing the load flow convergence (inner and outer loop iterations). Information of generators with reactive power compensation at output limit. Information on the total number of isolated areas (see [25.6.3](#)).

#### Warning messages

Warning messages do not need to be corrected for the load flow to solve, however they could give you an indication of where the problem is. Take note of the warning messages and evaluate them in terms of your system. Important warnings, such as “Exceeding Mvar limit range” may not be acceptable. “Unsupplied Areas” messages indicate that an isolated area with “Consumers” (such as loads and motors) is without a generator, power source or external supply.

#### Error messages

Error messages must be corrected for a load flow to solve. Error messages could be generated by *PowerFactory*'s data checking function, which include messages such as `missing type!`. In most cases the messages have links to the data base and graphic. The following options can be performed in order to trace errors:

- Use the *Network Data Assessment* tool ().
- Once errors have been detected, open the problematic element dialog window by double clicking on the name directly from the output window. Or alternatively, right-click mouse button over the name and select *Edit*, or *Show in Data Manager*, or *Mark in Graphic*.

The amount of information being printed to the *PowerFactory* output window can be changed by the user. Once error messages have been analysed and corrected and the load flow still does not solve, the user may want to print more detailed information on the convergence progress.

Tick the *Show Convergence Progress Report* option found in the Outputs page of the load flow dialog (refer to Section [51.4.6](#)).

This will print messages to the output window that can provide clues as to where the convergence problems may lie.

The single line graphic can also be coloured to show low and high voltages and overloadings. This will also provide a good indication of possible problems. Look at the undervoltage nodes and overloaded elements and investigate why they are overloaded; look at load setpoints, line lengths and line type data (the impedances may be too high, for example).

---

**Note:** As explained above, there are 3 different types of messages that are printed to the output window: warning, error and information messages. Only error messages must be corrected for a load flow to solve. Take note of the warning messages and evaluate them in terms of your system, however these do not need to be corrected for the load flow to solve. “Unsupplied Areas” means that an isolated area with “Consumers” is without a generator, power source or external supply.

---

If there is still no convergence then set the option *Out of Service* for most of the elements (see each elements *Basic Data* tab). Following this, bring these elements back into service, one at a time, from the source element *downwards*, performing a load flow calculation each time.

When experiencing large unbalances, such as when there are a number of single or dual phase elements, or when using power electronics elements, select the *Newton-Raphson (Current Iteration)* option on the *Advanced* page of the load flow dialog.

## 25.6.2 Data Model Problem

In *PowerFactory*, there are three different levels of data verification implemented:

### Parameter Level

Checks the consistency of the input parameter; for example, entering a negative value in the length of the line will prompt an error message. Other verifications implemented include checking if the parameter imputed is within certain limits.

### Object Level

Checks the consistency of the data being imputed from the component itself; for example, checking if the magnetising losses of a transformers are less than the total magnetising apparent power (i.e. magnetising current), checking if the inputting of the manufacture's data results in a feasible torque-slip characteristic, etc.

### System Level

Checks the consistency of the data being imputed from a system point of view; for example, checking if lines/cables are connected between the same voltage levels, checking if the HV/MV/LV side of transformers is compatible with the voltage level of busbars, checking if there are missing types, etc.

Data model problems can normally be fixed easily as the output window message refers directly to the element causing the problem.

Typical cases of data model problems are:

- `missing type!`: it indicates that input data (electrical data defined in types) is missing. In most cases the messages have links to the data base and graphic.
- `Check control conditions!`: it normally appears when more than one controller (for example a station controller) is set to control the same element, such as the same busbar. *PowerFactory* will print the name of the controlled element to the output window. Starting from the controlled element, access the controllers to fix the problem.
- `Line connected between different voltage levels!`

## 25.6.3 Some Load Flow Calculation Messages

- Grid split into 182 isolated areas

An “isolated area” indicates that a busbar or a group of busbars are not connected to the slack busbar. An isolated generator or an isolated external grid forms an isolated area. An isolated area refers basically to nodes.

Each isolated area is assigned an index (Parameter name `b:ipat`) and needs a load flow reference (slack) of its own.

These busbars can be found by colouring the single line graphic according to isolated grids.

- `2 area(s) are unsupplied`

An “unsupplied area” is an isolated area with “Consumers” (such as loads and motors) without a generator, power source or external supply. That is  $U=0$  and  $I=0$ . Unsupplied areas belong to

the group of isolated areas. The unsupplied areas can be identified by displaying the following parameter in the “Consumers” components (loads, synchronous/asynchronous motors):

- `r:bus1b:ipat` Gives the Index of the isolated area
  - `r:bus1:b:imode=0` Indicates there is no slack in the isolated area therefore indicating its unsupplied.
  - `r:bus1:b:imode>0` Indicates the area is supplied.
- Outer loop did not converge. Maximum number of iterations reached  
For some hints on this type of error refer to Section [25.6.5](#).

## 25.6.4 Too many Inner Loop Iterations

Too many inner loop iterations are “normally” related to voltage stability (voltage collapse) problems. For example, a large load causing voltage drops so high that a voltage collapse results. Also very weak connections resulting from faults or outages may lead to voltage collapse during contingency analysis.

The problem will not only be found in the simulation but would be found in the real world as well!

The main causes leading to a voltage stability problem can be summarised as follows:

- Excessive active power demand leading to a high voltage drop.
- Lack of reactive power compensation.

### Diagnosis and Solution:

The main source of Information is the output window. Enable the *Show Convergence Progress Report* option found in the *Outputs* page of the load-flow dialog. Analyse the convergence of the inner loop iterations: check the progress in the load flow error for nodes and model equations:

- Are they increasing or decreasing?
- If the error is not continuously decreasing, it could be an indication of a voltage stability problem.
- Identify the element (load, generator) with high convergence error. Use the *Mark in Graphic* option to identify the zone of the network having the problem.

Several possible countermeasures can be undertaken to fix the problem:

- Use the *Iteration Control* options on the load flow command (increasing the number of stairs as the first option, typically to 3).
- Load shedding: disconnect the load identified as responsible for the high convergence error.
- Connect additional reactive power compensation.
- Using the flexible data page, check if there are any heavily loaded circuits, these indicate weak connections.

Once the load flow converges, check if there are areas with voltages with high deviation from operating voltages.

### 25.6.4.1 Excessive Mismatch

Where there is a large mismatch between demand and generation ( $> 15\%$ ) the load flow is unlikely to converge. This is typified by a large number of iterations followed by warnings or errors such as:

No convergence in load flow! Equation system could not be solved.  
Check Control Conditions!

Depending on the size of the mismatch, the failure might occur during the initial Newton-Raphson or during subsequent outer loop iteration. There may also be a large number of maximum/minimum reactive power reached and transformer tap statements.

**Solution:**

- Set the option *Show Convergence Progress Report* on the *Outputs* page and observe which elements are having the highest mismatches. These elements should be closely checked.
- Check the mismatch on the Reference machine by performing a DC load flow as *Dispatched* allowing for normal losses. Rebalancing the network might alleviate convergence problems.

### 25.6.5 Too Many Outer Loop Iterations

Outer loops iterations are required to calculate discrete tap positions of transformers, number of steps of switchable reactive power compensation, etc. in order to match the voltage profile or reactive power control specified by the user.

Too many outer loop iterations is referring to a solution that is too far away from the starting point (default tap positions) to converge in the allowed number of outer loop iterations.

**Diagnosis and Solution:**

The outer-loop does the following:

- Increasing/Decreasing discrete taps.
- Increasing/Decreasing switchable shunts.
- Limiting/Releasing synchronous machines to/from max/min reactive power limits.

If the outer loop does not converge, it can have the following reasons:

- Tap upper and lower limits are too close, so that the voltage can never be kept in the desired range.
- The same with switchable shunts.
- Other toggling effects, for example synchronous machine limits and tap positions don't find a stable solution.

The main source of Information is the output window. Check first the following:

- Is the number of messages reducing with each outer loop iteration?

The following messages in the output window may indicate a problem and lead to a non-convergent solution.

```
-----  
'\....\Transformer.ElmTr2':  
Maximum Tap Position reached  
-----
```

The message indicates that more/less reactive power is required at this location (the tap is at maximum/minimum position). The message indicates therefore an area in the network where a lack/excess of reactive power is likely to happen.

```
'\....\Generator.ElmSym':  
Reactive Power Limit left  
-----
```

This will lead to a convergence error. A load flow calculation without considering reactive power limits may find a solution. Check then required reactive power at the generator.

```
-----  
\....\Generator.ElmSym':  
Maximum Reactive Power Reached  
-----
```

Basically means that there is no regulation margin in the specified generators.

In general the results from the last iteration should be available to view on the output window.

- Is the mismatch always in the same (or similar) location?
- How far away from the solution was the original starting point?

All actions (except for shunt switching) are displayed in the output window by blue messages. Observing these messages allows to conclude what the reason for the convergence problem was, for example if a synchronous machine toggles between limited/released, the problem is related to this particular machine.

- If no toggling can be observed, increasing the maximum number of outer iteration loops may help.
- If the load flow converges, improve the convergence of subsequent calculations by saving the tap positions (.

If the load flow does not converge after a large number of iterations then other methods of improving convergence are:

- Use the *direct* method on the advanced options page of the load flow command.
- Set the maximum tap changes per iteration to be a small number, for example 1. This will result in *PowerFactory* not changing all tap changers at once by several steps but only by maximum of 1 step at once. In larger networks this is often improving the convergence.
- Perform a load flow without *automatic taps and shunt adjustment*. If the load flow does not converge in this case, it could be an indication that the load is exceeding the voltage stability limits, thus the load is too high.

### 25.6.5.1 Tap Hunting

Tap hunting can be easily recognised when one or more transformers oscillate between tap positions until the number of outer loop iterations is exhausted. This is normally due to the transformer (controller) target voltage dead band being smaller than the transformer tap step size.

This problem of no converging load-flow with the *stepped* tap changing method is caused by a slightly different way of the iteration to reach the correct tap position and load-flow results. This might result in a non-convergence in the outer loop, when the controller range ( $V_{max}-V_{min}$ ) of the tap changer is near to the value of the additional voltage per tap.

#### Solution:

- Change the minimum relaxation factor on the Advanced Options page of the load flow command to a smaller value. This might help the load flow to converge.

- Check if the dead bands of the target or controlled busbars of the corresponding transformers are correctly set. Also check if the tap changer data on the load flow page of the transformer type is correct.
- Disable the automatic tap changing of the transformers where tap hunting occur. Run the load flow (it should converge in this case!) and then check the sensitivity of the tap changer increasing and decreasing the tap position by one step. Verify the results against the dead band of the target busbar.

# Chapter 26

## Short-Circuit Analysis

### 26.1 Introduction

Power systems as well as industrial systems are designed so that loads are supplied safely and reliably. One of the major aspects taken into account in the design and operation of electrical systems is the adequate handling of short-circuits. Although systems are designed to stay as free from short circuits as possible, they can still occur. A short-circuit condition generally causes large uncontrollable current flows, which if not properly detected and handled can result in equipment damage, the interruption of large areas (instead of only the faulted section) as well as placing personnel at risk. A well-designed system should therefore isolate the short-circuit safely with minimal equipment damage and system interruption. Typical causes of short-circuits can be the following:

- Lightning discharge in exposed equipment such as transmission lines.
- Premature ageing of the insulation due mainly to permanent overloading, inappropriate ventilation, etc.
- Atmospheric or industrial salt “Build-Up” in isolators.
- Equipment failure.
- Inappropriate system operation.

One of the many applications of a short-circuit calculation is to check the ratings of network equipment during the planning stage. In this case, the planner is interested in obtaining the maximum expected currents (in order to dimension equipment properly) and the minimum expected currents (to aid the design of the protection scheme). Short-circuit calculations performed at the planning stage commonly use calculation methods that require less detailed network modelling (such as methods which do not require load information) and which will apply extreme-case estimations. Examples of these methods include the IEC 60909/VDE 0102 method [27], the ANSI method and the IEC 61363 method [9] for AC short circuit calculation and the IEC 61660 method [8] and ANSI/IEEE 946 method [32] for DC short circuit calculation. A different field of application is the precise evaluation of the fault current in a specific situation, such as to find out whether the malfunction of a protection device was due to a relay failure or due to the consequence of improper settings (for example an operational error). These are the typical applications of exact methods such as the superposition method (also known as the *Complete Method*), which is based on a specific network operating point.

Using the *Complete Method* it is possible to calculate results which take account of the procedures outlined in Issue 1 (1992) and Issue 2 (July 2020) of the ENA Engineering Recommendation G74.

The short-circuit calculation in *PowerFactory* is able to simulate single faults as well as multiple faults of almost unlimited complexity. As short-circuit calculations can be used for a variety of purposes, *PowerFactory* supports different representations and calculation methods for the analysis of short-circuit currents.

This chapter presents the handling of the short-circuit calculation methods as implemented in *PowerFactory*. Further background on this topic can be found in Section [26.2](#).

## 26.2 Technical Background

Beside load flow calculations, short-circuit is one of the most frequently performed calculations when dealing with electrical networks. It is used both in system planning and system operation.

Typical application examples of short-circuit analysis in system planning include:

- Ensuring that the defined short-circuit capacity of equipment is not exceeded with system expansion and system strengthening.
- Co-ordination of protective equipment (fuses, over-current and distance relays).
- Dimensioning of earth grounding systems.
- Verification of sufficient fault level capacities at load points (e.g. uneven loads such as arc furnaces, thyristor-driven variable speed drives or dispersed generation).
- Verification of admissible thermal limits of cables and transmission lines.

Example applications of short-circuit analysis in system operation include:

- Ensuring that short-circuit limits are not exceeded with system reconfiguration.
- Determining protective relay settings as well as fuse sizing.
- Calculation of fault location for protective relays, which store fault disturbance recordings.
- Analysis of system faults, e.g. misoperation of protection equipment.
- Analysis of possible mutual interference of parallel lines during system faults.

AC short circuit calculation quantities available in *PowerFactory* are shown in Figure [26.2.1](#), also a graphical representation of the AC short-circuit current time function is illustrated in Figure [26.2.2](#). Note that the quantities relating to the IEC 61363 standard [9] and DC short-circuit quantities calculated in DC short circuit standards IEC 61660 and ANSI/IEEE 946 are not shown in Figure [26.2.1](#).

---

**Note:** The current waveform for a DC short circuit calculation is dependent on the type of DC current source(s), for more information refer to Section [26.2.5](#) and Section [26.2.6](#) and the relevant IEC and ANSI/IEEE standards.

---

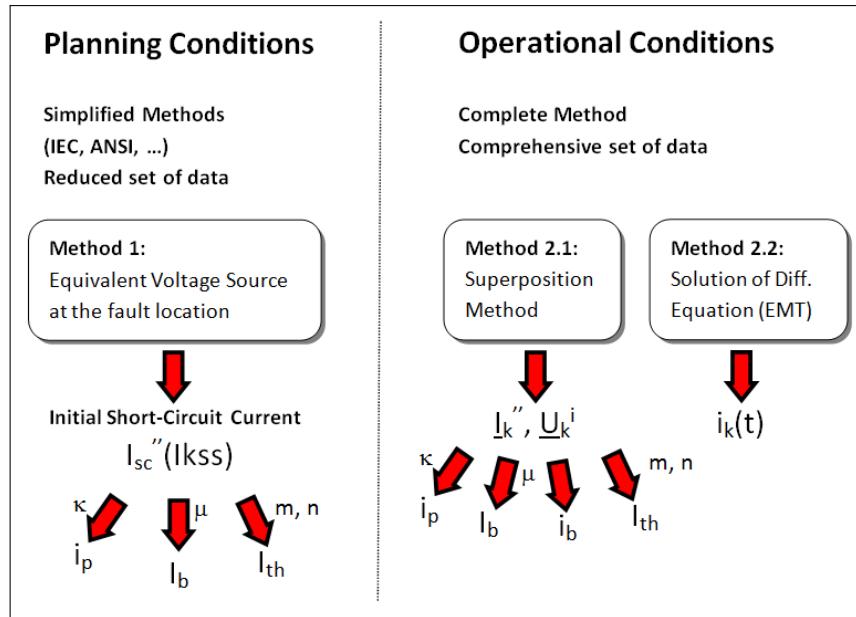


Figure 26.2.1: Areas of Application of Short-Circuit Calculations

According to IEC 60909 [27] the definition of the currents and multiplication factors shown in Figure 26.2.1 are as follows:

- $I_{kss}$  initial symmetrical short-circuit current (RMS),
- $i_p$  peak short-circuit current (instantaneous value),
- $I_b$  symmetrical short-circuit breaking current (RMS),
- $I_{th}$  thermal equivalent short-circuit current (RMS),
- $\kappa$  factor for the calculation of the peak short-circuit current,
- $\mu$  factor for the calculation of the symmetrical short-circuit breaking current,
- $m$  factor for the heat effect of the d.c. component,
- $n$  factor for the heat effect of the a.c. component, besides the above currents, the *Complete Method* introduces the following current definition:
- $i_b$  peak short-circuit breaking current (instantaneous value).

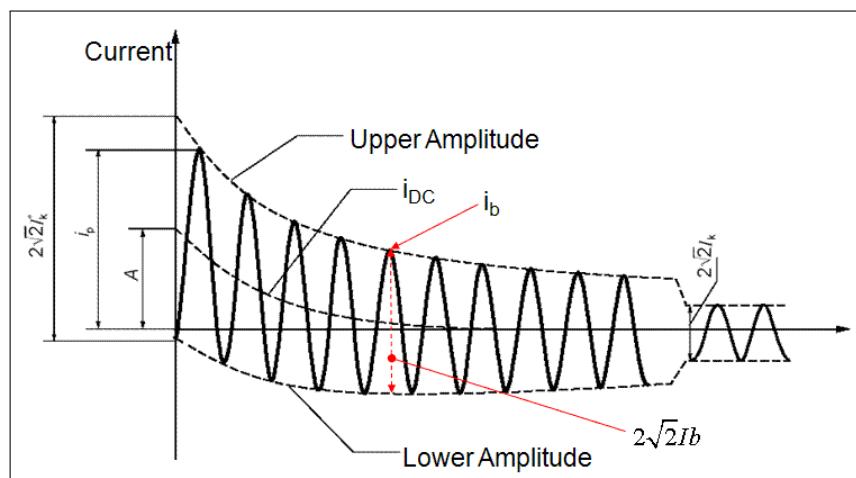


Figure 26.2.2: Short-Circuit Current Time Function

The fundamental difference between the assumptions used by the calculation methods is that for system planning studies the system operating conditions are not yet known, and therefore estimations are necessary. To this end, the IEC (and VDE) methods which use an equivalent voltage source at the fault location have become generally accepted in countries using IEC based standards. For AC fault calculation, the IEC 60909 [27] (and VDE 0102) methods work independently of the load flow (operating point) of a system. The methods are based on the nominal and/or calculated dimensions of the operating point of a system and uses correction factors for voltages and impedances, to give conservative results. For the calculation of minimum and maximum short-circuit currents, different correction factors are applied. However, it should be mentioned that both IEC 60909 and VDE 0102 do not deal with single phase elements (except single phase elements in the neutral conductor).

Another very similar method for AC fault calculation is the ANSI method, predominately used in North America but accepted in other countries as well. The ANSI method is based on the IEEE Standards C37.010 [1] which is for equipment applied in medium and high voltage systems (greater than 1000 Volts) and C37.13 [4] which is for power circuit breakers in low voltage systems (less than 1000 Volts).

For AC short-circuit calculations in a system operation environment, the exact network operating conditions are well-known. If the accuracy of the calculation according to approximation methods such as IEC 60909 [27] is insufficient - or to verify the results of these methods - the superposition method can be used. The superposition method calculates the expected short-circuit currents in the network based on the existing network operating condition. If the system models are correct, the results from this method are always more exact than the results of the approximation method (such as IEC 60909). Often the system analyst must, however, ensure that the most unfavourable conditions are considered with respect to the sizing of plant. This may require extensive studies when using a superposition calculation method.

Other short circuit calculation methods available in *PowerFactory* include:

- IEC 61363 [9]: Calculation of short-circuit currents on marine or offshore electrical systems such as ships.
- IEC 61660 [8]: IEC standard for the calculation of short-circuit currents in DC auxiliary systems in power plants and substations.
- ANSI/IEEE 946 [32]: ANSI/IEEE standard for the calculation of short-circuit currents in DC power Systems for Stationary Applications.

### 26.2.1 The IEC 60909/VDE 0102 Part 0/DIN EN 60909-0 Method

The IEC 60909/VDE 0102 part 0/DIN EN 60909-0 [27] calculation is a simplified short-circuit calculation method which doesn't require accurate system loading information to be present in the network model in order to produce useful results. Unlike the complete method (see Section 26.2.3), a preceding load-flow calculation is not required. Instead, the system nominal voltage at the point of fault is increased or decreased by a specific safety factor (the so called c factor) with the intention of producing either conservatively high or conservatively low results according to the user's specific requirement. Since loading information is not considered, this method is particularly useful in the design stage of a project where detailed operational data for the network will not yet be available. Where detailed operational data for the network is available the complete method should produce more accurate results and may be more appropriate.

---

**Note:** IEC 60909 [27], VDE 0102 and DIN EN 60909-0 are intended for usage with three phase networks only.

---

The equivalent voltage source at the faulted bus is illustrated in Figure 26.2.3. The equivalent voltage source method can be derived from the superposition method. The main simplifications are as follows:

- Nominal conditions are assumed for the whole network, i.e.  $U_i = U_{n,i}$
- Load currents are neglected, i.e.  $I_{Op} = 0$ .
- A simplified simulation network is used, i.e. loads are not considered in the positive and negative sequence network.
- To ensure that the results are conservatively estimated, a correction factor,  $c$ , is applied to the voltage at the faulted busbar. This factor differs for the calculation of the maximum and the minimum short-circuit currents of the network.

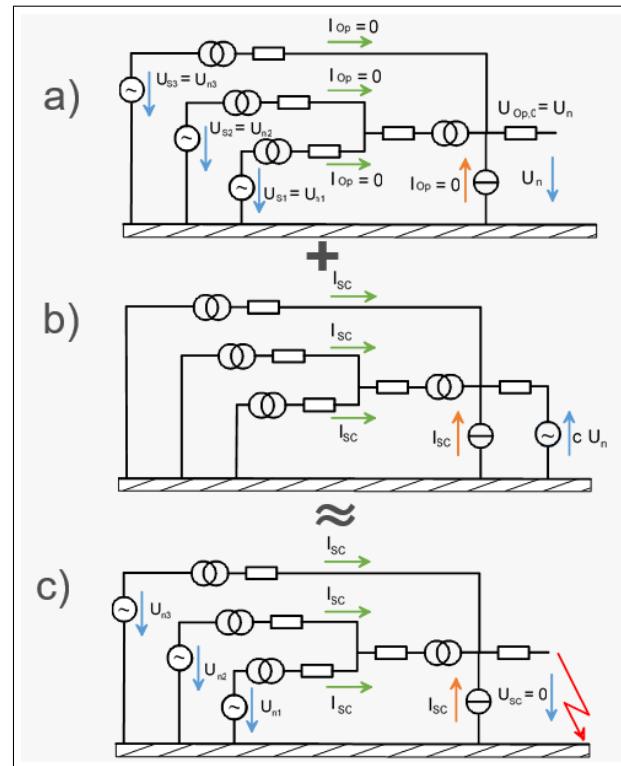


Figure 26.2.3: Illustration of the IEC 60909/VDE 0102 Method

### IEC Impedance Correction Factors

The short-circuit calculation based on the above simplifications may be insufficient for some practical applications. Therefore, the standards determine that in some specific circumstances additional impedance correction factors should be applied to the physical impedances of the network elements.

As an example consider that the short-circuit contribution of a synchronous generator depends heavily on the excitation voltage and on the unit transformer tap changer position. The worst-case value of this impedance is considered by applying a correction factor ( $< 1$ ).

This idea is illustrated in Figure 26.2.4. The correction factor  $c$  should be determined so that  $I''_k = I''_{k,IEC}$ . The IEC 60909 standard defines an equation for the correction factor for each element type.

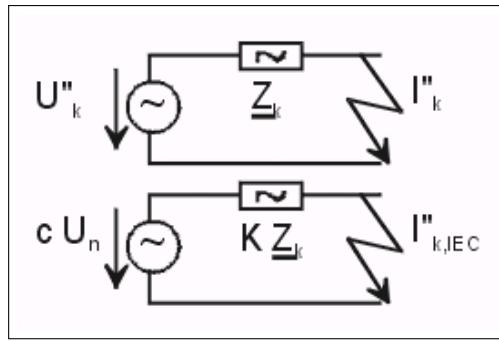


Figure 26.2.4: Principle of Impedance Correction (IEC/VDE Method)

As illustrated in Figure 26.2.1, IEC 60909 requires the calculation of the initial symmetrical short circuit current in order to derive the rest of the physical quantities, each of which is a function of the following:

- R/X ratio,
- Machine characteristics
- Synchronous generator type of excitation system,
- Contact parting time,
- Type of network (if it's radial or meshed),
- Determination if the contribution is “near to” or “far from” the short-circuit location,

Regarding the type of network, IEC 60909 describes three methods for the calculation of (peak short-circuit current) in meshed networks which are defined as follows:

**Method A: Uniform Ratio R/X** The  $\kappa$  factor is determined based on the smallest ratio of R/X of all the branches contributing to the short-circuit current.

**Method B: Ratio R/X at the Short-Circuit Location** For this method the  $\kappa$  factor is multiplied by 1.15 to cover inaccuracies caused by using the ratio R/X from a network reduction with complex impedances.

**Method C: Equivalent Frequency** An equivalent impedance  $Z_c$  of the system as seen from the short-circuit location is calculated assuming a frequency  $f_c = 20\text{Hz}$  (for a nominal frequency  $f_c = 50\text{Hz}$ ) or  $f_c = 24\text{Hz}$  (for a nominal frequency  $f_c = 60\text{Hz}$ ). This is the recommended Method in meshed networks.

---

**Note:** In *PowerFactory* methods B and C are available to the user. Method C is the one recommended in meshed networks. For more information refer to Section 26.4.3

---

As the IEC 60909 standard includes a worst-case estimation for minimum and maximum short-circuit currents, some *PowerFactory* elements require additional data. These elements are:

**Lines** In their type (*TypLne*), the maximum admissible conductor temperature (for minimum short-circuit currents) must be stated. Line capacitances are not considered in the positive/negative sequence systems, but must be used in the zero-sequence system.

**Transformers** Require a flag indicating whether they are unit or network transformers. Network transformers may be assigned additional information about operational limits which are used for a more precise calculation of the impedance correction factor. Unit transformers are treated differently depending on whether they have an on-load or a no-load tap changer defined in the transformer type (*TypTr2*).

**Synchronous Machines** Subtransient impedances are used. Additionally, information regarding the voltage range must be given.

**Asynchronous Machines** The ratio of starting current to rated current is used to determine the short-circuit impedance.

For calculations according to IEC 60909 version 2016 and VDE 0102 version 2016, which additionally include the handling of doubly-fed induction generators and full size converters, the following parameters are included as well.

**Doubly Fed Induction Generator**  $\kappa_{WD}$  can be entered, which is used to calculate the peak short-circuit current.

**Doubly Fed Induction Generator**  $i_{WDmax}$  is the maximum instantaneous short-circuit current

**Doubly Fed Induction Generator** ratio  $R_{WD}/X_{WD}$

**Full Size Converter** is modelled by a current source in the positive sequence. The current that depends on the type of short circuit can be entered.

Refer to the IEC 60909 [27] standard to find detailed information regarding specific equipment models and correction factors for each element.

## 26.2.2 The ANSI Method

ANSI provides the procedures for calculating short-circuit currents in the following standards:

- **ANSI/IEEE Standard C37.010** [1], IEEE Application Guide for AC High-Voltage Circuit Breakers Rated on a Symmetrical Current Basis.
- **ANSI/IEEE Standard C37.13** [4], IEEE Standard for Low-Voltage AC Power Circuit Breakers Used in Enclosures.
- **ANSI/IEEE Standard 141** [6], IEEE Recommended Practice for Electric Power Distribution of Industrial Plants (IEEE Red Book).
- **ANSI/IEEE Standard C37.5** [2], IEEE Application Guide for AC High-Voltage Circuit Breakers Rated on a Total Current Basis. (Standard withdrawn).

ANSI C37.010 details the procedure for equipment applied in medium and high voltage systems considering a classification of the generators as either “local” or “remote” depending on the location of the fault, as well as taking into account motor contribution. The procedure also covers first cycle and interrupting time currents, with emphasis on interrupting time currents.

ANSI C37.13 details the procedure for power circuit breakers applied in low voltage systems (less than 1000 Volts), while mainly focusing on first-cycle currents, impedance of motors and the fault point X/R ratio. Typically, fuses and low voltage circuit breakers begin to interrupt in the first half cycle so no special treatment for interruptive current is given. It could be the case however, that nevertheless the equipment test include a dc component specification.

Due to the differences in the high and low voltage standards, it would be understandable to say that two first-cycle calculations are required. The first calculation would be for high voltage busbars and a second calculation would be for low-voltage busbars.

In IEEE/ANSI Standard 141-1993 [6] (Red Book) a procedure for the combination of first cycle network is detailed. There is stated that in order to simplify comprehensive industrial system calculations, a single combination first-cycle network is recommended to replace the two different networks (high/medium-voltage and low voltage). This resulting combined network is then based on the interpretation of the ANSI C37.010 [1], ANSI C37.13 [4] and ANSI C37.5 [2] there given.

### Total and Symmetrical Current Rating Basis of Circuit Breakers and Fuses according to ANSI Standards

Depending on the circuit breaker year of construction different ratings are specified. High-voltage circuit breakers designed before 1964 were rated on “Total” current rating while now a day’s high-voltage circuit breakers are rated on a “Symmetrical” current basis. The difference between these two definitions is on how the asymmetry is taken into account. While a “Total” current basis takes into account the ac and dc decay, “Symmetrical” current basis takes into account only the ac decay. To explain further these definitions refer to Figure 26.2.5.

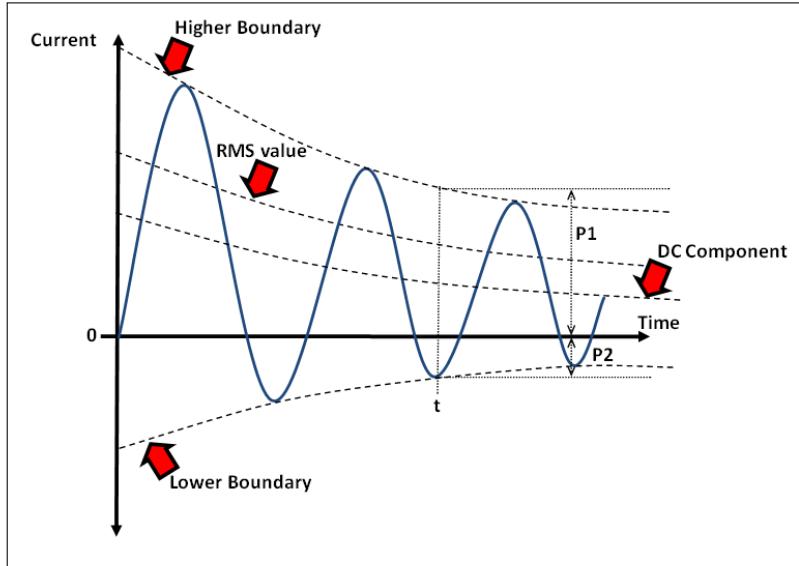


Figure 26.2.5: Asymmetrical Short-Circuit Current

The DC component “DC” is calculated according to the following equation:

$$DC = \frac{P_1 - P_2}{2} \quad (26.1)$$

The RMS value of the ac component (RMS) is then calculated as:

$$RMS = \frac{P_1 + P_2}{2.828} \quad (26.2)$$

The total interrupting current in RMS is then:

$$Tot = \sqrt{DC^2 + RMS^2} \quad (26.3)$$

From the above, Equation (26.2) corresponds to the “Symmetrical” current calculation and Equation (26.3) to the “Total” current calculation.

Some of the main ANSI guidelines for the calculation of short-circuit currents are the following:

- The pre-fault busbar voltage is assumed to be nominal (1.0 p.u.).
- The fault point X/R ratio is calculated based on a separate resistance network reduction which is latter used to calculate the peak and total asymmetrical fault current.
- Depending on the location of the fault, the generator currents being fed to the short circuit are classified as “local” or “remote”. A remote source is treated as having only a dc decay, while a local

source is treated as having a dc and ac decay. Depending on this classification, corresponding curves are used in order to obtain the multiplication factors.

According to ANSI standard, the following short-circuit currents are calculated:

- $I_{symm}$  symmetrical momentary (first cycle) short-circuit current (RMS),
- $I_{symi}$  symmetrical interrupting short-circuit current (RMS),
- $I_{16asymm}$  asymmetrical momentary (Close and Latch - Duty) short-circuit current (RMS). Obtained by applying a 1.6 factor to  $I_{symm}$ ,
- $I_{27peakm}$  peak short-circuit current (instantaneous value). Obtained by applying a 2.7 factor to  $I_{symm}$ ,
- $I_{asymm}$  asymmetrical momentary (Close and Latch - Duty) short-circuit current(RMS). Obtained by applying a factor to  $I_{symm}$  according to the calculated X/R ratio,
- $I_{peak_m}$  peak short-circuit current (instantaneous value). Obtained by applying a factor to  $I_{symm}$ , according to the calculated X/R ratio.

### 26.2.3 The Complete Method

The complete method is a superposition method of short-circuit analysis which can produce more accurate calculation results given realistic pre-fault loading of the system. The short-circuit results are determined by superimposing the set of healthy load-flow results for the network state at the moment before short-circuit inception (Figure 26.2.6a), with a set of results representing the system in the faulted state (Figure 26.2.6b). In the faulted state all network voltage sources are shorted whilst an additional voltage source is connected at the fault location injecting a voltage equal and opposite to the pre-fault voltage at that location, as determined from the load flow calculation.

The load flow calculation will account for the excitation conditions of the generators, the tap positions of regulated transformers and the switching status of the network according to the control options of the associated load flow command and the configuration of the models in the network.

Since accurate loading information is key for the accurate determination of fault currents, this method is particularly useful for mature network models where detailed operational data for the network should be readily available. For networks which are still in the design stage it may be more appropriate to use another short circuit calculation such as the IEC 60909 method (see Section 26.2.1), as this should produce more conservative results.

The superposition of the load flow and faulted state of the system is illustrated in Figure 26.2.6c.

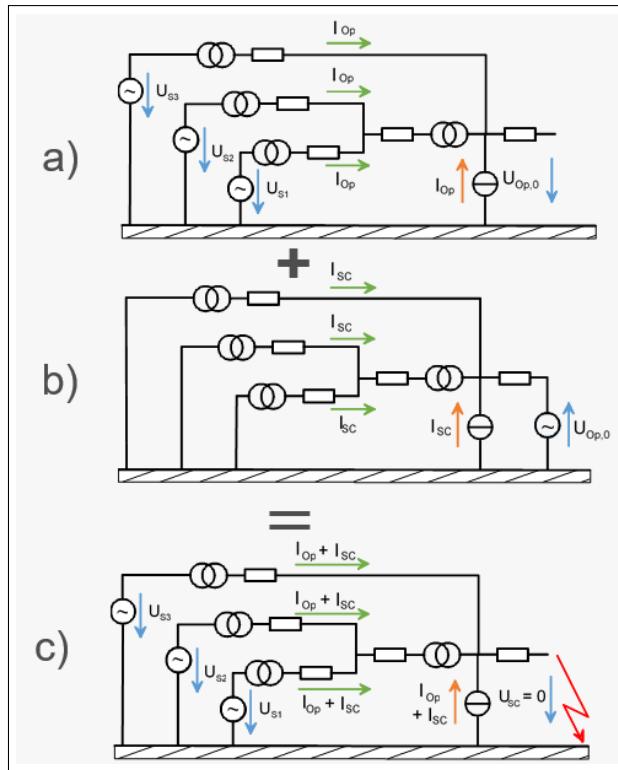


Figure 26.2.6: Illustration of the Complete Method procedure

The complete method is not constrained to three phase systems and may also be used for calculating short circuits in unbalanced network representations. However, note that for unbalanced network representations and indeed for unbalanced short circuits it must be initialised with an unbalanced load flow calculation.

The complete method additionally allows for the calculation of transient short circuit currents. In order to consider this accurately it is necessary to provide additional data in the machine models included in the network model. Further information on these parameters can be found in the associated technical references.

The *Complete Method* is not a calculation carried out in accordance with any specific standard. It is based on a more comprehensive set of data than is used by the simplified calculations of the standards and should theoretically produce more accurate results for the subtransient (initial) short-circuit currents. Just as the IEC 60909 standard derives additional result parameters from these currents via the application of various factors the complete method does the same. However, where possible these calculations are enhanced to take account of the additional transient results which are available. These enhancements are documented in the Issue 2 (July 2020) of the ENA Engineering Recommendation G74. However, some examples of how the complete short circuit calculation results have been enhanced are also outlined below. The mentioned quantities can be identified in Figure 26.2.1.

- A more precise Peak Short-Circuit Current  $i_p$  is calculated based on the accurate subtransient short-circuit current (which is calculated using the complete method) and the R/X ratio (which is based on the IEC 60909 standard[27]);
- The Short-Circuit Breaking Current  $I_b$  (RMS value) is calculated based on the subtransient short-circuit current and the transient short-circuit current (both of which are calculated by the complete method);
- The Peak Short-Circuit Breaking Current  $i_b$  is calculated from the RMS short-circuit breaking current  $I_b$  and the decaying d.c. component;
- The Thermal Equivalent Short-Circuit Current  $I_{th}$  is calculated based on the IEC standard, using

the m and n factors (see Figure 26.2.1). The n-factor calculation uses the transient current instead of the steady-state current;

- Additionally, loads can have a contribution to the short-circuit current, which can be defined in the load element (Fault Contribution section of *Complete Short-Circuit* tab).

Using the *Complete Method* it is possible to calculate results which take account of the procedures outlined in Issue 1 (1992) and further developed in Issue 2 (July 2020) of the ENA Engineering Recommendation G74. The Issue 2 update introduced two additional short-circuit models for converter driven plant, specifically a doubly fed induction generator model and a full size converter model. In *PowerFactory* the configuration of these models is carried out in either static generator models *ElmGenstat* or in a doubly fed induction machine models *ElmAmsc*. With further information on the configuration of these models available in the associated technical references.

#### 26.2.4 The IEC 61363 Method

The IEC 61363 standard [9]describes procedures for calculating short-circuit currents in three-phase AC radial electrical installations on ships and on mobile and fixed offshore units.

The IEC 61363 standard [9]defines only calculation methods for three phase (to earth) short circuits. Typically marine/offshore electrical systems are operated with the neutral point isolated from the hull or connected to it through an impedance. In such systems, the highest value of short-circuit current would correspond to a three phase short circuit. If the neutral point is directly connected to the hull, then the line-to-line, or line-to ship's hull short-circuit may produce a higher current. Two basic system calculation approaches can be taken, “time dependent” and “non-time dependent”.

According to the IEC 61363 standard [9], *PowerFactory* calculates an equivalent machine that feeds directly into the short circuit location. This machine summarises all “active” and “non-active” components of the grid.

The short-circuit procedure in IEC 61363 [9] calculate the upper envelope (amplitude) of the maximum value of the time dependent short-circuit (see Figure 26.2.2). The envelope is calculated using particular machine characteristics parameters obtainable from equipment manufacturers using recognised testing methods, and applying the following assumptions:

- All system capacitances are neglected.
- At the start of the short-circuit, the instantaneous value of voltage in one phase at the fault point is zero.
- During the short-circuit, there is no change in the short-circuit current path.
- The short-circuit arc impedance is neglected.
- Transformers are set at the main tap position.
- The short-circuit occurs simultaneously in all phases.
- For generator connected in parallel, all generators share their active and reactive load proportionally at the start of or during the short-circuit.
- During each discrete time interval, all circuits components react linearly.

The exact guidelines on how this is achieved is specified in the standard.

Because the standard considers specific system components and models (“active” and “non-active”) some of the models that can be used in *PowerFactory* will have no description according to the standard (such as External Grids, Voltage Sources, Static Generators, etc.). How these elements are considered and transformed to a replacement equivalent machine is described in the [Technical References Document](#).

According to this method, the following short-circuit values are calculated:

- $I''_k$  initial symmetrical short-circuit current,
- upper envelope of short-circuit current  $I_k(t)$ ,
- $i_{dc}(t)$  decaying (aperiodic) component of short-circuit current,
- $i_p$  peak short-circuit current,
- $I_k$  steady-state short-circuit current.

The calculating formulae and methods described produce sufficiently accurate results to calculate the short-circuit current during the first 100 ms of a fault condition. It is assumed in the standard that during that short time the control of the generators has no significant influence on the short circuit values. The method can be used also to calculate the short-circuit current for periods longer than 100 ms when calculating on a bus system to which the generators are directly connected. For time periods beyond 100 ms the controlling effects of the system voltage regulators may be predominant. Calculations including the voltage regulator effects are not considered in this standard.

In *PowerFactory* besides the standard IEC 61363 [9] method, an EMT simulation method is available which considers also the first 100 ms of a three phase short-circuit.

### 26.2.5 The IEC 61660 (DC)/VDE0102 part 10(DC)/DIN EN 61660 Method

The equivalent standards The IEC 61660 (DC)/VDE0102 part 10(DC)/DIN EN 61660 [8] describe a detailed method for calculating short-circuit currents in DC auxiliary systems in power plants and substations. Such systems can be equipped with the following equipment, acting as short-circuit current sources:

- rectifiers in three-phase AC bridge connection.
- stationary lead-acid batteries.
- smoothing capacitors.
- DC motors with independent excitation.

The IEC 61660 standard [8] defines equations and equivalent circuits which approximate the time-dependent fault contribution of different DC current sources. The standard also defines correction factors and approximation methods to determine the total DC short circuit current at the point of fault. A graphical representation of the DC short-circuit current time function of different DC sources is illustrated in Figure 26.2.7.

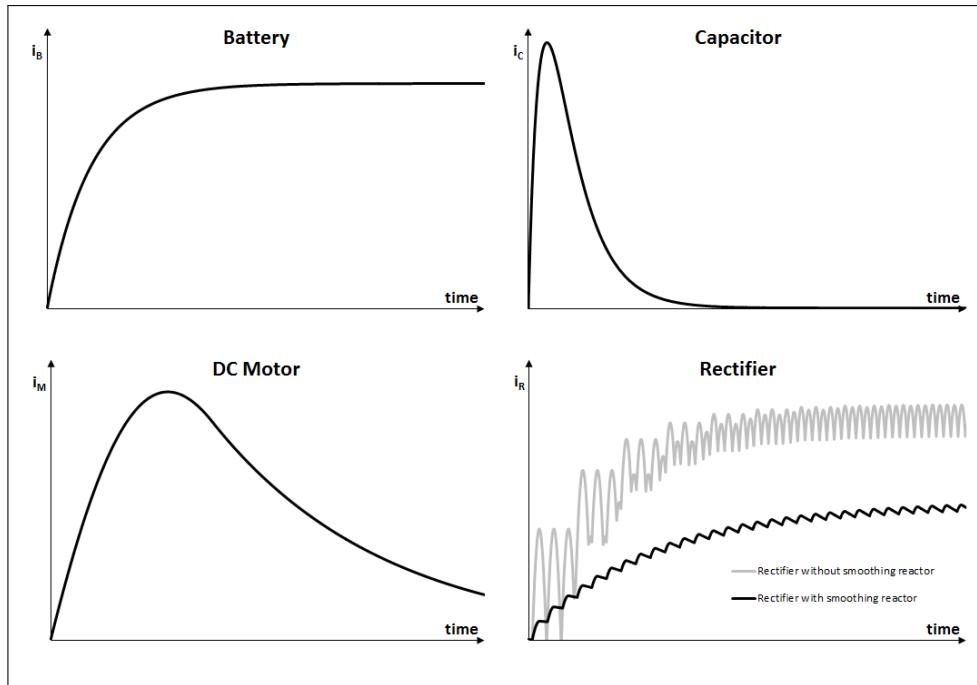


Figure 26.2.7: DC Short-Circuit Current Time Function of different sources

In accordance with standard IEC 61660 [8], *PowerFactory* calculates the total DC fault current by considering all of the DC current sources feeding in to the short circuit location. How different elements are considered and modelled is described in the [Technical References Document](#). Figure 26.2.8 shows the IEC 61660 standard approximation function which covers the different short circuit current variations. The equations which describe the function are detailed in IEC 61660.

There is no upper nominal voltage limit defined for the application of this method.

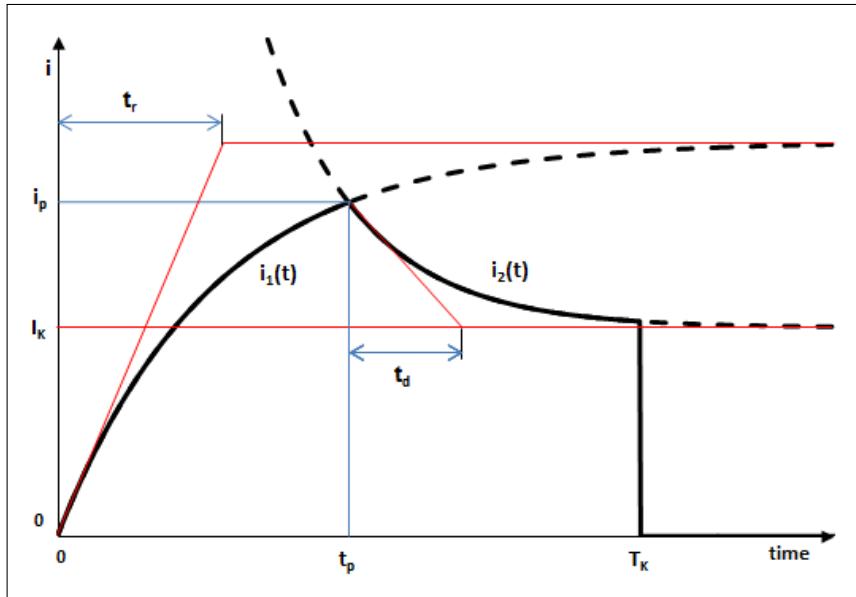


Figure 26.2.8: IEC 61660 standard DC short-circuit approximation function

According to the IEC 61660 method, the following short-circuit values are calculated:

- $i_p$  peak short-circuit current

- $I_k$  quasi steady-state short-circuit current
- $t_p$  time to peak
- $\tau_r$  rise-time constant
- $\tau_d$  decay-time constant
- $T_k$  short-circuit duration

### 26.2.6 The ANSI/IEEE 946 (DC) Method

The IEEE 946 standard [32] describes a recommended practice for the design of DC auxiliary power systems for nuclear and non-nuclear power stations. It also covers dc power system design in substations and telecommunication applications. The standard provides guidance on the selection of equipment including ratings, interconnections, instrumentation, control and protection. The calculation approach implemented in *PowerFactory* for the 2020 revision of the standard is equally applicable for calculations conducted in accordance with the 1992 revision of the standard. A notable difference between IEC 61660 and IEEE 946 is that for the latter, no provision is made for the consideration of smoothing capacitors as sources of short circuit current in its calculation. Provision is however made so that sources of short circuit current can be represented using rectifier, battery and DC motor models.

The IEEE 946 standard [32], is closely linked to General Electric's *Industrial Power Systems Data Book* [44]. The IEEE 946 standard includes examples for the calculation of short-circuit contribution from a battery and a battery charger, whilst the GE *Industrial Power Systems Data Book* includes a methodology for calculation of the DC fault contribution of Batteries, DC machines and Rectifiers. The DC short circuit calculation in *PowerFactory* is in accordance with the approach taken in the IEEE standard and the GE *Industrial Power Systems Data Book*. How different elements are specifically considered and modelled is described in the [Technical References Documentation](#).

According to the IEEE 946 method, the following short-circuit values are calculated:

- $i_p$  peak short-circuit current
- $I_k$  quasi steady-state short-circuit current
- $T_n$  network time constant
- $RR$  rate of rise of short-circuit current

Calculation cases requiring DC voltage supply above 1000 V nominal are excluded by the standard for the application of this method.

## 26.3 Executing Short-Circuit Calculations

There are different methods of initiating the short-circuit calculation command *ComShc* in *PowerFactory*, which may result in a different configuration of the command. These methods are described in Sections [26.3.1](#) and [26.3.2](#).

### 26.3.1 Toolbar/Main Menu Execution

The short-circuit command may be executed from the toolbar or main menu in *PowerFactory* as follows:

- By pressing the  icon on the main toolbar; or
- By selecting the *Calculation* → *Short-Circuit* → *Short-Circuit...* option from the main menu.

If the user is performing the short-circuit for the first time (by using one of the above options), the short-circuit command will be configured in a certain manner by default; that is the command will be set by default to execute a short-circuit calculation on all busbars/terminals in the network. If a short-circuit calculation has been already performed (the command exists in the study case) the settings displayed by the short-circuit command will be according to the most recent short-circuit calculation. As an example, if the user performs a short-circuit calculation according to ANSI for only one busbar in the system, the next time the user executes again the short-circuit, the command will have the most recent settings, that is, in this case according to ANSI and for the specified busbar.

### 26.3.2 Context Menu Execution

The short-circuit command may be executed from the context menu in *PowerFactory* by selecting an element(s) in the single-line diagram, right-clicking and selecting one of the following options:

- *Calculation → Short-Circuit → Short-Circuit...*: performs a short-circuit calculation for each element selected by the user. It should be noted that the short-circuit calculation for each element is carried out completely independently of the short-circuit calculation for each other element. For this calculation, only the following combinations of elements may be selected:
  - Single or multiple terminals/busbars; or
  - A single line; or
  - A single branch.

If several terminals/busbars are selected for analysis, the results of each individual short-circuit calculation will be displayed together on the single-line graphic.

- *Calculation → Multiple Faults...*: performs a short-circuit calculation according to the complete method, for the 'simultaneous' short-circuit of all elements selected by the user. Any combination of busbars, terminals, lines and branches can be selected for this calculation. Additionally, switch-/circuit breaker open/close operations can also be included in the calculation. When this calculation is selected, the option *Multiple Faults* in the *ComShc* dialog will be automatically ticked.

### 26.3.3 Faults on Busbars/Terminals

The short-circuit command should first be called using one of the methods described in Sections 26.3.1 and 26.3.2. The simplest way to calculate several busbar/terminal short-circuits individually and to then combine the results into one diagram is to select the option *All Busbars* (or alternatively, *Busbars and Junction Nodes*) in the *Fault Location* section of the *Short-Circuit Calculation ComShc* dialog. Note that to access this option, *Multiple Faults* in the dialog must be un-selected.

If the user would instead like to select from the single-line diagram a single busbar/terminal, or multi-select several busbars/terminals for calculation, the dialog will be configured as follows:

- When only a single busbar/terminal is selected, and *Calculation → Short-Circuit → Short-Circuit...* is chosen from the context menu, the *Fault Location* reference (bottom of dialog) is set to the selected element.
- When two or more busbars/terminals are selected and *Calculation → Short-Circuit* is chosen from the context menu, the *Fault Location* reference (bottom of dialog) is set to a so-called "Selection Set" *SetSelect* object, which contains a list of references to the selected busbars/terminals.

In either case, various options for the calculation can be modified. Refer to Section 26.4 for a detailed description of the options available. It should be noted that selecting or deselecting the option *Multiple Faults* may change the selection of fault locations and may therefore lead to a calculation for locations other than the busbars/terminals selected in the single line graphic. After pressing the **Execute** button, the calculation is executed and, if successful, the results are displayed in the single line graphic. In addition, a result report is available and may be printed out.

Once a selection of fault locations is made and the short-circuit calculation is performed, it is simple to execute further calculations based on the same selection of elements. This can be done by the following alternative means of executing the short-circuit calculation command:

- By pressing the  icon on the main toolbar; or
- By selecting the *Calculation → Short-Circuit → Short-Circuit...* option from the main menu.

The short-circuit setup dialog then shows the previously selected busbars/terminals in the *Fault Location* section under *User Selection*.

#### 26.3.4 Faults on Lines and Branches

It is not only possible to calculate short-circuits on busbars and terminals, but also on lines and branches. It should be noted, however, that only a single line or a single branch can be selected at a time, for each short-circuit calculation. It is not possible to select multiple lines and/or branches for calculation. To calculate a short-circuit on one of these types of elements, proceed as follows:

- From the single-line diagram, select a single line or a single branch where the fault should be calculated.
- Right-click on the element and select *Calculation → Short-Circuit → Short-Circuit...*. The short-circuit command *ComShc* dialog opens and the user can then define the location of the fault relative to the element's length (see Figure 26.3.1), including which terminal the fault distance should be calculated from. It should be noted that the *Short-Circuit at Branch/Line* section of this tab is only available when a line or branch has been selected for calculation.
- Clicking the button located in the *Short-Circuit at Branch/Line* section of the tab will enable the user to select whether the fault location is defined as a percentage or as an absolute value.

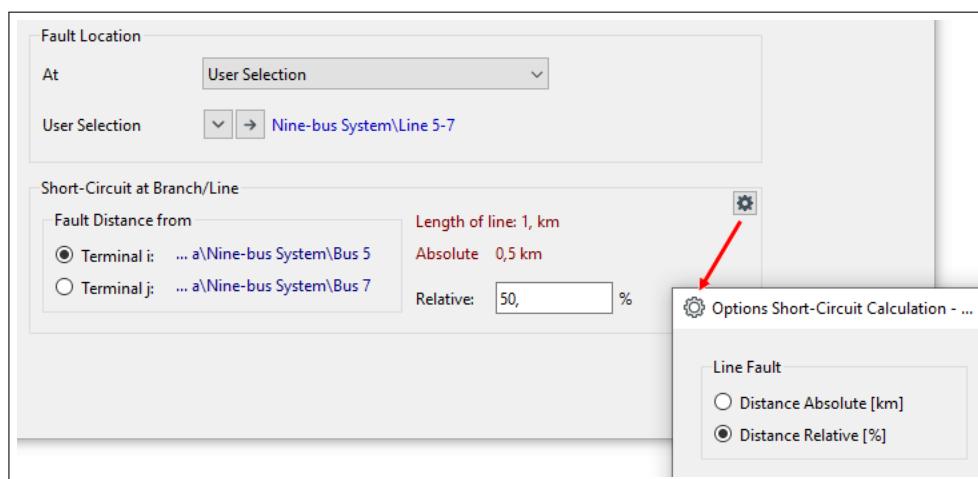


Figure 26.3.1: Configuration of Line/Branch Faults in *ComShc* dialog

When a fault on a line/branch is calculated, a box containing the calculation results is displayed next to the selected element.

#### 26.3.5 Multiple Faults Calculation

Multiple faults involve the simultaneous occurrence of more than one fault condition in a network. This option is only available for the complete method. To calculate simultaneous multiple faults, proceed as follows:

- Select two or more elements (i.e. busbars/terminals, lines, ...) and right-click.
- Select the option *Calculation → Multiple Faults*. . . . The *Short-Circuits* dialog pops up, displaying the short-circuit event list. A 3-phase fault is assumed by default at all locations in the event list. Click **OK**. The *Short-Circuit* command dialog then pops up. In this dialog, the *Multiple Faults* option is ticked in combination with the *complete* short-circuit method.
- Finally, press **Execute** to start the calculation.

In cases where the event list has to be adapted to reflect the intended fault conditions (that is, not necessarily the calculation of 3-phase faults), please proceed as follows:

- Open the short-circuit events object using one of the following methods:
  - In the *Fault Location* section of the short-circuit *ComShc* dialog, press the **Show** button (see Figure 26.3.2; or
  - Press the  icon located on the main tool bar (just besides the short-circuit command button); or
  - In a Data Manager window, open the *IntEvtshc* object from the current study cases.

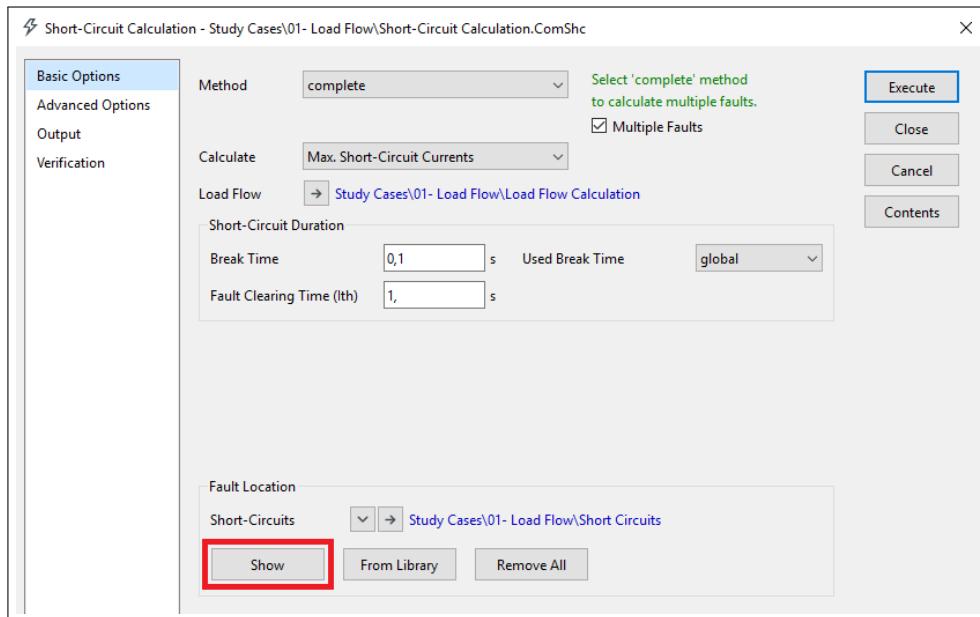
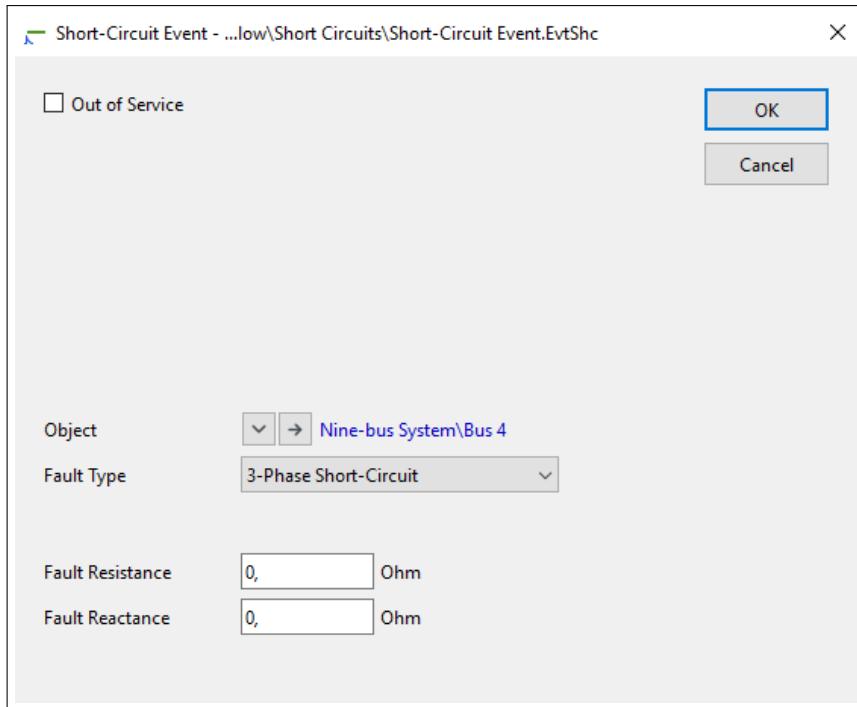


Figure 26.3.2: Accessing the Short-Circuit Events List

- A window opens up which shows the list of events (that is short-circuits at the selected locations). When double-clicking on one entry in this list (double-clicking on the entire row), a window with a description of the event is opened.
- The short-circuit event settings can now be modified. The list of fault locations consists of a “Short-Circuit Event List”(*IntEvtshc*) object, which holds one or more short-circuit events (*EvtShc*). Each of these events has a reference to a fault location (a busbar/terminal, line, etc.) and displays a short description of the fault type. An example is shown in Figure 26.3.3.

Figure 26.3.3: A Short-Circuit Event (*EvtShc*)

**Note:** To re-use the event list (*IntEvtshc*) later, this object can be copied to a user-defined folder in the Data Manager. This will prevent it from being modified during future calculations. When repeating the calculation with the same configuration, the reference in *Calculation → Multiple Faults...* can be set to this object. The other option would be to copy the events to the Fault Cases folder located in the “Operational Library/Faults” folder of the project. The user would then need to press the **From Library** button (26.3.2).

## 26.4 Short-Circuit Calculation Options

The following sections describe the options available in the *Short-Circuit Calculation* command (*ComShc*). Options are split into pages, which can be found on the left side of the command dialog. The option pages which are active at any time depend on the calculation method which is selected on the *Basic Options* page.

### 26.4.1 Basic Options

This section describes the basic options common to all calculation methods, as well as basic options that are method-specific. Options common to all methods are covered together in a single section, while those options that are dependant on the selected method will be covered in a separate subsection each.

#### 26.4.1.1 Basic Options (All Methods)

The sections of the short-circuit command dialog, which are common to all calculation methods are:

## Method

*PowerFactory* provides the following calculation methods for short-circuit calculation:

- *VDE 0102* [27] (the German VDE standard);
- *IEC 60909* [27] (the International IEC standard);
- *ANSI* (the American ANSI/IEEE C37 standard);
- *complete* (superposition method which considers the pre-fault load-flow results (see Section [26.2.3](#));
- *IEC 61363* [9];
- *IEC 61660 (DC)* [8]; (the International IEC standard for DC short circuit calculation)
- *ANSI/IEEE 946 (DC)* [5] (the ANSI/IEEE standard for DC short circuit calculation).

The specific options for each of these methods are available on the *Advanced Options* page of the short-circuit command *ComShc* dialog.

## Fault Type

The following fault types are available:

- 3-Phase Short-Circuit
- 2-Phase Short-Circuit
- Single Phase to Ground
- 2-Phase to Ground
- 1-Phase to Neutral
- 1-Phase Neutral to Ground
- 2-Phase to Neutral
- 2-Phase Neutral to Ground
- 3-Phase to Neutral
- 3-Phase Neutral to Ground
- 3-Phase Short-Circuit (unbalanced)

The fault types with a neutral conductor should only be used for systems which are modelled using neutral conductors.

## Fault Impedance (Except for IEC 61363)

The fault impedance corresponds to the reactance and the resistance of the fault itself (such as the impedance of the arc or of the shortening path). This can be defined by means of an enhanced model, where line to line ( $X_f(L - L)$ ,  $R_f(L - L)$ ) and line to earth  $X_f(L - E)$ ,  $R_f(L - E)$ ) impedances are regarded (note: requires option *Enhanced Fault Impedance* to be enabled). If the option *Enhanced Fault Impedance* is not enabled, fault impedances are defined by their equivalent values,  $X_f$  and  $R_f$ .

Figures [26.4.1](#) to [26.4.3](#) illustrate the differences between the enhanced and the simplified representation of fault impedances for the following fault types: (i) 3-phase short-circuits; (ii) 2-phase faults to ground; and (iii) 2-phase faults.

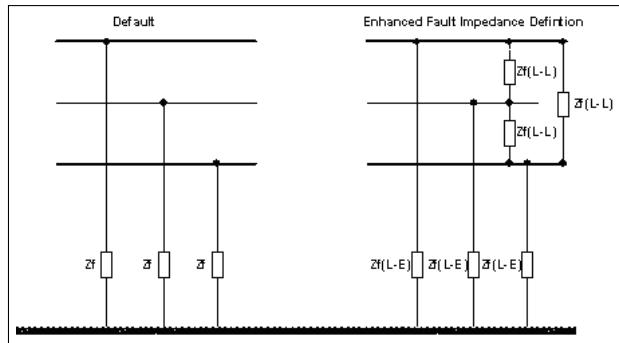


Figure 26.4.1: Fault Impedance Definition: 3-Phase Short-Circuit

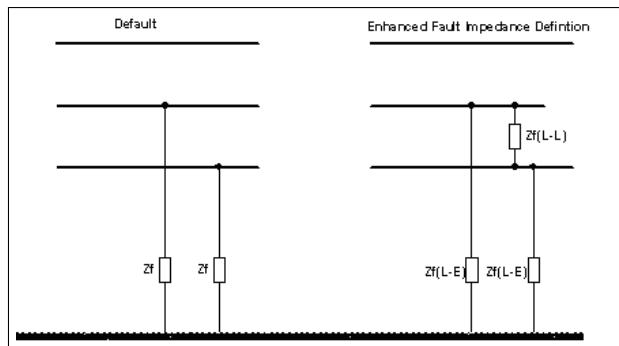


Figure 26.4.2: Fault Impedance Definition: 2-Phase to Ground Fault

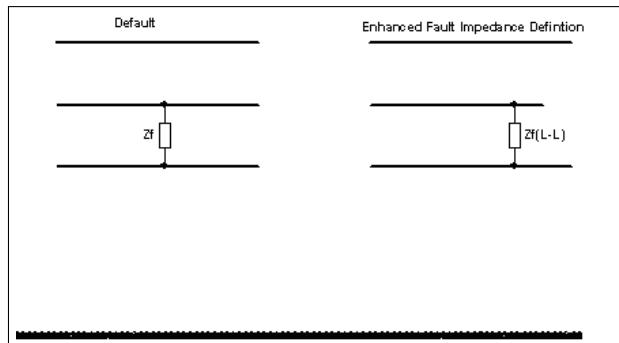


Figure 26.4.3: Fault Impedance Definition: 2-Phase Fault

## Fault Location

The fault location selection options are:

- **At User Selection:** In this case a reference to a single terminal/ busbar/ line/ branch or to a selection of busbars/ terminals *SetSelect*, as explained in Sections 26.3.3 and 26.3.4 must be given.
- **At Busbars and Junction Nodes:** For every terminal (*ElmTerm*) in the network whose *Usage* is set to *Busbar* or *Junction Node*, a short-circuit calculation is carried out, independently (one after the other).
- **At All Busbars:** For every terminal (*ElmTerm*) in the network whose *Usage* is set to *Busbar*, a short-circuit calculation is carried out, independently (one after the other).

If the option *Multiple Faults* has been ticked when the *Complete Method* is being used, a reference to a set of fault objects (*IntEvtshc*), as explained in Section 26.3.5, must be set. This is done in the *Fault Location* section of the dialog; using the *Short Circuits* reference.

---

**Note:** Multiple faults will only be calculated for the *Complete Method*, when the option *Multiple Faults* is enabled. When this option is enabled, a short-circuit calculation is carried out for each individual fault location, simultaneously. When this option is disabled, cases where more than one fault location have been selected (i.e. several busbars/terminals), a sequence of short-circuit calculations is performed (i.e. each short-circuit calculation is carried out independently of each other short-circuit calculation).

---

#### 26.4.1.2 Basic Options (VDE 0102/IEC 60909)

In general, note that the calculation according to IEC 60909 [27] and VDE 0102 does not take into account line capacitances, parallel admittances (except those of the zero-sequence system) and non-rotating loads (e. g. *ElmLod*). Single phase elements are considered only if they are located in the neutral conductor.

##### Published

This option offers a sub-selection for the selected *Method*, where the version of the standard to be used can be selected according to the year in which it was issued. The most recent standard is 2016, however 1990 and 2001 are still available for the verification of documented results. If 2016 is selected, wind farms (*ElmGenstat*), solar systems (both *ElmGenstat* and *ElmPvsy*) and asynchronous generators (*ElmAsm*) will all be supported in the short-circuit calculation.

##### Calculate

The drop-down list offers the choice between the minimum or maximum short-circuit current.

If external grids are defined, the corresponding maximum or minimum value will be selected automatically. For example if in the short circuit command you select “Calculate” according to “Maximum Short Circuit currents”, the maximum short circuit value from the external grid is considered for the calculation.

The equivalent voltage source is based on the nominal system voltage and the voltage factor *c*. The voltage factor *c* will depend on the voltage level and on the selection of the “Calculate according to...” stated in the short-circuit command.

##### Max. Voltage tolerance for LV systems

In accordance with the IEC/VDE standard, this voltage tolerance is used to define the respective voltage correction factor, *c*. The voltage tolerance is not used when a user-defined correction factor is defined.

##### Short-Circuit Duration

The value for the *Break Time* is used to calculate the breaking current of a circuit breaker. For more information on *Break Time* see Section 26.4.1.4. The value for the *Fault Clearing Time* (*Ith*) is required for the equivalent thermal current.

---

**Note:** The fields *Method*, *Fault Type*, *Fault Impedance*, *Output* and *Fault Location* are described in Section 26.4.1.

---

### 26.4.1.3 Basic Options (ANSI C37)

#### Prefault Voltage

Value of the pre-fault voltage. In ANSI, the pre-fault voltage is the system rated voltage (1.0 p.u.). Although a higher or lower voltage can be used in the calculation if operation conditions show otherwise.

#### Consider Transformer Taps

The ANSI standard optionally allows the current tap positions of the transformers to be considered. This can be selected here.

#### NACD Mode

Depending on the location of the fault, ANSI classifies the different currents being fed to the short circuit as “local” or “remote”. A remote source is treated as having only a dc decay, while a local source is treated as having a dc and ac decay. Depending on this classification, corresponding curves are used in order to obtain the multiplication factors.

In *PowerFactory* the ANSI short-circuit method has the option of selecting the NACD (No AC Decay) mode.

The NACD factor is the ratio of remote current contribution to the total fault current:  $NACD = I_{remote}/I_{fault}$ . This NACD factor is used to calculate the breaker currents, including the DC component of the current. The remote current contribution required to evaluate the NACD factor is the sum of all remote generator contributions (induction generators, synchronous machines, and external grids).

The calculation of the NACD factor can be very time consuming, as the contribution of each generator is calculated individually. Therefore, different approximation methods can be selected, which represent the most common interpretations of the ANSI standard:

**Interpolated** The NACD factor is calculated, and the correction factor for the asymmetrical fault current is interpolated between the “dc decay only” and “AC/DC decay” curves with the following equation:  $MF = AC/DC\ factor + (DC\ factor - AC/DC\ factor) * NACD$  If (NACD = 1) then only the DC factor is used; if (NACD = 0) then only the AC/DC factor is used.

**Predominant** The NACD factor is calculated. If the resulting factor is greater than or equal to 0.5, then the “dc decay only” curve is used, which means that the remote generation is higher than the local generation.

**All Remote** All contributions are set to ‘remote’; the NACD factor is not calculated, but assumed equal to 1 and only the “dc decay only” curve is used.

**All Local** All contributions are set to ‘local’; the NACD factor is not calculated, but assumed equal to 0 and only the “AC/DC decay” curve is used.

#### Current/Voltages for

The calculation mode for the currents and voltages to be evaluated:

**LV/Momentary** Evaluates the subtransient short-circuit currents.

**LV/Interrupting** Evaluates the breaker currents.

**30 Cycle** Evaluates the 30-cycle (steady-state) current.

---

**Note:** The fields *Method*, *Fault Type*, *Fault Impedance*, *Output* and *Fault Location* are described in Section [26.4.1](#).

#### 26.4.1.4 Basic Options (Complete Method)

As opposed to the calculation methods according to IEC/VDE and ANSI, which represent short-circuit currents by approximations, the complete method evaluates currents without using approximations. This accurate evaluation of the currents takes into account the system conditions immediately prior to the onset of the fault.

##### Load Flow

The pre-fault system condition used by the complete method can be determined either by the evaluation of a load flow, or by means of a simplified method, which initialises the internal voltages of all components that contribute to the short-circuit current with their nominal values, multiplied by a scaling factor,  $c$ .

The load flow command used to initialise the short-circuit calculation (when *Load Flow Initialisation* on the *Advanced Options* page is selected, see Section 26.4.5) is displayed next to the button labelled *Load Flow* ( $\rightarrow$ ). The load flow command can be accessed and modified by pressing this button  $\rightarrow$ . The load flow command displayed here is initially taken from the currently active study case.

##### Short-Circuit Duration

The value for the *Break Time* (when set to “Global”) is used to calculate the breaking current of circuit breakers. Depending on the user selection, the value used for the break time within the calculation is:

**global** When set to “Global”, the breaking current is calculated according to the *Break Time* specified in the short-circuit command.

**min. of local** When set to “min. of local”, the breaking current is calculated according to the shortest *Break Time* of all circuit breakers (defined in the *Complete Short-Circuit* page of *ElmCoup* objects) connected to the busbars being studied.

**local** When set to “local”, the breaking current is calculated for each connected circuit-breaker according to its own *Break Time* (defined in the *Complete Short-Circuit* page of *ElmCoup* objects), however, the busbar results will show the breaking current according to the shortest *Break Time* of all circuit breakers.

---

**Note:** The fields *Method*, *Fault Type*, *Fault Impedance*, *Output* and *Fault Location* are described in Section 26.4.1. The option *Multiple Faults* is described in Section 26.3.5.

---

#### 26.4.1.5 Basic Options (IEC 61363)

##### Calculate Using

In that section the user could select between the options:

- Standard IEC 61363 Method
- EMT Simulation Method

With the first option the short-circuit is calculated according to the IEC 61363 standard [9]this is outlined in Section 26.2.4. This short-circuit calculation method is only an approximation and therefore the results are not exact.

When selecting the EMT method *PowerFactory* calculates for each fault case a three phase short-circuit with a fault impedance of 0 ohm on the selected locations. This additional, high precision short-circuit calculation method provides further valuable information, and especially when power systems objects must be considered, which are not covered by the IEC 61363 standard[9].

The *Break Time* input parameter represents the contact separation time for circuit-breakers. The default setting is 100 ms.

If the *EMT Simulation Method* option is active the configuration of the *Simulation* and also the *Simulation Results* are available. The *Simulation* option displays the \*.ComSim dialog that is described in more detail in Chapter 29 (RMS/EMT Simulations). The simulation time is set per default to 160 ms. This is necessary because the short circuit is started after phase A voltage crosses zero and because the first 100 ms after the short-circuit are displayed as results.

The *Simulation Results* pointer indicates where the results of the EMT short-circuit simulation will be stored (*ElmRes*). Typically no changes are required. In another note, this EMT simulation setup (*Initial Conditions* and *Run Simulation* command) is stored separately from the normal EMT simulation in order to avoid confusion.

### Fault Impedance

The *Fault Impedance* option is disabled since the IEC 61363 [9] standard considers the short-circuit impedance to be zero.

#### 26.4.1.6 Basic Options (IEC 61660)

The *Basic Options* page of the *Short-Circuit Calculation* dialog provides options to set the fundamental settings of the IEC 61660 DC short circuit calculation. The calculation according to IEC 61660 [8] can be undertaken considering minimum and maximum fault conditions and a DC fault impedance.

### Calculate

The drop-down list offers the choice between the minimum or maximum short-circuit current. Depending on the selection *PowerFactory* will select model parameters from the provided data to taking account of the following requirements.

For the maximum short-circuit case the standard requires the following considerations:

- resistance of joints (in busbars and terminations) is ignored.
- conductor resistances are referred to 20°C.
- rectifier current limiting controls are disabled.
- decoupling diodes are considered as either blocked (infinite impedance) or conducting (zero impedance) depending on setting and batteries are assumed to be fully charged (maximum voltage).

For the minimum short-circuit case the standard requires the following considerations:

- resistance of joints is considered.
- conductor resistances are referred to their *maximum end temperature* rating.
- the contribution of rectifiers is set at the rated short-circuit current.
- battery operation is considered at the minimum voltage.
- individually specified resistances of decoupling diodes are considered.

The equivalent voltage source is based on the nominal system voltage and the *pre-fault voltage factor*.

## Short-Circuit Duration

The *Short-circuit duration (Tk\_dc)* is set here. The default value is 1 second.

---

**Note:** The fields *Method*, *Fault Impedance*, *Output* and *Fault Location* are described in Section [26.4.1](#).

---

### 26.4.1.7 Basic options (ANSI/IEEE 946)

The *Basic Options* page of the *Short-Circuit Calculation* dialog provides options to set the fundamental settings of the IEEE 946 DC short circuit calculation. The calculation according to IEEE 946 [32] can be used to undertake short circuit calculations considering minimum and maximum fault conditions including or excluding a DC fault impedance at the point of fault.

#### Calculate

The drop-down list offers the choice between the minimum or maximum short-circuit current. For the maximum short-circuit case the following assumptions are applied:

- conductor resistances are referred to  $20^{\circ}\text{C}$ .
- rectifier impedance is adjusted to use minimum impedance parameterisation.
- decoupling diodes are considered as either blocked (infinite impedance) or conducting (zero impedance) depending on a setting in the individual model.

For the minimum short-circuit case:

- conductor resistances are referred to their *maximum end temperature* rating.
- individually specified resistances of decoupling diodes are considered.

The equivalent DC voltage source is based on the nominal system voltage.

#### Short-Circuit Duration

The *Short-circuit duration (Tk\_dc)* is set here. The default value is 1 second.

---

**Note:** The fields *Method*, *Fault Impedance*, *Output* and *Fault Location* are described in Section [26.4.1](#).

---

## 26.4.2 Advanced Options

In this section, the advanced options listed on the *Advanced Options* page of the *Short Circuit Calculation* command dialog are described. In general options which are listed here are common to multiple (but not necessarily all) calculation methods.

---

**Note:** If an option listed here does not appear in the *Advanced Options* page of the command, it means that the option is not available for the configured calculation method

---

In addition to the advanced options page here, each of the calculation methods has its own specific advanced options page. These options are described in subsequent sections of this manual beginning at Section [26.4.3](#) below.

## Consider Protection Devices

This option will calculate measured currents for *all*, only *main* or only *backup* protection devices and will evaluate tripping times. To increase the speed of the calculation, this option can be set to *none* when protection devices do not need to be analysed.

## Calculate max. Branch Currents = Busbar Currents

This option is used to change the way that the “max” result variables (e.g. m:maxlkss, m:maxlb, m:maxip etc) of branch objects are calculated. If this option is activated, the more conservative, total current flowing in the busbar, is used to determine the variables. Otherwise the variables are calculated as described in the note below.

---

**Note:** Consider two external networks connected together by a branch element such as a circuit breaker. For a three phase fault on either terminal of the circuit breaker the current flowing in the fault will have a contribution from both external networks. However, the current flowing in the circuit breaker will include a contribution from only one of the external networks. The maximum current the circuit breaker could be exposed to (assuming a fault could be applied to either of its terminals) will depend on which external network provides the most current and not on the total current flowing in the fault. If a fault is applied at the connected terminal of a circuit breaker, the maximum current seen by the circuit breaker can be evaluated as  $\max([I_{busbar} - I_{branch}], [I_{branch}])$ . The “max” parameters can be used for recording such maximum currents and these parameters can then be compared against the manufacturer ratings of the circuit breaker. Alternatively, the total current flowing in the fault ( $I_{busbar}$ ) may also be used as a conservative overestimate.

---

## Consider Contingencies

The *Consider contingencies* option allows the user to run short-circuit calculations in combination with the *Contingency Analysis* functionality. Contingency cases can be separately defined with the *Contingency Analysis* tool, described in Chapter 27, and the *Short-Circuit Calculation* can be executed with single or multiple contingency cases. Calculation results are written to a result file and it is possible to display these results in *Tabular Reports*, which can be generated from a dedicated command found in the *Additional Tools* toolbox and described in Section 26.5.3.

## Parallel Computation

In cases where a large number of contingency cases are to be considered, it can be beneficial to use parallel computation in order to speed up the analysis by using multiple CPU cores and threads.

---

**Note:** The option to enable parallel computation is only visible in the short circuit command when the *Consider Contingencies* checkbox is selected.

---

Some settings for parallel computation are defined via the user object (*IntUser*) and can only be modified by the Administrator account, as described in Section 6.6.1. However, the user can make certain modifications to these settings, via the *Parallel Computing* page of the *User Settings* dialog (see Section 7.11). On this page there is a further link to the Parallel Computing page of the User object, from where a further link to the Parallel Computing Manager object can be found.

In the *Short Circuit Calculation* command the minimum number of contingencies is user definable. If a calculation considers fewer contingencies than the specified value, then parallel computation will not be applied. A package size is also specifiable, which defines how many contingencies will be calculated each time a parallel process is run. The options are explained in more detail in Section 27.4.10.

### 26.4.3 VDE/IEC

The *VDE/IEC* page is used for setting advanced options specific to the IEC/VDE 60909 methods. Familiarisation with the IEC/VDE 60909 standard is strongly recommended before modifying these options.

#### Voltage Factor c

The standard defines the voltage *factor c* to be used for the different voltage levels. In special cases the user may want to define the correction factor. In this case, there are several options.

1. *Standard defined table*: the table defined in the corresponding version of the standard IEC 60909/VDE 0102 will be used.
2. *User defined equivalent voltage source factor*: the explicitly entered equivalent voltage source factor, which is displayed in the *Advanced Options*, will be applied to the voltage source at the fault location. Note that a different factor can be entered for the maximum calculation as compared to the minimum calculation by changing the calculation option on the basic options page. For the impedance correction factor calculation, the table according to the standard will be used.
3. *User defined table*: equivalent voltage source factors can be entered in a user defined table.
4. *User defined table and equivalent voltage source factor*: adaptable user defined equivalent voltage source factor table for application in calculation of impedance correction factors, together with the possibility to enter the equivalent voltage source factor explicitly. This factor is displayed in the *Advanced Options* and will be applied to the equivalent voltage source for the Short Circuit calculation. Note that a different factor can be entered for the maximum calculation as compared to the minimum calculation by changing the calculation option on the basic options page.

#### Grid Identification

The calculation of the factor *kappa* is different in the cases of meshed or radial feeding of the short-circuit. Normally *PowerFactory* will automatically find the appropriate setting. The option *Always meshed* will force a meshed grid approach.

#### Conductor Temperature

When activating the *User-Defined* option, the initial (pre-fault) conductor temperature can be set manually. This will influence the calculated maximum temperature of the conductors, as caused by the short-circuit currents.

#### Asynchronous Motors

Whether the calculation considers the influence of asynchronous motors on short-circuit currents depends on this setting, which may be selected to *Always Considered*, *Automatic Neglection*, or *Confirmation of Neglection*.

1. *Always Considered* - asynchronous motors will always be considered.
2. *Automatic Neglection* - asynchronous motors will automatically be neglected if its contribution is not higher than 5 % of the initial short-circuit current calculated without motors.
3. *Confirmation of Neglection* - user will be informed via an input dialog if a motor contribution is not higher than 5 % of the initial short-circuit current calculated without motors. Using the dialog the user can choose if the asynchronous motor should be neglected or not.

### Peak Short-Circuit Current (Meshed network)

In accordance with the IEC/VDE standard, the following methods for calculating *kappa* can be selected:

**Method B'** Uses the ratio R/X at the short-circuit location.

**Method C(1)** Uses the ratio R/X calculated at a virtual frequency of 40% of nominal frequency (20 Hz for  $f_n = 50$  Hz, or 24 Hz for  $f_n=60$  Hz), based on the short-circuit impedance in the positive sequence system.

**Method (012)** Uses an R/X ratio calculated at a virtual frequency like C(1). However, the impedance of the system as seen from the fault location from which the ratio is calculated, will be based on a combination of positive-, negative- and zero-sequence impedances, with the combination depending on the fault-type carried out. This method is not mentioned in IEC 60909-0 but is demonstrated in IEC 60909-4.

### Decaying Aperiodic Component

Allows for the calculation of the DC current component, for which the decay time must be given. According to the IEC/VDE standard, methods *B*, *C* and *C'* can be selected.

The following nomenclature is used:

- $T_b$ )Breaker Time (see *Short-Circuit command*)
- $f_n$ )Nominal frequency
- $I_k''$ )Initial short-circuit current

**Method B:** Uses the complex calculated equivalent impedance of the network with a security factor of 1.15:

$$i_{DC} = \sqrt{2} \cdot I_k'' \cdot e^{-\omega \cdot T_b \cdot \frac{R}{x}} \quad (26.4)$$

**Method C** Uses the R/X ratio calculated with the equivalent frequency method. The equivalent frequency is dependent on the breaking time (see Table 26.4.1). This method is recommended for maximum accuracy.

$f_n * T_b$	< 1	< 2.5	< 5	< 12.5
$f_c/f_n$	0.27	0.15	0.092	0.055

Table 26.4.1: Breaking Times

$$I_{DC} = \sqrt{2} \cdot I_k'' \cdot e^{-\omega \cdot T_b \cdot \frac{R_f}{X_f}} \quad (26.5)$$

$$\frac{R_f}{X_f} = \frac{R_c}{X_c} \cdot \frac{f_c}{f_{nom}} \quad (26.6)$$

The ratio  $R_c/X_c$  is the equivalent impedance calculated at the frequency given by:

$$f_c = \frac{f_c}{f_{nom}} \cdot f_{nom} \quad (26.7)$$

**Method C'** Uses the R/X ratio as for the peak short-circuit current, thus selecting the ratio  $f_c/f_n = 0.4$ . This option speeds up the calculation, as no additional equivalent impedance needs to be calculated.

## Calculate $I_k$

The steady-state short-circuit currents can be calculated using different means to consider asynchronous machines:

- **Ignore Motor Contributions** Partial short-circuit contributions from motors will be displayed in the associated branch elements feeding into the fault. However, these contributions will be ignored for the calculation of the current flowing in the point of the fault itself. This means that at the fault location, the impedance of the network is considered, including the impedance of any connected motors. Based on this, an intermediate  $I''_k$  value is calculated. The proportion of this quantity, which originates from motors, is determined and this is then subtracted from the intermediate  $I''_k$  value, providing a final value of  $I''_k$ , which excludes motor contributions. From this final  $I''_k$  value,  $I_k$  is then calculated in accordance with the standard.
- **Without Motors** Asynchronous motors will not be considered in the calculation of the current  $I''_k$  and consequently  $I_k$ . The motors will therefore be treated as disconnected. In other words, at the fault location, the impedance of the network is considered excluding the impedances of any connected asynchronous motors and from which the value of  $I''_k$  can be calculated.  $I_k$  is then calculated from this quantity in accordance with the standard.
- **DIGSILENT Method** A value for  $I''_k$  with no motor contributions is calculated as per the *Ignore Motor Contributions* method.  $I_k$  is then calculated from this value using the equations from the standard, defining the breaking current. The breaking current is calculated under consideration of an infinite (very long) break time.

---

**Note:** The *Ignore Motor Contributions* and the *Without Motors* methods are based on two alternative interpretations of wording specified in the IEC 60909-0:2001 standard, relating to the consideration of mesh networks. Both are considered to be valid interpretations of the standard. Both approaches can be used to reproduce the results given in section 5 of the IEC 60909-4 technical report. However, only the *Without Motors* method can be used to precisely reproduce the results given in section 4 of the IEC 60909-4 technical report. Nevertheless, the *Ignore Motor Contributions* method is selected as the default calculation method on the basis that it offers a slight improvement in calculation performance whilst removal of the motors from the network topology as is done with the *Without Motors* method introduces additional complexity to the calculation result by potentially altering contributions from other machines in the network. A negative by-product of the *Ignore Motor Contributions* calculation approach is that the partial  $I_k$  contributions from motors which are displayed in the associated branch elements feeding into the fault, may be incoherent with the  $I_k$  value calculated at the fault point due to the calculation procedure explained above. For this reason these partial  $I_k$  contributions should generally be disregarded.

---

## Power Station Unit Detection

The IEC/VDE standard forces a different impedance correction factor to be applied to separate generators and transformers than that applied to a unit/block (power station) consisting of a generator, including its step-up transformer. *PowerFactory* tries to detect power stations.

When the *Automatic* option is checked and enabled, *PowerFactory* will automatically search for power station units according to the following criteria:

1. *Maximum search distance over line < [m]* length defined here represents the maximum length of the line that can be connected from the low voltage side of a transformer and the transformer will still be taken into account by the power station unit detection algorithm. If the line from the low voltage side of the transformer is longer, transformer will not be considered in a power station unit detection.
2. *Consider generators < [kV]* defines the voltage level where the power station unit detection will be performed.

When this option is disabled, block transformers must be marked accordingly by setting the *Unit Transformer* option available in the *VDE/IEC Short-Circuit* page of the transformer element dialog.

If the option power station unit detection is activated, short circuit command will start from all suitable generators in the network and look for the transformer in order to detect power station unit so that the corresponding impedance factor can be applied. If this option is not activated, short circuit command will look for all transformers with *Unit Transformer* option on the *VDE/IEC Short-Circuit* page set to true and will search for the suitable generator directly connected on its low voltage side.

#### 26.4.4 ANSI

##### Calculate

This option is used to select the various currents (according to the ANSI standard) which are to be calculated. The options are as follows:

- Momentary Current (Close and Latch Duties)
- Interrupting Current
- 30 Cycle Current

##### System

The selection of a particular *System* will determine which reactance multipliers are to be used for the rotating equipment. According to the IEEE 551-2006 standard, different reactance multipliers should be used for such equipment in medium/high voltage networks and low voltage networks. In order to consider the system in its entirety, the IEEE 551-2006 provides a convenient multivoltage approach.

The selection of the *System* is as follows:

- Multivoltage: this option uses reactance multipliers according to the multivoltage system approach and considers all voltage levels.
- HV/MV: the reactance values according to the high/medium voltage levels are used and only these voltage levels are considered.
- LV: the reactance values according to the low voltage level are used and only the low voltage level is considered.
- HV/MV/Separate LV: the reactance values corresponding to the high/medium and low voltage levels, respectively, are used and all voltage levels are considered.

The checkboxes allow the selection of the currents to be calculated.

##### Bypass Series Capacitance

Series capacitances may be optionally bypassed for the ANSI short-circuit calculation, or bypassed depending on the type of short-circuit being calculated.

The series capacitor bypass options are as follows:

- No Bypassing
- All Currents
- LV & Interrupting & 30 Cycle Current
- 30 Cycle Currents

## X/R Calculation

The user may select between a complex number X/R ratio calculation, or a calculation which considers R and X separately. The fault point X/R will determine the system dc time constant and consequently the rate of decay of the transient dc current. Although in *PowerFactory* the X/R ration can be calculated from the complex network reduction, using this approach will not insure a conservative result. In an attempt to provide a conservative approach, ANSI C37.010 requires that the X/R ratio be determined by a separate R network reduction.

## Additional Parameters

The *Additional Parameters* input box can be used to further fine tune the behaviour of the ANSI/IEEE C37 short-circuit calculation method. Users can input a string to control internal parameters of the calculation. The purpose of the parameter is primarily to enable the calculation to ignore the zero-sequence capacitance for short-circuit calculations with unbalanced faults. Improper use of the parameter could lead to misleading and incorrect results, therefore it is strongly advised that the user proceeds with caution when utilising this option. In order to ignore the zero-sequence capacitance, the following string should be entered in the text field:

```
/ignoreCO
```

## 26.4.5 Complete Method

### Peak, DC Currents, R/X ratio (ip, ib, idc)

This option allows the definition of the method used to determine the factor kappa ( $\kappa$ ) and the  $R/X_b$  ratio, required for the calculation of the peak and the DC component of the short-circuit current. The methods available correspond to those given in the IEC/VDE standard.

**B** Uses the ratio R/X at the short-circuit location. In this case both ratios ( $R/X_p$  for the calculation of  $\kappa$ , and  $R/X_b$ ) are equal.

**Method C(1)** Uses the ratio R/X calculated at a virtual frequency of 40% of nominal frequency (20 Hz for  $f_n = 50$  Hz, or 24 Hz for  $f_n=60$  Hz), based on the short-circuit impedance in the positive sequence system (as for Method C from the IEC 60909 standard). It should be noted that the IEC correction factors are not considered in this case.

**Method (012)** Uses an R/X ratio calculated at a virtual frequency like C(1). However, the impedance of the system as seen from the fault location from which the ratio is calculated, will be based on a combination of positive-, negative- and zero-sequence impedances, with the combination depending on the fault-type carried out. (This method is not mentioned in IEC 60909-0 but is demonstrated in IEC 60909-4)..

### Initialisation

The user may select to initialise the complete method by one of the following options:

- the load flow calculation referred to in the *Load Flow* field of the *Basic Options* tab; or
- the nominal voltages with a user-defined correction factor (*c-Factor*). It should be noted that this option is only available in the dialog when *Load Flow Initialisation* is not selected.

In cases when the user wants to initialise the calculation with a voltage factor, further options to disregard positive sequence data of the following network elements are available:

- Loads,
- Capacitance of Lines,

- Magnetising current of transformers and
- Shunts/Filters and SVS.

### Current Iteration

The current iteration approach is intended to be used for equipment models which have current injection behaviour during short circuit calculations which is described by non-linear equations that require an iterative solution in order to obtain an accurate result. If the option is selected, a fast current iteration algorithm is initiated during execution of the short circuit calculation for all applicable models.

**Acceptable current error** The iteration aims to achieve a set of solutions where the maximum tolerance between successive iterations for all relevant models is within the specified setting. When this is achieved, the algorithm terminates.

**Relaxation factor** Typically, the algorithm converges upon a solution within 5-10 iterations but if necessary, the convergence of the algorithm can be slowed down by setting a lower value of relaxation factor. This can help to achieve convergence upon an acceptable solution where with a higher relaxation factor this was not possible or can speed up convergence by avoiding toggling between extreme solution states at each iteration.

**Max. number of iterations** Used to prevent an excessive number of iterations from being computed. A higher maximum setting can be specified in order to accommodate more iterations if necessary.

**Detailed convergence output** By selecting this checkbox, a report on the progress of the current iteration algorithm can be printed to the output window.

Models which are impacted by the *Current iteration* option are described below:

- Generator models such as *Static generators* (ElmGenstat) with current source injection short circuit model configurations as follows:
  - *Dynamic voltage support*
  - *Doubly fed asynchronous generator*
  - *Full size converter*.
- Series capacitor models (ElmScap) where the *Consider Metal Oxide Varistor* option is selected. In this case, the current iteration is used to determine the operating point of the metal oxide varistor.

More information on the calculation approach with and without the current iteration option is available in the Technical References associated with the listed models.

---

**Note:** The current iteration approach is also used to ensure that the maximum current injection limit is respected for generators with current source injection short circuit models. Without consideration of the current iteration approach, it is possible that the current injection of these models can exceed the specified maximum current.

---

### Skip transient calculation

By default, transient and subtransient currents and associated parameters will be calculated when short circuit analysis is carried out using the Complete method. However, the user now has the option to omit the transient calculations if these results are not required, leading to a shorter execution time.

### Consider motors for min. short-circuit calculation

Motor contributions are usually ignored during a minimum short circuit calculation. However, with this option the user can choose to consider the motor contributions.

## Overhead Line Modelling: Phase Matrices

For the unbalanced short-circuit calculation, *PowerFactory* always uses the phase component matrix. The following options define which phase matrix is used:

- **Untransposed:** the short-circuit calculation uses the untransposed phase matrix.
- **Symmetrically Transposed:** the short-circuit calculation uses the symmetrically transposed phase matrix for untransposed lines.

## Calculate relay tripping with

This setting determines which result variables of the short circuit calculation are used for the calculation of certain relay block tripping times, for certain fault calculations. In *PowerFactory* the default approach, when using the complete short circuit method, is to consider the tripping of relays in response to subtransient short circuit result parameters. The setting described in this section offers an alternative approach for certain cases. The main application of this setting is for examining tripping times of overcurrent relays with time delayed elements or a combination of time delayed elements and instantaneous elements, in time overcurrent plots.

There are three options:

- **Subtransient Values:** some relay elements are not time delayed and react very quickly to short circuits i.e. within the subtransient period following fault inception. For these relay elements it may therefore be more appropriate to estimate their tripping time using *subtransient* short circuit result values. This option is the default option and can be selected for those cases. Currents shown in relay plots will represent the subtransient short circuit current.
- **Transient Values:** relay elements which are time delayed will not normally respond during the subtransient time period following fault inception but will operate later, once the time delay has elapsed. For these relay elements it may be more appropriate to estimate their tripping time using *transient* short circuit result values. This option can be selected for those cases. Currents shown in Time Overcurrent plots will represent the transient short circuit current.
- **Mixed Mode:** most relays consist of a combination of relay elements, some with instantaneous operation and some with time delayed operation. If this option is selected, *PowerFactory* will evaluate the tripping time of instantaneous or very fast acting elements against the subtransient calculation results while evaluating the tripping time of time delayed elements against the transient calculation results. When selected, it is necessary to specify the duration of the expected subtransient period. Any element that has a tripping time in excess of this setting in response to subtransient calculation results will have its tripping time evaluated against the transient calculation results, while the tripping times of the remaining elements will be evaluated against the subtransient calculation results. In this case currents shown in Time Overcurrent plots can represent either transient or subtransient currents. In cases where both currents are relevant, both will be shown on the same plot.

---

**Note:** Care should be taken when using this setting, as not all relay blocks and fault calculations react in the same way to the setting. The setting affects blocks Relloc, RelToc, RelChar, RelFuse and RelUlim. Other blocks are unaffected by the setting. The setting does not directly alter the output signals from any instrument transformers supplying relays; instead the outputs of the relay blocks that are affected by the setting are directly adjusted. The setting only applies when individual fault cases are being considered. The option should be set to 'subtransient' when a short circuit sweep is being carried out in relation to a time-distance diagram.

---

## 26.4.6 IEC 61363

The settings available on this page will depend on the selected calculation method.

### Create Plots

By enabling the *Create Plots* option, the user can select between the following:

- Show only *short-circuit currents at faulted terminal* With this option selected, *PowerFactory* will create automatically a time domain plot of the short-circuit current at the selected terminal, which includes its upper envelope and DC component.
- Show *all short-circuit current contributions* With this option selected *PowerFactory* will create automatically a time domain plot of the short-circuit current at the selected terminal and a plot for all connected elements to the faulted terminal. Each created plot will consist of the short-circuit current, the upper envelope and the DC component.

### Standard IEC 61363 [9]

With the standard calculation method the pre-load condition can be configured. The available options are:

- **use load flow initialisation:** if this option is selected, a load flow calculation is first carried out (a reference to the low flow command is shown). If the load flow is successful, the results are then used to calculate the short circuit.
- **use rated current/power factor:** if this option is selected, the preload condition is obtained from the rated values of the grid elements (no load flow calculation is executed).
- **neglect preload condition:** if this option is selected, no preload information is used to calculate the short circuit.

Furthermore, the user will notice the option 'Consider Transformer Taps'. According to the standard however, all transformers should be considered with their main position, therefore this option should be normally disabled.

### EMT Simulation Method

If the short-circuit is calculated using the 'EMT simulation method', in the Advanced Options page the user will have the option to assume the inertia as infinite, meaning that if selected, the acceleration time constant of all rotating machines will be set to 9999 seconds.

## 26.4.7 VDE/IEC (DC)

The *VDE/IEC (DC)* page contains more detailed settings applicable to the IEC 61660 calculation method. Detailed familiarisation with the IEC 61660 standard [8] on which this calculation is based, is strongly advised before modifying these options.

### Initialisation

This option is provided by *PowerFactory* to give the user more flexibility and represents a potential enhancement of the calculation standard. Instead of using the nominal system voltage a *Pre-fault voltage factor* applicable to the nominal voltage can be specified.

### Apply Line Loop Impedance

If this option is selected the command will double the inductance and resistance of line elements used in the calculation so as to account for the resistance and inductance of the return conductor.

### Joint Resistance For Lines

If the minimum fault level calculation is selected on the Basic Options page, the *Joint Resistance for Lines* parameters will become visible on the Advanced options page. Here it is possible to specify whether a globally specified line termination (joint) resistance value is used for all line terminations or alternatively whether the local values should be used. The locally specified values are specified geometrically using data from the line elements and types themselves according to equations from the standard. It is assumed that a joint is present at each end of each line element. The geometry and resistance of both of these joints is considered to be identical irrespective of whether a global value or a local value calculation is conducted.

## 26.4.8 ANSI (DC)

The *ANSI (DC)* page contains more detailed settings applicable to the ANSI/IEEE 946 calculation method. Detailed familiarisation with the IEEE 946 recommended practice document [32] on which this calculation is based, is strongly advised before modifying these options.

### Apply line loop impedance

If this option is selected the command will double the inductance and resistance of line elements used in the calculation so as to account for the resistance and inductance of the return conductor.

### Approximate equivalent rectifier resistance value

The *Industrial Power Systems Data Book* [44] describes a detailed step by step by step procedure for calculating rectifier short circuit current contributions within which, an equivalent rectifier resistance is determined, however, the book offers the alternative possibility of using a more easily calculated approximate value for the rectifier resistance. If the checkbox is selected then the approximate value is used for the rectifier resistance otherwise the mentioned alternative resistance is used. Only those rectifiers which have been selected to use the *Industrial Power Systems Data Book* model will be affected by this setting.

## 26.4.9 Output/Results

The *Output/Results* page of the *Short-Circuit Calculation* command offers the user an option to control the results output of the said calculation. The page itself is split into two tabs: *Output* and *Results*.

### 26.4.9.1 Output tab

In this context output refers to text which is printed to the *PowerFactory* output window when a short circuit calculation is executed. On the *Output* tab the user can adapt a variety of setting used to control the messages which are printed.

#### Verification (Except for IEC 61363, IEC 61660 and ANSI/IEEE 946)

The purpose of this function is to highlight network elements in the output window for which a user specified loading threshold is exceeded under short circuit conditions. When the (*Verification*) checkbox is enabled, the user can enter thresholds for peak, interrupting and thermal maximum loading. Upon

execution of the short circuit command a loading report is printed to the output window listing all network elements which have higher loadings than the defined maximum values. The report shows the various maximum loading and calculated currents for rated devices. Rated devices include, for instance:

- Lines which have a rated short-time current in their line type which is greater than zero; and
- Breakers or coupling switches which have a type with a valid rated current.
- Busbar terminals which have a type with valid current rating limits specified.

### Show ASCII output

Results from the short-circuit calculation can be reported using built in ASCII reports. These print key results of the short circuit calculation to the output window using predefined report formats. Configuration of the report formats can be accessed by following the link to the report command or by using the  icon on the main toolbar.

### Output

When executing the short circuit command information about the short circuit locations can be printed to the output window along with other information messages about the calculation. This option can be used to controls the level of detail which is reported. The following options are available:

- Off - the reporting is turned off completely,
- Short - only basic information is shown (for example, in the case where a batch of faults are executed all at once, only the number of faulted locations will be reported),
- Detailed - additional data will be shown (for the example above, each faulted location will be reported and listed separately).

#### 26.4.9.2 Results tab

The *Results* tab is only available for configuration when *Consider contingencies* is selected on the *Advanced Options* page of the short circuit command (see Section 26.4.2).

The consideration of contingencies in the *Short-Circuit Calculation* means that multiple sets of results are generated. In order to facilitate the storage of these results, the *Short-Circuit Calculation* command now additionally supports the configuration and use of result files. Use of the result file functionality is automatically enabled when the *Consider contingencies* check-box is selected. A reference to the used result file is listed next to the *Results* field.

---

**Note:** The Tabular report for the *Short-Circuit Calculation* available in earlier *PowerFactory* versions is still available and can be found in the *Additional Tools* toolbox.

---

### Element and variable selection

By default *PowerFactory* will record a number of specific variables, associated with a limited selection of network element object classes. These can be observed by clicking the *Variable selection* button. Once pressed, a new window opens listing the variable selection objects corresponding with each of the different object classes. By editing these objects a *Variable Selection (IntMon)* object dialog is displayed listing the available variables along with the variables selected for recording in the result file. The user has the option to introduce additional variable selection objects for other object classes or the existing objects can be modified to include more or fewer selected variables.

If the default variables are removed for any reason, they can be easily reselected by clicking the *Add default variables* button. This will not deselect any additionally selected variables.

---

**Note:** It is important to note that the default variables are pre-defined for each calculation method separately and can only be managed by the Administrator account. The settings can be accessed via the *Data Manager* by for example going to *System → Modules → Short Circuit Used Method → Results → Balanced*. The file path may be organised differently for different methods!

---

### Limits for recording

This parameter is used to set the current threshold above which a calculated result is recorded in the *Results* object. The monitored variable is the *Initial Short-Circuit Current (Ikss)*. If the threshold for this variable is exceeded for any object then all of the results associated with its object class' variable selection will be recorded in the result file.

Results from the calculation can be reported using a number of inbuilt tabular or ASCII reports (written to the output window). These can be executed separately after the calculation, as described in Section 26.5.3.

## 26.5 Results Analysis

In *PowerFactory* the results can be displayed directly in the single line diagram, in tabular form or by using predefined report formats. Several diagram colouring options are also available which facilitate a simplified results overview.

### 26.5.1 Viewing Results in the Single Line Diagram

Once a short-circuit calculation has been successfully executed, the result boxes shown in the single-line diagram will be populated. There is a result box associated with each “side” of an element.

The information shown inside a result box depends on the element to which it is associated. Several predefined formats can be selected, as described in Chapter 10: Network Graphics, Section 10.5. Result boxes can also be personalised as described in Chapter 19: Reporting and Visualising Results, Section 19.2.

### 26.5.2 Flexible Data Page

Once a short-circuit calculation has been successfully executed, pressing the *Open Network Model Manager...* button () located on the main menu will open a browser window with a list of all classes on the left side of the window that are currently used in the calculation. Clicking any of the classes will display all elements of that class that are currently used in the calculation in a table on the right side of the window. The left-most tab-page at the bottom of the browser is the Flexible Data tab page. Click on this tab page to show the flexible data. To change the columns in the flexible page, press the *Define Flexible Data* button () . This will bring a selection window where the set of variables can be edited. Section 19.3 in Chapter 19: Reporting and Visualising Results, describes the *Variable Selection* object which is used to define the variables to be presented.

### 26.5.3 Predefined Reports

There are a number of predefined report formats available to the user following a short-circuit calculation. The Report Generation command (*ComReport*) can be accessed from the  icon on the main

tool-bar, and lists the available reports, which are mostly in the “*DlgSILENT Library - Legacy*” section.

A general description of reports and the Report Generation command can be found in Chapter 19: Reporting and Visualising Results, Section 19.5, but it should be noted that much of the functionality of the command is not available for these “legacy” reports. The reports are available in both tabular format and as ASCII output, written to the output window. Legacy reports executed from the *Report Generation (ComReport)* tool can only be set as ASCII. To use tabular reports, the user must open the *Short-Circuit Calculation Tabular Report (ComSchreport)* tool, found in the *Additional Tools* toolbox.

#### 26.5.4 Diagram Colouring

When performing short-circuit calculations, it is very useful to colour the single line-diagram in order to have a quick overview of the results, for example if elements have a loading above rated short-time current or if peak short-circuit currents are higher than the specified values. In *PowerFactory* there is the option of selecting different colouring modes according to the calculation performed.

The use of colouring in network diagrams is described in Section 10.3.10.1, and more detail about the use of colours and colour palettes can be found in Section 4.7.8.

If a specific calculation is valid, then the selected colouring for this calculation is displayed. As an example, if the user selects the colouring mode “Areas” for “No Calculation” and “Loading of Thermal-/Peak Short-Circuit Current” for the short-circuit calculation, then the initial colouring will be according to “Areas”. However, as soon as the short-circuit is calculated, the diagram will be coloured according to “Loading of Thermal/Peak Short-Circuit Current”. If the short-circuit calculation is reset or invalid, the colouring mode switches back to “Areas”. The *Diagram Colouring* has also a 3-priority level colouring scheme also implemented, allowing colouring elements according to the following criteria: 1st Energising status, 2nd Alarm and 3rd “Normal” (Other) colouring.

- **Energising Status:** if this check box is enabled “De-energised” or “Out of Calculation” elements are coloured according to the settings in the “Project Colour Settings”. The settings of the “De-energised” or “Out of Calculation” mode can be edited by clicking on the *Colour Settings* button.
- **Alarm:** if this check box is enabled a drop down list containing alarm modes will be available. It is important to note here that only alarm modes available for the current calculation page will be listed. If an alarm mode is selected, elements “exceeding” the corresponding limit are coloured. Limits and colours can be defined by clicking on the *Colour Settings* button.
- **“Normal” (Other) Colouring:** here, two lists are displayed. The first list will contain all available colouring modes. The second list will contain all sub modes of the selected colouring mode. The settings of the different colouring modes can be edited by clicking on the *Colour Settings* button.

Every element can be coloured by one of the three previous criteria. Also, every criterion is optional and will be skipped if disabled. Regarding the priority, if the user enables all three criterions, the hierarchy taken account will be the following:

“Energising Status” overrules the “Alarm” and “Normal Colouring” mode. The “Alarm” mode overrules the “Normal Colouring” mode.

### 26.6 Short-Circuit Calculation for Single Contingency

After executing a short circuit calculation considering numerous contingency cases it may be useful to investigate specific contingency cases or even a specific location within a contingency case in detail. This can be achieved using the *Short-Circuit Calculation for Single Contingency (ComShccnt)* command. The contingency case and if necessary a specific fault location from the previously executed calculation can be specified in the ComShccnt command. The corresponding data from the associated result file can then be accessed and used to reproduce the calculation as though it had been performed

in isolation. In other words, the corresponding result data can be accessed from the *Single Line Diagram*, as well as the *Network Model Manager*.

The command can be found under the *Additional Tools* toolbox, or can be run directly from the *Fault locations with contingencies* and *Edge elements with contingencies Tabular Reports*. This is done by selecting the row containing the desired *Fault Location* or *Component* in the tabular report and by then pressing the *Detailed Study* button.

### Short-Circuit Calculation

This field provides the user with information about the type of calculation that will be reproduced, with information on the method, fault type, case and fault location. The command can't be modified from the tool, only selected and viewed.

### Contingency

The *Contingency* field is used to select the particular contingency case, for which the short-circuit calculation results will be reproduced.

### Detailed study at a single fault location

When this option is enabled, a specific fault location in the network can be specified, allowing the user to examine the results associated with the fault at one specific location for one specific contingency.

## 26.7 Capacitive Earth-Fault Current

In medium-voltage networks, resonant grounding can be used for suppressing transient ground-fault currents, thereby continuing power supply during single-phase earth faults. The dimensioning of the arc suppression coil (Petersen coil) depends on the capacitive earth-fault current of the grounded network area. The exact compensation of the capacitive earth-fault current by an inductive coil current of equal magnitude results in a minimum residual fault current.

In *PowerFactory* the capacitive earth-fault current of lines and cables in relation to the grounding device impedance and its rated current can be displayed for each grounding area in the network model in a pre-defined tabular report. Starting from a substation or node, a grounding area subsumes all parts of the network that are connected in the zero-sequence system to this substation or node.

To access this functionality in *PowerFactory*:

- Open the Network Data Assessment command ( ) and select “Reports”
- Select “Capacitive earth current”
- Press **Execute** to generate the tabular report

For each grounding area, the following results are shown in individual columns:

- **Grounding Area:** the starting point for the search of connected components in the zero-sequence system
- **Number of lines:** the number of lines belonging to the grounding area
- **Ice [A]:** the lumped-sum capacitive earth-fault current of the lines for each grounding area
- **Number of grounding devices:** the number of grounding devices for each grounding area
- **Effective grounding:** the type of effective grounding (isolated, compensated or solid)
- **Grounding device:** the grounding devices for each grounding area are listed

- **Re [Ohm]:** the grounding resistance for each grounding device
- **Xe [Ohm]:** the grounding reactance for each grounding device
- **Ir [A]:** the rated current for each grounding device  $I_r = \frac{U_r}{\sqrt{3}|R_e+jX_e|}$ .

The *Grounding Area's effective grounding* column result shall be displayed as compensated in cases where the *Petersen Coil* check box has been selected on the *Grounding/Neutral Conductor* tab of the *Basic Data* page of the relevant transformer network element. Beyond this, this flag has no influence on the calculations and is otherwise used for information purposes only.

To output these results to Excel or to HTML click the  icon and select either *Export as HTML* for HTML output in the default web browser, or *Export to Excel* to export the results to an Excel workbook.

---

**Note:** If network element data is modified, the report is not automatically updated. The option *Refresh* available via the  icon must be used to update the report.

---

# Chapter 27

# Contingency Analysis

## 27.1 Introduction

In Chapter 25 (Load Flow Analysis) the general aspects of load flow analysis and its main areas of application are presented. Two perspectives are discussed: that of planning and that of system operation; it is made clear that the behaviour of the system must always be analysed under both normal and abnormal conditions.

The term “contingency analysis” is essentially referring to the analysis of abnormal system conditions. In general, contingency analysis can be considered as the process of evaluating the network states resulting from unplanned “outages” of single elements (such as transformers, busbars, transmission lines, etc.) or groups of elements, in terms of post-fault loads and voltages.

Contingency analysis can be therefore used to determine power transfer margins or for detecting the risk inherent in changed loading conditions. This chapter deals with deterministic contingency analysis. The structure of this chapter is as follows:

- Section 27.2 gives a short overview of the contingency analysis functionality and associated concepts.
- Section 27.3 looks at the Contingency Analysis toolbar and describes briefly what each button does.
- Section 27.4 describes the various options of the Contingency Analysis command in detail.
- Section 27.5 looks at the standard reporting options available once the analysis has been run.
- Section 27.7 explains the different methods for creating contingencies.
- Section 27.8 goes into detail about the use of fault cases.
- The remaining sections cover various other aspects of Contingency Analysis such as Remedial Action Schemes and managing variables to be recorded.

## 27.2 Short Overview

This section gives an overview of the Contingency Analysis functionality, together with some basic concepts which are useful to know. More detail is available in the following sections.

Contingency Analysis is generally executed using  from the Contingency Analysis toolbar (see 27.3), where the *Contingency Analysis* command dialog allows the user to select which contingencies are to be analysed and specify settings as required.

There are two basic types of contingency analysis: AC, using an iterative AC load flow calculation, or DC, using the DC load flow (which is faster and also useful for cases when AC convergence is difficult). Contingency analysis consists of a base case load flow, then subsequent load flows where each contingency is considered in turn in order to analyse its effect on the network.

The user will also see a third method in the Contingency Analysis dialog, namely “AC Linearised Calculation”. This is a fast calculation method for contingency analysis, which represents the contingency case by using equivalent injections to reduce the flow through the faulted area to zero. The injections are calculated using a linearised estimate and the process avoids the need for a new load flow to be run for the contingency case. Where the algorithm detects that the linear method is not suitable for a particular contingency case it will revert to the standard method. The linearised method is faster than the traditional contingency analysis using a full load flow, but it does not consider the response of controllers and so is a more approximate method.

One important concept to appreciate is that of time phases. There are two basic options: Single Time Phase and Multiple Time Phase. You will find a detailed explanation of this concept in Section [27.4.3](#).

When executing the contingency analysis, you may notice in the output window some messages about the Optimised and the Standard methods used in *PowerFactory*. The Optimised method makes use of the existing Jacobian matrix from the base case and is used where possible, as it is faster; the Standard method is required when topology changes mean that the matrix needs to be rebuilt, and it is somewhat slower. This is all handled automatically by the Contingency Analysis function.

Nevertheless, there is an option to always use the Standard method, but this may greatly increase the calculation time.

Once the analysis has been run, the in-built reports can be used to look at the results.

The remainder of this section provides some additional information which may be useful in understanding how Contingency Analysis works.

## 27.2.1 Contingency Analysis Objects

Contingency Analysis is executed using the Contingency Analysis Command, *ComSimoutage*, which is stored in the Study Case. The command will execute individual contingencies.

The contingency objects are called *ComOutage*. They can contain simply a list of elements to be “outaged” to represent a fault on the network, or - more commonly - references to Fault Cases (\*.*IntEvent*), which define the fault using one or more events, with associated times. Fault cases are stored in the Operational Library.

### 27.2.1.1 Creating Contingencies before running the analysis

Contingency cases can be generated in three different ways:

- Via the definition and use of *Fault Cases* and *Fault Groups*; and/or
- Using the *Contingency Definition* (*ComNmink*) command, via its toolbar icon (
- By selecting component(s) in the single-line graphic or filter, right-clicking and selecting *Calculation* → *Contingency Analysis...*

Contingency cases can be created using references to user defined *Fault Cases* and *Fault Groups* (introduced in Chapter [14](#): Libraries, Section [14.6.4](#)) from the *Operational Library*. By means of a topological search, *PowerFactory* determines which circuit breakers must be opened in order to clear the faults, and generates the corresponding contingency cases. See Section [27.8](#) for more details.

Alternatively, contingencies can be created using the *Contingency Definition* command, as described in Section 27.7.1 (Creating Contingency Cases Using the Contingency Definition Command).

It is also possible to select elements via a graphic or filter, and then right-click *Calculation* → *Contingency Analysis*. . . . The contingency container of the contingency command will then be populated by contingency cases for each of the selected elements. Existing fault cases in the Operational Library will be checked and any which are found for the selected elements will be used. Where no fault case is found for a particular element, one will be created.

### 27.2.1.2 Dynamic Contingencies

For some applications, the contingencies of interest are very much dependent on the operational state of the network. For example, faults on circuits which are already heavily loaded will be more significant than faults on lightly-loaded circuits. The user has the option therefore to create contingencies “on the fly”, so-called dynamic contingencies.

This is done via the Contingency Analysis dialog, and is described in more detail in Section 27.7.3. This feature is particularly useful for contingency timesweep (see 27.4.5), as the contingency requirements may vary with time.

It is possible to use both static contingencies (as described above in Section 27.2.1.1) and dynamic contingencies in one Contingency Analysis calculation.

### 27.2.2 Results Recording

Results from contingency analysis are stored in results files outside the project, in the workspace area or in the result file directory (depending upon configuration).

The results files are then referenced from within the project (so that to the user they appear to be contained in the project) and they can be accessed for reporting or for exporting to a range of different formats and locations.

Options within the Contingency Analysis command dialog (Recording of Results page) allow the user to set voltage and loading limits to control the amount of information recorded, as well as specifying variables to be recorded. Additional filters for results recording can be defined.

If the in-built reporting is used, this offers further filtering of results of interest to the user, including maximum loading of branch elements, exceeded voltage limits, etc. Refer to Section 27.4 (Contingency Analysis command and options) for further information on configuring the reporting settings, and Chapter 13 Study Cases, Section 13.11 (Results Objects) for information on handling results objects (*ElmRes*) in *PowerFactory*.

### 27.2.3 Configuring Network Restoration

In *PowerFactory*, there are options available for reconfiguring the network following a fault.

One option is the use of Remedial Action Schemes, described later in Section 27.11.

Alternatively, the contingency analysis can be setup to consider (or not consider) predefined switching rules of substations; refer to Chapter 12: Building Networks, Section 12.2.7 for further information. The Switching Rule defines switching actions for different fault locations (arranged in a matrix) that can be reflected at a certain time. These switching actions will always be relative to the current switch positions of the breakers.

## 27.2.4 Visualisation

When contingency analysis is carried out by pressing Execute in the *ComSimoutage* command, the user will be able to display results on graphics. The result boxes show in this case the results of the final load flow carried out at the end of the calculation. However, if the graphic is coloured using the colouring option *Voltages / Loading*, the colouring for each element will reflect the result of the “worst” contingency result for that particular element.

A useful option for visualising the effect of a single contingency is to execute it alone. Then the actual network state resulting from that contingency, including the power flows and resultant voltages, will be seen in the graphic. This is described in section [27.4.11](#).

## 27.3 Contingency Analysis Toolbar

To access the various contingency analysis related functions within *PowerFactory*, click on the icon *Change Toolbox* ▾ and select “Contingency Analysis”. The figure below shows the functions available on the Contingency Analysis Toolbar.

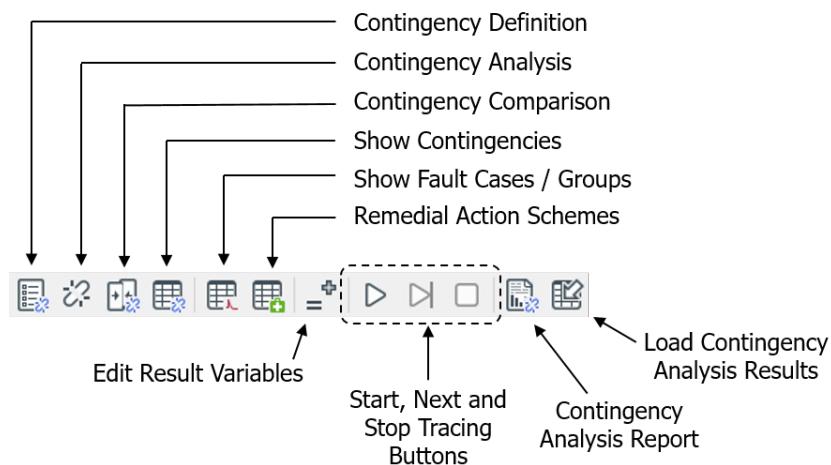


Figure 27.3.1: Contingency Analysis Toolbar Functions

### 27.3.1 Contingency Definition

This button gives access to the Contingency Definition Tool, which provides an easy method for the user to generate fault cases or contingencies. See section [27.7.1](#).

### 27.3.2 Contingency Analysis Command

Once the contingencies have been set up, the *ComSimoutage* command can be configured as required and executed. The configuration is highly flexible to cater for different users’ requirements. All the options are described in Section [27.4](#).

### 27.3.3 Contingency Comparison

A tool is provided to allow the user to compare the results of different contingency analysis. See Section [27.9](#) for details.

### 27.3.4 Show Contingencies

This button can be used to look at the current selection of contingencies.

Normally these are presented as a single list, but if the Dynamic Contingencies option has been used and dynamic contingencies have been generated, the contingencies are presented in a tree-structure so that the static and dynamic contingencies are separately listed.

### 27.3.5 Show Fault Cases / Groups

This button gives access to the Faults folder of the Operational Library.

### 27.3.6 Remedial Action Schemes

This button gives access to the Remedial Action Schemes folder of the Operational Library. See Section [27.11](#) for a description of Remedial Action Schemes.

### 27.3.7 Edit Results Variables

For each element class relevant to the contingency analysis, there is a standard set of results variables which are recorded in the results file (see Section [27.10](#)). The user can specify additional variables and this button provides easy access to the variable definitions.

### 27.3.8 Tracing Buttons

For Multiple Time Phase calculations, a Trace function is available. See Section [27.6](#) for details.

### 27.3.9 Contingency Analysis Reports

A set of in-built reporting scripts is available. These are described in Section [27.5](#).

### 27.3.10 Load Contingency Analysis Results

Once a contingency analysis calculation has been executed, it is possible to load results into memory at a later time. This feature is described in Section [27.12](#).

## 27.4 Command dialog and Options

### 27.4.1 Basic Options

This section describes the options of the Contingency Analysis Command, and their purpose. Some of the options are different depending on whether a Single Time Phase or Multiple Time Phase calculation is being done, and these differences are indicated in the description.

#### 27.4.1.1 Calculation Method

- **AC Load Flow Calculation.** The contingency analysis uses an iterative AC load flow method to calculate the power flow and voltages per contingency case.
- **DC Load Flow Calculation.** The contingency analysis uses a linear DC load flow method to calculate the active power flow per contingency case.
- **AC Linearised Calculation.** This is a fast calculation method for contingency analysis, which represents the contingency case by using equivalent injections to reduce the flow through the faulted area to zero. The injections are calculated using a linearised estimate and the process avoids the need for a new load flow to be run for the contingency case. Where the algorithm detects that the linear method is not suitable for a particular contingency case it will revert to the standard method. The linearised method is faster than the traditional contingency analysis using a full load flow, but it does not consider the response of controllers and so is a more approximate method.
- **Linearised screening + AC Load Flow for critical cases.** The contingency analysis will perform two runs (if required). First it will use a linearised load flow method to calculate the active power flow per contingency case; if for certain contingencies loadings are detected to be above a certain threshold, then these cases will be recalculated using the iterative AC load flow method. The choice of screening method and the criteria (thresholds) to be used for the AC recalculation of critical cases are entered on the *Screening* page (see Section [27.4.7](#)).

#### 27.4.1.2 Static Contingencies

The *Static contingencies* section of the *Basic Data* tab, as shown in Figure [27.4.1](#), allows the display, addition and removal of the static contingencies selected for analysis.

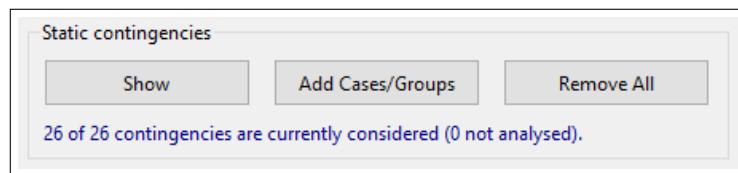


Figure 27.4.1: *Static contingencies* Section of Contingency Analysis Dialog

- **Show:** displays a list of all defined contingencies.
- **Add Cases/Groups:** used to create the contingency cases (*ComOutage* objects) based on fault cases and/or fault groups. A fault case contains events: one for the fault location, and (optionally) others specifying post-fault actions. Fault groups contain a set of references to fault cases. In order to use the **Add Cases/Groups** option, the fault cases and/or groups must have been previously defined in the *Operational Library*. If these have been defined, when the **Add Cases/Groups** button is pressed, a data browser listing the available fault cases/groups appears. The user can then select the desired fault cases/groups from this browser and press **OK**. The corresponding contingencies are then created automatically by *PowerFactory*. One contingency is created for

each selected fault case, and one contingency is created for each fault case referred to within each selected fault group. For further information on fault cases/groups, refer to Section 27.8 (Creating Contingency Cases Using Fault Cases and Groups).

- **Remove All:** removes all contingency cases (*ComOutage* objects) stored in the contingency analysis command.

#### 27.4.1.3 Dynamic Contingencies

If the *Dynamic contingencies* option is selected, the user can then configure one or more filters for generating dynamic contingencies during the contingency analysis. This feature is described in Section 27.7.3.

#### 27.4.1.4 Consider Remedial Action Schemes (RAS)

- **Consider Remedial Action Schemes (RAS):** If this option is enabled, all selected RAS (unless Out of Service) will be applied during the Contingency Analysis calculation. See Section 27.11 for more information about Remedial Action Schemes.

The selection of RAS works in exactly the same way as the selection of Contingencies as described above, with options to remove, add and view the selected RAS.

### 27.4.2 Recording of Results

#### 27.4.2.1 Elements and variable selection

- **Results for AC/DC:** Depending on the calculation method selected, the reference to the corresponding results file object (*ElmRes*) is defined. If, for example, the calculation method *DC Load Flow + AC Load Flow for Critical Cases* is selected, two results file objects will be referenced (one for AC calculations and another for DC calculations). The results stored in this file are filtered according to the global threshold set in the *Limits for Recording* panel, and also according to the individual limits defined within each component's respective dialog (such as on the *Load Flow* page of the element's own dialog). For further information on results objects, refer to Chapter 13 Study Cases, Section 13.11 (Results Objects).

There are also three buttons which give the user direct access to the “results object” (\**ElmRes*), which is held inside the Study Case:

- The **Element Filter** button allows the user to set up or modify filters in order to prescribe for which elements results should be recorded (for example, according to nominal voltage).
- The **Variable Selection** button allows the user to change the variable selection for the recording of results. There is a default set of variables which are recorded during the analysis and it is possible to add additional variables, like the bay loading or the busbar loading. It is also possible to remove variables, even those which are normally included. This gives the user great flexibility but removing variables should be done with care: if variables required by the in-built reports are removed, for example, those reports will fail to run.

---

**Note:** For the busbar loading also the according load flow setting need to be applied (see Section 25.3.3.4).

---

- The **Add default variables** button allows the user to restore all default variables to the selection used for the recording of results.

The user can optionally record additional information about the contingency cases:

- **Record additional result variables:** If this is selected, information will be recorded for each contingency to indicate the following outcomes:
  - Processed (meaning that the calculation was run)
  - Not solved (a subset of “Processed”; no converged solution)
  - Inactive (not processed because no components interrupted; the elements may be already out of service, for example)
  - Causing grounding (not processed because events would cause network to become earthed)
  - Causing islanding (results in isolated area(s) where the loads are still supplied)
  - Causing blackouts (results in isolated area(s) where the loads are no longer supplied)
  - Causing substation split or merge (results in a change in the number of coupled busbars in at least one substation)
  - Causing loss of load (at least one load is no longer supplied)
  - Causing loss of generation (at least one generator is no longer connected)

It is then possible to generate a summary report (a count of all these outcomes for the contingency run) and/or a report which gives the detail at a contingency level. Refer to Section [27.5.1](#) for more information about the reporting.

#### 27.4.2.2 Limits for Recording

These parameters set the global threshold used to determine whether a calculated result is recorded in the *Results* object. Whenever one of the defined constraints is violated, the calculated result (for the corresponding contingency case and network component) is recorded.

- **Different Limits for n-1 and n-k:** If required, the limits can be set differently for different order faults, that is, different thresholds can be specified for n-1 and n-k ( $k > 1$ ) faults.
- **Recording limits:**
  - **Record thermal loadings above (%)** Only loadings exceeding this value will be recorded in the results file for the corresponding component.
  - **Record if absolute voltages is below (p.u.)** Voltages lower than this value will be recorded in the results file for the corresponding terminal. The absolute value is used to cover DC elements that may have a negative voltage.
  - **Record if absolute voltages is above (p.u.)** Voltages higher than this value will be recorded in the results file for the corresponding terminal. The absolute value is used to cover DC elements that may have a negative voltage.
  - **Record voltage step changes above (%)** Voltage changes (change as a percentage of pre-fault) larger than this will be recorded in the results file for the corresponding terminal.
  - **Change of voltage angle above (deg)** Changes in the voltage angle above this limit will be recorded. If the value is set to 180 deg no voltage angles will be recorded. Take in mind that by default the voltage angle of busbars is not recorded during the Contingency Analysis. Therefore the angle need to be added to the result variables in beforehand.

---

**Note:** If the voltage is outside of the steady state limits defined in the busbar, it will always be recorded, irrespective of the recording limits.

---

#### 27.4.2.3 Recording filters

- **Recording filters for contingency loading results**
  - **Do not record if the base case is above (%)** If the pre-fault load flow elements have loadings above this value, then they are not recorded in the results.

- **Do not record if the absolute change in loading is below (%)** Decrease the recorded elements by filtering out elements which are less effected by the contingency.
- **Recording filters for contingency voltage results**
  - **Do not record if the absolute base case voltage is below (p.u.)** If the pre-fault voltages of the elements are below this value, then they are not recorded in the results.
  - **Do not record if the absolute base case voltage is above (p.u.)** If the pre-fault voltages of the elements are above this value, then they are not recorded in the results.
  - **Do not record if the absolute change in voltage is below (p.u.)** Decrease the recorded elements by filtering out elements with little change in voltage.
- **Additional option:**
  - **Record asynchronism of busbars next to the fault (deg)** This option recorders the voltage angle between the busbars around the fault if it is above the defined threshold. This is to check whether a fault could lead to a malfunction of an automatic circuit recloser. The parameter “Record busbars if asynchronism above” decreases the recorded elements by filtering the faults that leads to a voltage angle difference below the threshold. This parameter is not recorded for contingencies including transformers. The maximal asynchronism is reported within the summary report (see Section 27.5.1.4).

### 27.4.3 Time Phases

The *Time Phases* page allows the user to change between Single Time Phase and Multiple Time Phase and select appropriate settings according to the method.

#### Single Time Phase

Single Time Phase analysis uses a load flow calculation to assess the effect on the network of each of the specified contingencies in turn, at a particular time after the fault or as a steady-state final condition.

The single time phase contingency analysis function first performs a pre-fault (base) load flow calculation. Following this, for each contingency it performs a corresponding post-contingency load flow, which takes one or more primary components out of service. A final base case load flow is carried out at the end of the calculation.

By default, all calculations will use the same load flow settings, these being those defined in the Load Flow Calculation command (*ComLdf*) in the Study Case. However, some settings can be changed for the contingency case. For example, different Active Power Control methods may be used in the base load flow and the contingency analysis (with one exception that active power control according to secondary control in the contingency load flow is only supported if the base case load flow is also secondary controlled).

The results of the single time phase contingency analysis correspond to the steady-state operational points of the network being studied, considering each one of the defined contingencies at the given *Post Contingency Time*, which is found on the *Time Phases* page of the Contingency Analysis command dialog.

It is important to mention here that if the load flow command being used by the contingency analysis has *Automatic tap adjustment of transformers* or *Automatic tap adjustment of shunts* selected, they will only be considered if their time constants are not greater than the current *Post Contingency Time*.

Likewise, events in the fault cases (see section 27.8) have times associated with them, and so will not be considered if their times are later than the *Post Contingency Time*.

If the *Consider Specific Time Phase* flag is not enabled at all (and therefore there is no *Post Contingency Time* defined), then these constraints do not apply and the outcome is effectively the eventual steady state, with all actions taken.

The *Post Contingency Time* is also relevant if Thermal Rating objects that have short-term ratings are being used. Short-term ratings allow circuits to be operated at a level higher than their normal rating for a prescribed time. The *Post Contingency Time* is used to determine the applicable short-term rating for reporting purposes.

### Multiple Time Phase

As with Single Time phase, the multiple time phase contingency analysis function first performs a pre-fault (base) load flow calculation. Following this, for each contingency one or more load flows is calculated (depending on the number of time phases which have been specified). A final base case load flow is carried out at the end of the calculation. All the results are stored, so that the state of the network at each time phase can be reported.

The general principles described above also apply to Multiple Time Phase. The Multiple Time Phase, however, allows the user to specify more than one time phase and each contingency will be analysed at each of the requested time phases. The change in calculation results between one time phase and the next will result not only from the relationship between the calculation time and tap controller /shunt time constants, but also from the relationship between the calculation time and the time associated with the events in the fault cases.

Multiple Time Phase calculations are always executed using the Standard method, so although it is possible to use the Multiple Time Phase option with only one time phase being considered, for performance reasons the Single Time Phase would be preferable in that case.

Each defined time phase uses a corresponding load flow calculation, and by default, this is the same load flow calculation as that used for the base case load flow. If the option *Allow different settings* in the *Base Case versus Contingency Load Flow* section of the *Multiple Time Phases* page is selected, the user can define individual load flow commands for each time phase, as illustrated in Figure 27.4.2. Access to each load flow command and its settings is via the → button.

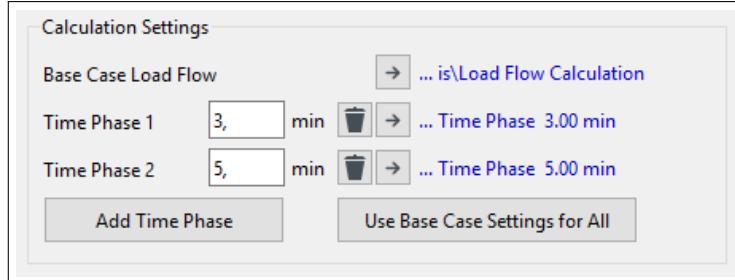


Figure 27.4.2: Different Settings for Base Case and Contingency Load Flows

There are some restrictions in which settings are allowed to be different between the different load flows (i.e. base case and the various time phases). Specifically, there is a restriction regarding the use of the Active Power Control option *According to Secondary Control*: Secondary Control can only be used in a time phase if it is used in the base case load flow, and if Secondary Control is used for one time phase it must be used for all.

The *Contingency Analysis* time phases (which are essentially load flow commands) are stored within a folder inside the *ComSimoutage* command and can be accessed in by clicking on the → button next to each defined time phase in the *Calculation Settings* section of the *Time Phases* tab; by doing so, the edit dialog of the corresponding load flow command is opened.

**Note:** Transformer tap changer controllers and switchable shunts are only considered by a time phase if their time constants are smaller than the current Post Contingency Time. The operational thermal ratings of branch elements during a contingency (if 'short term' thermal ratings (see Section 14.6.12) have been defined) will also depend on the duration of the contingency (i.e. the current Post Contingency Time).

### 27.4.3.1 Method

- **Single Time Phase.** Performs the contingency analysis for a single time phase.
- **Multiple Time Phase.** Performs the contingency analysis for multiple time phases.

### 27.4.3.2 Base Case versus Contingency Load Flow

- **Use same settings.** Uses the settings from the base case load flow for the contingency case load flow.
- **Allow different settings.** Allows different settings for the base case load flow and the contingency case load flow.

### 27.4.3.3 Calculation Settings *for Single Time Phase*

- **Base Case Load Flow.** Only available when option *Allow different settings* is selected. This is a reference to the load flow command used to calculate the network operational point before the simulation of contingencies. The settings of this load flow command can be edited by pressing the → button.
- **Contingency Load Flow.** Only available when option *Allow different settings* is selected. This is a reference to the load flow command used to assess the network in contingency situations. It takes account of the *Post Contingency Time*. The contingency load flow command referred to by the *Contingency Load Flow* is always stored inside the contingency analysis command itself. The settings of this load flow command can be edited by pressing the → button. The *Contingency Load Flow* command settings can be set to those of the currently used by the *Base Case Load Flow* command by pressing the BC button.

---

**Note:** If no 'Contingency Load Flow' command is defined, the 'Base Case Load Flow' command is used to asses the network under contingency situations. In this case the action of automatic transformer tap changers and switchable shunt compensators is directly considered (provided that the corresponding options are selected in the 'Basic Options' page of the assigned load flow command).

- **Consider Specific Time Phase.** Only available when option *Use same settings* is selected in the *Base Case versus Contingency Load Flow* section. This option must be enabled to define a post contingency time.
- **Post Contingency Time (End of Time Phase)** This value defines the time phase of the contingencies. This means that all events with an event time less than or equal to this are considered in the contingency.

### 27.4.3.4 Calculation Settings *for Multiple Time Phase*

- **Base Case Load Flow.**

Only available when option *Allow different settings* is selected. This is a reference to the load flow command used to calculate the network operational point before the simulation of contingencies. The settings of this load flow command can be edited by pressing the → button.

- **Time Phase n**

Lists the defined time phase(s). The button  next to each time phase can be used to remove the corresponding time phase. If the option *Allow different settings* has been selected on the *Advanced Options* tab, the *Time Phase* will have its corresponding load flow accessible by pressing the  button next to the defined time phase.

- **Add Time Phase** Opens an input dialog to define the new time phase by entering its *Post Contingency Time*. If the option *Allow different settings* has been selected on the *Advanced Options* tab, the previous load flow settings (i.e. those with the preceding occurrence in time) will be used for the new time phase. In the case that there is no previous time phase load flow, the base case settings will be used for the new time phase.
- **Use Base Case Settings for All** Copies the settings from the base case load flow to all time phase load flows.

New time phases can be defined in the data browser by clicking on the **Add Time Phase** button. Existing time phases can be deleted using the  button. Note that after several time phases have been defined, this list is then scrollable using the up/down arrow buttons ( ) available in the dialog.

- **Post contingency time for order identification**

The order of the contingencies stored inside the command is calculated according to the time defined in this field. Only the events (actions) taking place before this point in time are considered when calculating the contingency order.

---

**Note:** In *PowerFactory* a region is defined as a set of topologically connected components. A region is interrupted if it is energised (topologically connected to a network reference bus) before a fault and de-energised afterwards. The order of a contingency corresponds to the number of interrupted regions at the time of its calculation (i.e. the 'Post contingency time for order identification').

---

## 27.4.4 Effectiveness

### Only available for Single Time Phase

The *Effectiveness* page of the contingency analysis command, allows the display, addition and removal of quad boosters (or tap controllers containing quad boosters) and generators in order to calculate their effectiveness.

The effectiveness is only calculated if the loading of the respective element is recorded.

#### 27.4.4.1 Calculate Quad Booster Effectiveness

Quad Booster Effectiveness is a measure of the ability of a quad booster transformer to alter the power flow on a given circuit. It is normally defined per tap as a percentage change or actual power flow change. When this option is checked, the user has to define a list of transformers (quad boosters) or tap-controllers to be considered in the analysis. If a tap-controller is selected, the effectiveness will be calculated for the case where all the transformers associated with that tap-controller are tapped in parallel. The following buttons can be used:

- **Show:** shows a list of the transformers/tap-controllers for which the effectiveness should be calculated.
- **Add:** adds references to transformers/tap-controllers for which the effectiveness should be calculated. Only transformers where the additional voltage per tap is different to 0 and multiples of 180 degrees will be listed (*Load Flow* page of the transformer type (*TypTr2*) *Phase of du* parameter).

- **Remove All:** removes all references to transformers/tap-controllers for which the effectiveness is currently calculated.

Two calculation methods are available:

- **Linearisation of transformer tap changes:** uses linearised load flow equations around the operating point to derive sensitivities to quad booster tap positions.

- **Discrete transformer tap assessment:** actually solves the load flow at the current operating point, then with the tap position increased by one tap and then with it decreased by one tap. The change which decreases the overload on the branch is then stored as the sensitivity.

This method provides a more accurate assessment in cases when a strong dependence of the impedance on the current tap position is present, which, e.g., may result from a user-defined measurement report for the transformer.

For a DC calculation, the algorithm additionally checks whether the degree of dependence between the impedance and the current tap position is significant. If this is not the case the (faster) linearisation algorithm is used.

#### 27.4.4.2 Calculate Generator Effectiveness

Generator effectiveness is a measure of the ability of a generator to alter the apparent power flow on a given circuit or network element.

It is normally defined per MW injection (at the adjacent busbar of a generator) as a percentage change or actual power flow change. When this option is checked, the user has to define a list of generators to be considered in sensitivity analysis, the following buttons can be used for that purpose:

- **Show Gen.:** shows a list of the generators for which the effectiveness should be calculated.
- **Add Gen.:** adds references to generators for which the effectiveness should be calculated.
- **Remove All:** removes all references to generators for which the effectiveness is currently calculated.

#### 27.4.5 Time Sweep

##### ***Only available for Single Time Phase***

PowerFactory provides a *Calculate Time Sweep* option, whose settings allow the automatic modification of the date and time of the active *Study Case* according to a list predefined by the user. The Time Sweep calculation is designed for cases where the contingency analysis needs to be done for many different times (for example, each hour of a day), to take into account different system conditions such as changing load and generation.

---

**Note:** When enabled, the Time Sweep will automatically change the Date and Time of the active Study Case. However, in order for the Study Case to activate the corresponding scenario automatically, a Scenario Scheduler (*IntScensched*) object needs to first be created and afterwards activated. Once the execution of the contingency analysis has finished, the Study Case date and time is restored to its original setting. For more information on the Scenario Scheduler refer to Chapter 16(Operation Scenarios)

---

The Time Sweep calculation can be enabled on the *Time Sweep* page. To define the calculated points in time, edit the *Definition of Study Times* pane - the time period to run the simulation can be selected as follows:

- *Complete day* - a specific user-defined day can be selected as simulation time range. The day is chosen in the corresponding field *Day*
- *Complete month* - a specific user-defined month can be selected as simulation time range. The month is chosen in the corresponding field *Month*
- *Complete year* - a specific user-defined year can be selected as simulation time range. The year is chosen in the corresponding field *Year*
- *User defined calculation times* - specific user-defined calculation times can be added for the simulation (using the time format *DD:MM:YYYY hh:mm:ss*). The simulation will only be executed at the defined calculation times. The User also has the option to ignore certain points in time by activating the associated option.
- *User defined time range* - a customisable time period can be chosen for simulation by defining the *Begin* and *End* time points (using the time format *DD:MM:YYYY hh:mm:ss*).

*Step size* - Sets the time step size of the *Time Sweep* calculation. A fixed step size simulation algorithm is used for time advance. The step size is defined by the *Step* (integer value) and the *Unit*. The *Unit* can be selected from the corresponding drop-down list (*Seconds, Minutes, Hours, Days, Months or Years*). It is not possible to define a step size for the *User-defined calculation times*.

### Planned outages

If this option is selected, the contingency analysis will automatically apply for each time-step any planned outages (\*.IntPlannedout) which are valid for that time-step.

The button **Show Used Ones** can be used to show a list of all currently considered outages. Only those outages that occur during the defined simulation period on the basic data page are shown in this list.

The button **Show All** can be used to open the planned outages folder in a data navigator. From here it is possible to see all outages that are defined within the project, including those that are not applicable during the time sweep period.

## 27.4.6 Topology

### 27.4.6.1 General tab

#### Handling of busbar fault

- **Open both local and remote breakers.** For a bus fault, not only all local breakers which are directly connected to this bus, but also relevant remote breakers will be opened to isolate this bus and isolate the connected branches.
- **Open local breakers only.** Only the local breakers, which are directly connected to this bus will be opened to isolate this fault..

#### Contingency Analysis for specific region:

If this option is selected, the analysis can be restricted to part of the network, defined by a selected Grid, Area, Zone or Boundary. The rest of the network will be reduced and therefore only contingencies containing elements within the monitored region will be executed; likewise only results pertaining to elements within the monitored region will be reported.

If the reference bus does not lie within the monitored region, it will not be reduced, but be retained along with the slack machine.

It is possible to extend the region using the parameter *Region extension by k-neighbourhood*, which extends the monitored network from the cubicles at the edge of the region by the specified number of

elements, the default being 1. There is also an option to recalculate the base case load flow, or not, after running the contingencies. This base case load flow will be for the entire system.

#### Consider Switching Rules of Substations.

If this option is selected, any predefined switching rules in substations will be considered. The Switching Rule defines switching actions for different fault locations (arranged in a matrix) that will be done right after the fault. During the preprocessing of the contingency analysis, a topology search will be automatically carried out for each contingency to find out the interrupted elements. If any terminal in a substation is identified as interrupted in a contingency, the corresponding switch actions specified by the Switching Rule will be immediately applied after fault for that contingency. The additional execution time required is negligible. These switching actions will always be relative to the current switch position of each breaker. If the Switching Rule tries to operate a breaker which is already in that status, that rule will be ignored. For more information on Switching Rules, refer to Chapter 12, Section 12.2.7.4.

#### 27.4.6.2 Advanced tab

Four options are available on the Advanced tab:

- **Node Reduction Mode:** By default, any unsupplied components, for example a transformer which is switched out, will be removed from the calculation. On this page, an option is made available for users to prevent unsupplied components from being removed, for example if they wish to be able to include events or a RAS designed to switch in such components when the contingency is calculated.
- **Update Contingencies before running calculation:** if this option is selected, the list of interrupted elements for each *ComOutage* object will be updated. This provides additional information but also results in an increase in total calculation time.
- **Topology rebuild:** If this option is selected, the network topology will be rebuilt prior to the base case load flow, resulting in an increased total calculation time. If not selected, the topology rebuild will only take place as required.
- **Always use standard method:** If this option is selected, *PowerFactory* uses the standard method for all load flows and not for only those where the optimised method did not converge. This option is not available if the linearised calculation or the linearised screening is selected.

#### 27.4.7 Screening

The options on this page are only available if the calculation method *Linearised screening + AC Load Flow for critical cases* has been selected on the Basic Options page. With this option, a fast initial screening of contingencies is carried out, and cases that are identified as critical are then re-run using an AC Load Flow.

- **Screening Method:** The initial fast screening of contingencies can be carried out using one of two methods:
  - The normal DC load flow calculation
  - The AC linearised calculation.

Either of these two methods gives a solution which is potentially less accurate than the normal full AC analysis but have the benefit of being faster. Note that if DC screening is selected, the results from the initial screening will be in the DC results file, whereas the critical cases will be in the AC results file.

- **Criteria for AC Recalculation of critical cases:** Here the user specifies the criteria for identifying the critical cases to be recalculated using the AC load flow method. Either or both options may be selected.

- **Simple loading criterion:** The maximum loading of a component is greater than or equal to the first value specified; for example 100% (parameter name: *maxLoadAbs*);
  - **Combined loading criteria:** The maximum loading of a component is greater than or equal to the second value specified; for example 80% (parameter name: *maxLoad*) **and** the maximum relative change of loading compared to the base case is equal to or greater than the value specified; for example 5% (parameter name: *stepLoad*).
- **Components to be ignored:** There may be elements in a network for which high loading is not considered important, and such loadings should not (on their own) make a contingency case critical. The user can create a set of such elements so that they will not trigger a critical case. This set of components is assigned via the *Components to be ignored* field.
  - **Ignore components that are overloaded in base case:** If any component is loaded in the base case above the maximum loading threshold for critical cases, it is probable that most if not all contingencies would then cause a similar loading and so be considered critical. In that situation, the screening does not make sense, and so the calculation stops with a warning. This flag allows the user to override this action, with a request to ignore any loadings above the threshold for components which were loaded above the threshold in the base case.
  - **Screen only recorded elements:** This option can be used to ignore, for the purpose of screening, loadings on elements which are not going to be recorded. It is useful if the element filtering option is used (see 27.4.2).

## 27.4.8 Output

### Output per Contingency Case

- **Short.** Displays only the number of iterations required for each contingency case.
- **Detailed.** Displays the full load flow output per contingency case.
- **Show triggered RAS for each contingency in the output window.** If this option is enabled, a message will be output each time a RAS is triggered during the analysis. The message includes the name of the RAS as a hyperlink.

## 27.4.9 Linearised Calculation

The following options are only presented if the calculation method “AC Linearised Calculation” is selected on the Basic Options page, or if screening is selected and “AC Linearised Calculation” is used as the screening method.

### Sensitivity threshold used for linearised method

For a calculation using the linearised method, sensitivities have to be stored in memory. In general terms the more sensitivities values are stored, the more contingencies can be calculated using the linearised method (therefore the faster the overall calculation). However, storing this data consumes memory. The thresholds on this page enable the user to have some control over the memory usage by setting thresholds for the storing of sensitivities.

- **Minimal considered branch sensitivity:** Expressed as a percentage with 0.000001 as a default.
- **Minimal considered bus sensitivity:** Expressed as a percentage with 0.000001 as a default.

## 27.4.10 Parallel Computing

### Only available for Single Time Phase

The computation time required to perform a contingency analysis largely depends on the size of the power system and the number of contingencies considered. For lengthy analyses, parallel computation of contingencies speeds up the process by distributing the calculation effort over multiple processor cores of the host machine or in a distributed network with a number of remote machines.

There are two types of settings associated with the *Parallel Computing* option.

The first and more general group of settings are the ones related to the management of the parallel computation function (computing method and the assignments of parallel processes). Settings for parallel computing are defined centrally by the Administrator, as described in Section 6.6.1. However, the user can make certain modifications to these settings, via the User Settings dialog (see Section 7.11). Therefore, on this page there is a link to the Parallel Computing page of the User Settings, from where a further link to the Parallel Computing Manager will be found.

The second group of settings are the ones related to the execution of the contingency analysis; and which are located in the *Parallel Computing* page of the contingency analysis command.

- **Enable Parallel Contingency Analysis for AC, DC or Time Sweep.** If the corresponding option is enabled, the contingencies will be calculated in parallel; otherwise the contingency analysis is executed in its default mode (i.e. sequential calculation).
- **Minimum Number of Contingencies.** The parallel contingency analysis will be started only if the number of contingencies is greater than this setting.
- **Package Size for Optimised Method and Package Size for Standard Method.** The master distributes the contingencies to the parallel processes per package. The package size indicates how many contingencies will be calculated by a parallel process each time. The contingencies can be calculated using either optimised method or standard method. As the standard method is much slower than optimised method, the package size of the standard method should be smaller than that used for the optimised method to balance the calculation.

#### 27.4.11 Calculating an Individual Contingency

To calculate an individual contingency, click on the **Show** button in the contingency analysis command dialog (see Figure 27.4.1) to open the list of contingencies included in the analysis. From here the user can right-click on a contingency of interest, and select *Execute* from the context menu. Additionally, the corresponding element can be marked in the single line graphic by right-clicking on the contingency object in the list and selecting *Mark in Graphic* from the context menu.

#### 27.4.12 Representing Contingency Situations Contingency Cases

Contingency cases (*ComOutage* objects) are objects used in *PowerFactory* to define contingency situations within the analysed networks. A contingency case determines which components are put on outage. When a contingency analysis (*ComSimoutage*) is executed, the contingency analysis command considers each of the contingency cases stored inside it, taking the corresponding components out of service and performing a contingency load flow.

As mentioned previously, the contingency cases used by a specific contingency analysis command are stored inside the command itself. Contingency cases are created either by using *Fault Cases* and/or *Fault Groups* (see Section 27.8), or via the *Contingency Definition* command (grid icon, see Section 27.1). Once the contingencies have been defined in the contingency command, the cases can be viewed by using the **Show** button available in the dialog (see Figure 27.4.1). Additionally, the contingency cases within the active study case's contingency analysis command may be viewed by clicking on the *Show Contingencies* icon (grid icon), located on the main toolbar (only available when the *Contingency Analysis toolbar* is selected). In both cases a new data browser showing the defined contingencies is opened,

with the contingencies listed inside. By double-clicking on a contingency from the list, the corresponding dialog for that particular contingency is opened (as illustrated in Figure 27.4.3). The dialog displayed in Figure 27.4.3 shows the following fields:

- **Name.** Name of the contingency case.
- **Not Analysed.** If enabled, the case is not considered by the contingency analysis command.
- **Number.** An identification number given to the contingency and which is stored in the results. This number can be used for reporting purposes.
- **Fault Case.** Reference to the fault case (if any) from where the contingency case originated.
- **Fault Group.** Reference to the fault group (if any) from where the contingency case originated. This field is only available if the contingency case has an associated fault group.
- **Events Used for this Contingency** As shown in Figure 27.4.3, the user can specify whether to generate the events based on the fault case definition (automatically), or to use locally defined events. If the user chooses to use locally defined events, then the *ComOutage* object which defines the contingency (located in contingency command of the study case) can be modified independently.
- **Interrupted Components.** This is a table showing the components put on outage by the contingency case. The table, which is read-only, is automatically generated when the contingency case is created.
- **Fault Type.** Displays the fault type and the contingency order. See Figure 27.8.1.
- **Contingency Analysis.** Reference to the contingency analysis command where the contingency case is stored.

The **Mark in Graphic** button highlights the interrupted components in the single line diagram.

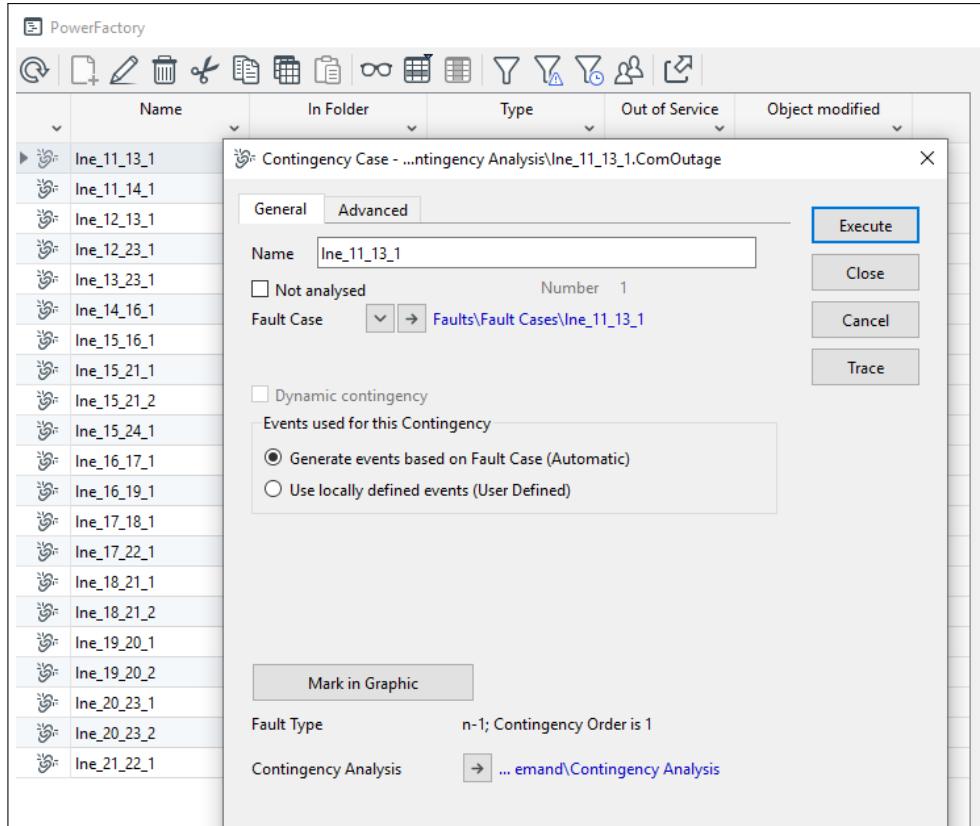


Figure 27.4.3: Contingency Cases (*ComOutage* objects)

Normally, contingency cases (*ComOutage* objects) are analysed by the contingency analysis command (*ComSimoutage*) in which they are stored. However, each contingency case provides the functionality of a command itself, and can be executed individually using the **Execute** button at the top right of the *ComOutage* dialog. In this case the actions taken by the circuit breakers, which must switch to clear the fault, are shown in the single line graphic (only if the contingency case was created using fault cases/groups).

---

**Note:** The 'Interrupted Components' table is updated by the program each time the contingency analysis is executed.

---

For further information on contingency cases generated using fault cases and/or fault groups, refer to Section 27.8 (Creating Contingency Cases Using Fault Cases and Groups). For information on contingency cases created using the *Contingency Definition* (*ComNmink*) command, refer to Section 27.7.1 (Creating Contingency Cases Using the Contingency Definition Command).

## 27.5 Reporting Results

### 27.5.1 Predefined Reports

The built-in Contingency Analysis reports are by default presented in tabular format, although many are also available as ASCII reports. They are accessed via the *Contingency Analysis Reports* button (Excel icon) in the Contingency analysis toolbar. This brings up the following dialog:

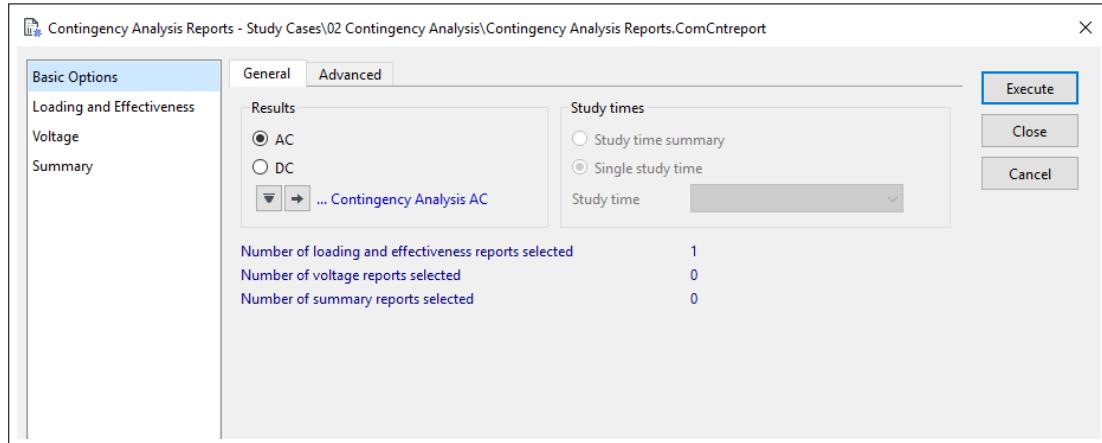


Figure 27.5.1: Contingency Analysis Reports dialog

#### 27.5.1.1 Basic Options page

On this page, AC or DC results can be selected according to the calculations executed, and there is a link to the relevant result file. The reports are selected on the other pages, so here on the Basic Options page there is an overview of which reports are to be generated.

#### Timesweep reports

If the Time Sweep option has been used (see Section 27.4.5), the panel called *Study times* will become active. Two options are available:

- **Study time summary:** will result in reports containing the results from all study times; for each result, the Study Time as well as the contingency is displayed. In addition, once the report has been generated, the user may also select the individual study times from within the report.
- **Single study time:** provides a drop-down list of the study times, from which required study time can be selected.

The following reports are available when the *study time summary* option is selected:

- Maximum Loadings
- Voltage Steps
- Maximum Voltages
- Minimum Voltages
- Non-convergent Cases

### Advanced tab

On the Advanced tab, the user can change the format of the reports from Tabular to ASCII. Note that some reports are not available in ASCII format.

#### 27.5.1.2 Loading and Effectiveness page

##### Loading Reports

- **Worst loading violations:** reports the greatest loading violation for each component (according to the specified loading limit), considering all contingencies. Any such component is reported only once, i.e. it is reported for the contingency causing this violation.
- **All loading violations:** all overloaded components (according to the specified loading limit) for each contingency are displayed in a single list.
- **Loading violations per case:** all overloaded components (according to the specified loading limit) for each contingency are displayed in separate lists (i.e. one list per contingency case).

A typical tabular report is shown below in Figure 27.5.2. Note the options to export to html or Excel and the fields for selecting limits. In the individual columns, the usual filtering and sorting options are available. Within the table, references to objects can be used to edit the object itself or mark it in graphic.

Component	Branch, Subs... or Site	Loading Continuous [%]	Loading Short-Term [%]	Loading Base Case [%]	Contingency Number	Contingency Name	Base Case and Continuous Loading [0,0 % - 156,3 %]
► 1 NE-SW_L1		156,3	156,3	37,0	26	⌚ Double West Interco...	
2 NE_L4		153,4	153,4	72,8	23	⌚ NE_L3	
3 NE_L4		108,5	108,5	72,8	25	⌚ NE_L5	
4 NW-SW_L1		125,5	125,5	70,6	6	⌚ NW-SW_L2	
5 NW-SW_L1		93,3	93,3	70,6	20	⌚ NE-SW_L1	
6 NW-SW_L2		125,5	125,5	70,7	5	⌚ NW-SW_L1	
7 NW-SW_L2		93,5	93,5	70,7	20	⌚ NE-SW_L1	
8 NE_L5		109,1	109,1	36,9	24	⌚ NE_L4	

Figure 27.5.2: Tabular Report of Loading Violations

## Effectiveness Reports

- **Generator effectiveness:** generators having an effectiveness greater than or equal to the specified value (%/MW) are displayed in a single list. The *Delta P* indicates whether generation increase or decrease reduces the overload. A (+) means that the generation needs to be increased whereas a (-) indicates a need to decrease. The *Max. Effectiveness* gives the maximum change in loading of the overloaded element possible for this generator with regard to its limits.
- **Quad-booster effectiveness:** Quad-booster transformers (phase-shifters) having an effectiveness greater than or equal to the specified value (%/tap) are displayed in a single list, with an indication of which direction of tap change reduces the overload.

## Filters

- **Branches: report highest loading only:** this can reduce the amount of reporting when the network contains many branches consisting of multiple line elements.
- **Suppress contingency violation if base case is violated:** This is typically used to filter out overloads which are inherent in the network and are of little interest.
- **Loading threshold:** results for elements loaded at or above this limit will be reported (subject to the above two filters).

### 27.5.1.3 Voltage page

#### Voltage Reports

- **Voltage steps:** all voltage deviations of terminals (between the base case and the contingency case) for each contingency are displayed in a single list. Reports the highest voltage deviation of terminals (between the base case and the contingency case) considering all contingencies. Any such terminal is reported only once. Only terminals with the highest voltage deviation greater than the specified maximum voltage step are reported.
- **Voltage angle step:** all voltage angle deviations of terminals between the base case and the contingency case are displayed in a list. Keep in mind that the voltage angle steps limit of the report must be higher as the limit for the recording filter within the calculation command (see Section 27.4.2.3) to receive a correct report.
- **Voltage violations per case:** all busbars with exceeding voltage (maximum or minimum) are displayed in separate lists.
- **All voltage violations, Maximum voltage:** reports all voltage violations of a terminal (greater than or equal to the specified upper voltage limit) considering all contingencies.
- **All voltage violations, Minimum voltage:** reports all voltage violations of a terminal (less than or equal to the specified lower voltage limit) considering all contingencies.
- **Worst voltage violations, Maximum voltage:** reports the greatest voltage violation of a terminal (greater than or equal to the specified voltage limit) considering all contingencies. Any such terminal is reported only once (i.e. it is reported for the contingency causing this violation).
- **Worst voltage violations, Minimum voltage:** reports the greatest voltage violation of a terminal (less than or equal to the specified voltage limit) considering all contingencies. Any such terminal is reported only once (i.e. it is reported for the contingency causing this violation).
- **Asynchronous busbars next to fault:** reports all contingency cases leading to a voltage angle difference of the connected busbars above the defined asynchronism angle. For each contingency the highest angle difference for the connected busbar during the contingency is shown as well as the angle difference in the base case. If no angle difference for the base case is shown it might be the case that the voltage angle was not recorded and need to be added to the variable selection. This option is just visible if *Record asynchronism of busbars next to the fault* is selected on the recording filters page (see Section 27.4.2.3).

## Filters

- **SUPPRESS CONTINGENCY VIOLATION IF BASE CASE IS VIOLATED:** This is typically used to filter out voltage violations which are inherent in the network and are of little interest.

### 27.5.1.4 Summary page

- **Non-convergent cases:** the non-convergent cases of the contingency analysis are listed.
- **Summary variables:** If the option *record additional result variables* has been selected as described in Section 27.4.2 above, this report can be used to see the summary of outcomes. In addition this summary also shows the number of contingencies causing a critical asynchronism in case the according recording filter was selected.
- **Summary variables per contingency:** Likewise, the summary variables can be shown at contingency level. For contingencies which cause loss of generation and/or demand, the report also includes the number of generation/load objects lost, together with the total MW and Mvar loss. For each contingency that does not include a transformer the maximal voltage angle difference between the interrupted busbars is reported.

### 27.5.2 Customised reports

Although the tabular reports are already predefined, the user can modify them if required. The report formats are accessed via the Format tab on the relevant page of the *Report Contingency Analysis Results* dialog and using the right-arrow icon.

Alternatively, users can write their own reporting scripts, which directly access the results in the *ElmRes* results file.

## 27.6 Trace Function for Multiple Time Phase and/or RAS

The Trace functionality allows the user to visualise the changing system state calculated in a Multiple Time Phase contingency analysis, or in a contingency analysis which incorporates Remedial Actions Schemes (RAS), using either Single or Multiple Time Phase.

First the Contingency Analysis command is configured. Then the trace is initiated using the *Start Trace* button (▶) on the Contingency Analysis toolbar. When this button is pressed, a dialog opens allowing the user to select a contingency. Following the selection of a contingency by the user and pressing **OK**, the contingency dialog is closed and the base case load flow is executed. The execution of the first event(s) and all subsequent event(s) is triggered by pressing the *Next Time Step* button (▶) on the main toolbar. At each time step the load flow calculation results and the state of the network circuit breakers are displayed in the single line graphic. It should be noted that the *Next Time Step* evaluates events according to their time of occurrence, and not according to the time phases defined in the *Contingency Analysis* command. After the last time event(s) have been executed, the *Next Time Step* button becomes inactive. The *Stop Trace* button (□) can be pressed to clear the calculation. Alternatively, the **Trace** button in each *ComOutage* dialog can be used to initiate the Trace for that particular contingency.

Note that the Trace Function is only available for static contingencies, not dynamic contingencies (see Section 27.2.1.2)

## 27.7 Creating Contingencies

There is an important distinction to be made between contingencies which use fault cases and those which do not:

If a contingency does not use a fault case, the “faulted” equipment is simply taken out of calculation. If there is a fault case, the faulted equipment can be removed through the operation of circuit breakers (either explicitly or by using in-built topology tracing). Consider the example in Figure 27.7.1 below, where a fault on a transformer is being modelled. For realistic modelling, the use of fault cases is generally recommended.

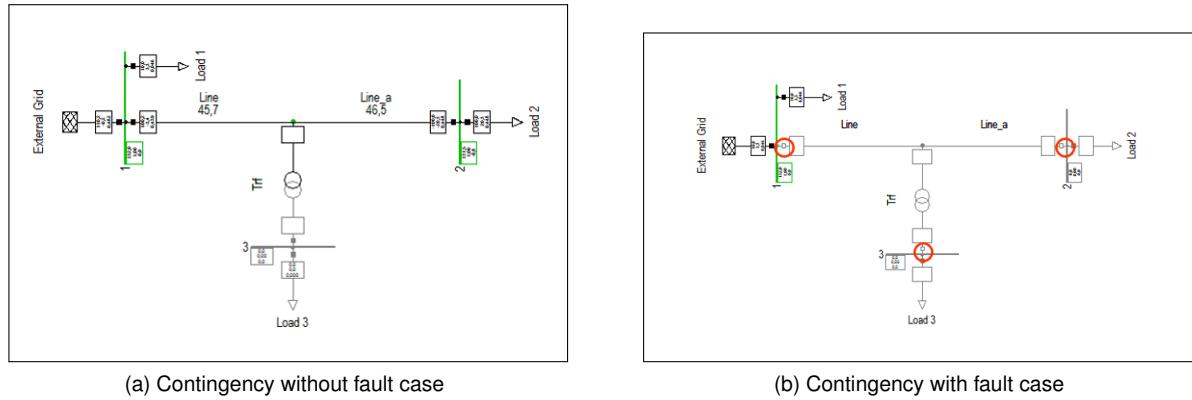


Figure 27.7.1: Use of fault cases in contingency analysis

### 27.7.1 Creating Contingencies Using the Contingency Definition Command

The *Contingency Definition* command (*ComNmink*) is used to automatically generate contingency cases based on selected components. It is accessible via the *Contingency Analysis* toolbar using the button. The *Contingency Definition* command can be used to automatically generate contingency cases either for the complete system or from pre-defined sets of elements.

The *Contingency Definition* command offers the following options:

#### 27.7.1.1 Creation of Contingencies

To create contingencies as opposed to fault cases, this option is selected:

- **Generate Fault Cases for Library.** Generates fault cases which are stored in the Fault Cases folder (Operational Library, Faults) as explained in 27.8. In case the regarding fault cases already exists a warning is printed out.
- **Generate Contingency Cases for Analysis.** Generates contingencies which are stored in the contingency analysis command.

#### 27.7.1.2 Outage Level

- **n-1.** Creates single cases for each of the selected components.
- **n-2.** Creates cases for every unique combination of two selected components.
- **n-k cases of mutually coupled lines/cables.** Creates cases for every set of mutually coupled lines/cables. If for example, three lines are modelled as having a mutual coupling, by selecting this option a fault case is created considering the simultaneous outage of the three coupled lines.

### 27.7.1.3 Network Components

There are three main options if selecting Network Components:

- **Whole System** Cases generated for the whole network, with options to choose which element classes should be considered.
  - **Lines/cables.** Cases according to the selected outage level will be generated for all lines and cables (*ElmLne* objects) in the system.
  - **Transformers.** Cases according to the selected outage level will be generated for all transformers (*ElmTr2*, *ElmTr3*, *ElmTr4* objects) in the system.
  - **Generators.** Cases according to the selected outage level will be generated for all synchronous generators (*ElmSym* objects) in the system.
  - **Series Capacitors.** Cases according to the selected outage level will be generated for all series capacitors (*ElmScap* objects) in the system.
  - **Series Reactors.** Cases according to the selected outage level will be generated for all series reactors (*ElmSind* objects) in the system.
- **Selection.** This option allows the user to use a set of elements. Such sets are stored in the Study Case.
- **Filtered Elements.** This option allows the user to select elements according to a user-defined filter. Using this option, it is possible for example to run a load flow and select elements based on load flow results such as loading.

The selection of elements to outage in the *Contingency Definition* command can also be created by the use of DPL scripts. Refer to the *ComNmink* methods in the [DPL Reference](#).

When the *Contingency Definition* command is executed and the option “Generate Contingency Cases for Analysis” is selected, it generates the corresponding contingency cases according to the options and elements selected. The *Contingency Analysis* command, which is automatically created inside the current active *Study Case* is then automatically opened, and the analysis can be run. Note that when a new list of contingencies is created using the *Contingency Definition* command, the previous content of the contingency analysis command is overwritten.

### 27.7.1.4 Outputs

#### Fault case folder

If the *Generate Fault Cases for Library* option is selected on the *Basic Options* page, the fault cases will be stored in a *Fault Cases* folder (*IntFltcases*) within the project's operational library. If more than one folders of the class *IntFltcases* exist, the following options are available on the *Outputs* page:

- Prompt user to select: a window with the available *IntFltcases* folders will be shown for one to be selected.
- Predefined folder: an *IntFltcases* folder can be selected a the predefined one.

#### Naming of created contingencies or fault cases

The newly created contingency or fault case is automatically given a name. There are two options available for the naming:

- Use name(s) of outaged element(s)
- Use site/substation/branch name in addition to outaged element(s) name(s)

## 27.7.2 Creating Contingencies Using Fault Cases and Groups

If fault cases are to be used to create contingencies, this can be done in various ways:

1. First create fault cases then use them to populate the Contingency Analysis Command **or**
2. Select one or more objects and right-click, *Calculation* → *Contingency Analysis...* **or**
3. Select one or more objects and right-click, *Calculation* → *Execute single contingency*

The creation and management of Fault Cases is described in detail in Section [27.8](#)

For Option 1, the required fault cases are selected within the Contingency command dialog, as described in Section [27.4.1.2](#).

Option 2 enables one to populate the Contingency Analysis command with fault cases for the elements of interest. If relevant fault cases already exist in the project, they will be selected. For elements for which no fault cases exist, they will be created in the Faults, Fault Cases folder of the Operational Library. To use the function, the element(s) are selected, then right-click, *Calculation* → *Contingency Analysis...*. The Contingency Analysis command will be presented, already populated with the required fault cases, and the analysis can be run.

Option 3 offers the possibility of executing a single contingency case directly from a selected element or element(s) without going via the contingency command dialog at all. This is slightly different from in Section [27.4.11](#), where the contingency has already been created and is being selected to be run on its own. But the visualisation of the results is the same.

The purpose of the option described here is to be able to select an element or group of elements and execute a relevant fault case. A fault case, possibly containing additional post-fault actions, may already exist in the library and this is an easy way to find it and run it.

To use the function, the element(s) are selected, then the user should do right-click, *Calculation* → *Execute Single Contingency*. If a suitable fault case exists in the Operational Library it will be executed (if there is more than one, the user may choose which to use); if no suitable fault case exists then one will temporarily be created and executed but not retained afterwards. In either case, the post-fault results from the contingency are available on the graphics or via element filters.

## 27.7.3 Creating Dynamic Contingencies

Dynamic contingencies are contingency cases which are not defined before executing the contingency analysis, but are generated “on the fly” by the contingency command, using criteria which have been supplied by the user. It is possible to use both the normal static contingencies and dynamic contingencies in one Contingency Analysis calculation.

The *Dynamic contingencies* option is selected on the basic data page and the user then defines one or more filters.

(Note that even if more than one filter is used, the numbers in the *Order* column are not relevant.)

The default is a filter based on the loading in the base case load flow. During the contingency analysis, contingencies are then created for any element which meets at least one of the criteria. To avoid confusion with static contingencies of the same name, dynamic contingencies will be named after the element but prefixed with an underscore (e.g. *\_Line01*).

After running the analysis, the user can make use of the *Show contingencies* button on the Contingency Analysis toolbar to see the contingencies, both static and dynamic (if any are created).

A *Fault location* field provides a link to the object which was faulted in the contingency.

In addition, information about the contingencies created will be seen in the output window.

For the purposes of reporting, the dynamic contingencies are treated exactly the same way as static contingencies.

Dynamic contingencies are particularly useful for contingency timesweep (see [27.4.5](#)), as the contingency requirements may vary with time.

## 27.8 Fault Cases and Groups

Contingency cases created from fault cases can be regarded as contingency situations produced in a network as a consequence of the clearing of a fault. Fault cases without switching events (created following the procedure described in Chapter 14: Libraries, Section [14.6.4](#): Fault Cases and Fault Groups) are used to automatically generate contingency cases in the contingency analysis command, by pressing the **Add Cases/Groups** button and selecting the desired objects from the data browser that pops up.

For every selected fault case, the calculation automatically detects which circuit breakers must open in order to clear the defined fault(s). All components which lose their connection to the network reference bus following the switching actions that clear the fault(s), are regarded as 'interrupted' and are subsequently added to the *Interrupted Components* table of the corresponding contingency case. In other words, these components are put on outage by the contingency case. Depending on the fault defined in the fault case that generates a contingency, the *Fault Type* field in the contingency case dialog (Figure [27.8.1](#)) is set to:

- Busbar fault:**

If the contingency originates from a fault on a busbar

- n-k fault:**

With contingency order equal to  $k$  (where  $k \geq 0$ ).  $k$  corresponds to the number of network regions (sets of topologically connected components) which are disconnected during a fault, by the switching actions performed. It should be noted that the switching actions which are considered depend on the post contingency time used by the update (this time differs between single- and multiple time phase analysis).

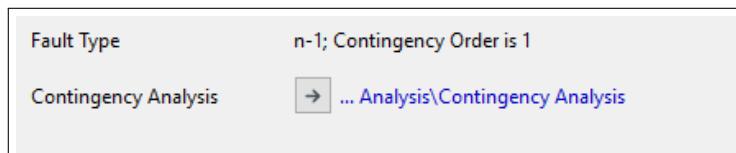


Figure 27.8.1: *Fault Type* Field in the Contingency Case (*ComOutage*) Dialog

**Note:** In *PowerFactory* an interrupted component is a network primary element that is energised before a fault and de-energised afterwards. A component is considered to be energised if it is topologically connected to a network reference bus. A region is defined as a set of topologically connected components. Like components, regions can have energised, de-energised and interrupted states, depending on their connection to a network reference bus.

Contingency cases can be created from fault cases/groups, which reside in the *Operational Library*, by pressing the **Add Cases/Groups** button in the contingency analysis command (see Section [27.4.1](#) (Basic Options) and Figure [27.4.1](#)). In the case of creating contingencies from fault group(s), a contingency case will be generated for each fault case referred to in the selected fault group(s).

**Note:** The 'topological search' algorithm used by the program to set contingency cases from fault cases requires the explicit definition of at least one reference bus in the analysed system. A bus is explicitly set as a reference if it has connected to it either a synchronous generator (*ElmSym*), or an external grid (*ElmXnet*) with the option 'Reference Machine' enabled (available on the element's 'Load Flow' tab).

---

### 27.8.1 Browsing Fault Cases and Fault Groups

There are two types of subfolders inside the *Faults* folder in the *Operational Library*: *Fault Cases* and *Fault Groups*.

In order to make a new folder of either of these types, left-click on the *Faults* folder and then press the *New Object* button (⊕) on the Data Manager toolbar. Select whether a new *Fault Cases* or *Fault Groups* folder should be created.

The *Fault Cases* folder holds every contingency (n-1, n-2, or simultaneous) defined for the system, as described in Section 27.8.2 (Defining a Fault Case). Alternatively, several fault cases can be selected and stored in a *Fault Group*, as described in Section 27.8.5 (Defining a Fault Group).

### 27.8.2 Defining a Fault Case from Network Element(s)

To define a fault case for an element in the grid, select it in the single-line diagram. Then right-click and choose one of: *Operational Library* → *Fault Cases* → *New Single Fault Case* or *Operational Library* → *Fault Cases* → *New Multiple Fault Cases, n-1* (or *New Multiple Fault Cases, n-2*) or *Operational Library* → *Fault Cases* → *New Mutually Coupled Lines/Cables, n-k*.

If *Multiple Fault Cases, n-2* is selected, fault cases will be created for the simultaneous outage of every unique combination of two elements in the selection. If the user selects *New Single Fault Case*, a fault case will be created for the simultaneous outage of all elements in the selection.

If *Mutually Coupled Lines/Cables, n-k* is selected, then fault cases will be created for the simultaneous outage of each coupled line in the selection.

Alternatively, a filter can be used. This can be done (for example) with the help of the *Open Network Model Manager...* button (grid icon), to list all elements for which outages are to be defined. These elements can then be highlighted and the user can then right-click on the highlighted selection and choose (for example) *Operational Library* → *Fault Cases*. The *Simulation Events/Fault* dialog opens, as shown in Figure 27.8.2, where the user can enter the desired name of the fault case in the *Name* field.

On the Advanced tab of the *Basic Data* page of the same dialog, the user can create the corresponding switch events, by clicking on the **Create** button.

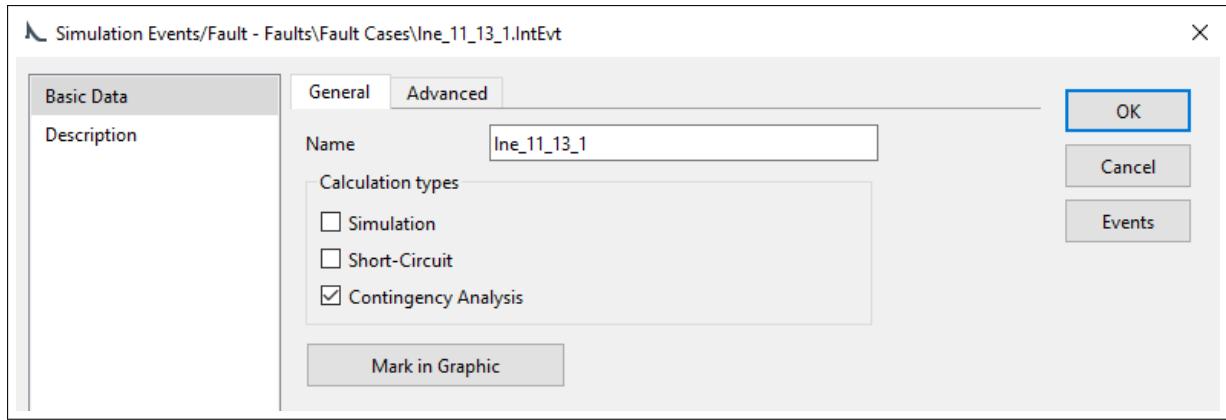


Figure 27.8.2: Creation of Fault Case (*IntEvt*)

For further background on fault cases, refer to Chapter 14: Libraries, Section 14.6.4 (Fault Cases and Fault Groups).

### 27.8.3 Defining Fault Cases using the Contingency Definition Command

Another way of creating a set of fault cases is to use the Contingency Definition Command.

In Section 27.7.1, the use of this tool to either create fault cases or contingencies without fault cases is described.

### 27.8.4 Representing Contingency Situations with Post-Fault Actions

As a default, if a fault case is created for a fault it will just contain one or more short circuit events. In this case, the Contingency Analysis uses a topological search to find which breakers need to be opened to isolate the faulted elements.

But a fault case can define such switch operations explicitly and can also include additional switch actions and other events to represent post-fault actions.

The types of events which are allowed in the contingency analysis as post-fault actions are:

- Load Event (*EvtLod*)
- Dispatch Event (*EvtGen*)
- Switch Event (*EvtSwitch*)
- Tap Event (*EvtTap*)
- Power Transfer Event (*EvtTransfer*)

The list of Events is displayed by clicking on the **Events** button in the fault case (*IntEvt*) dialog. Events can be edited and/or deleted and new events can be created by using the *New Object*  icon at the top of the opened browser window.

If the post-fault actions which are going to be created include switch events, it is very important to understand how these are handled within when a fault case is executed as a contingency:

---

**Note:** By default, a simple fault case created for a circuit, for example, will contain just a short-circuit event for the relevant element(s). When the contingency is executed, topology tracing is used to determine which circuit breakers need to be opened in order to isolate the faulted elements.

The user has an option to create these events within the fault case by using the **Create** button on the Advanced tab. This is not normally needed but can be useful if the user wishes to change the events in order to model some different switching behaviour.

The default behaviour of *PowerFactory* is to assume that if there are switch events present in the fault case, the user wants *only* these switch events to be executed and no others. In other words, it does not execute any additional switch events in order to isolate the elements. This means that if the user has a fault case which contains just a short-circuit event, and then adds a single open-switch event (to represent a post-fault action), this then would be the only switch to open. This of course is not desirable.

So, if the user wants to ensure that the faulted element is isolated *and* the extra switch event is considered, there are two possible approaches:

- Use the **Create** button to create all the events needed to isolate the faulted equipment (and adjust if required), then add the extra switch event.  
*or*
  - Do not create the events, and instead select the “Always create switch events to clear the faults” option on the Advanced tab of the fault case. Then the correct breakers will be opened to isolate the faulted equipment *and* the post-fault actions executed.
- 

Contingencies are created based on fault cases defined in the *Operational Library*. These fault cases define the location of the fault events, and *may* also define post contingency actions taken to isolate the fault and mitigate the effects of the outage of the component(s). Whenever a new contingency is created, a link from the *ComOutage* object to the fault case is set. New contingencies can be created in a *Contingency Analysis* command by clicking on the **Add Cases/Groups** button in the *Configuration* section of the *Basic Data* page (see Section 27.4.1: Basic Options).

### 27.8.5 Defining a Fault Group

To define a fault group, left-click on the *Fault Groups* folder. Then click on the *New Object* button (). A *Fault Group* dialog will be displayed. In this dialog the user can specify the name of the fault group in the *Name* field, and add fault cases to this new group using the **Add Cases** button. Click the **Cases** button to view existing cases (if any) in the fault group.

**Note:** When a fault group is defined and fault cases are added to it, a reference is created to each of these fault cases. The fault case itself resides in the *Fault Cases* subfolder. This means that if an item in the fault group is deleted, only the reference to the fault case is deleted. The fault case itself is not deleted from the *Fault Cases* subfolder.

---

## 27.9 Comparing Contingency Results

In order to compare contingencies in a fast and easy way, *PowerFactory* provides a *Contingency Comparison* function (). The *Contingency Comparison* function is only enabled if the user has previously defined the contingency cases in the *Contingency Analysis* command, as explained in sections 27.7.1 and 27.7.2. The general handling of the Contingency Comparison function is as follows:

1. Define the contingency cases in the *Contingency Analysis* command (see sections 27.7.1 and 27.7.2).
2. Click on the *Contingency Comparison* button (). A window will pop up allowing the user to select the required contingency cases (Figure 27.9.1). The selection can correspond to one, several, or all contingency cases.

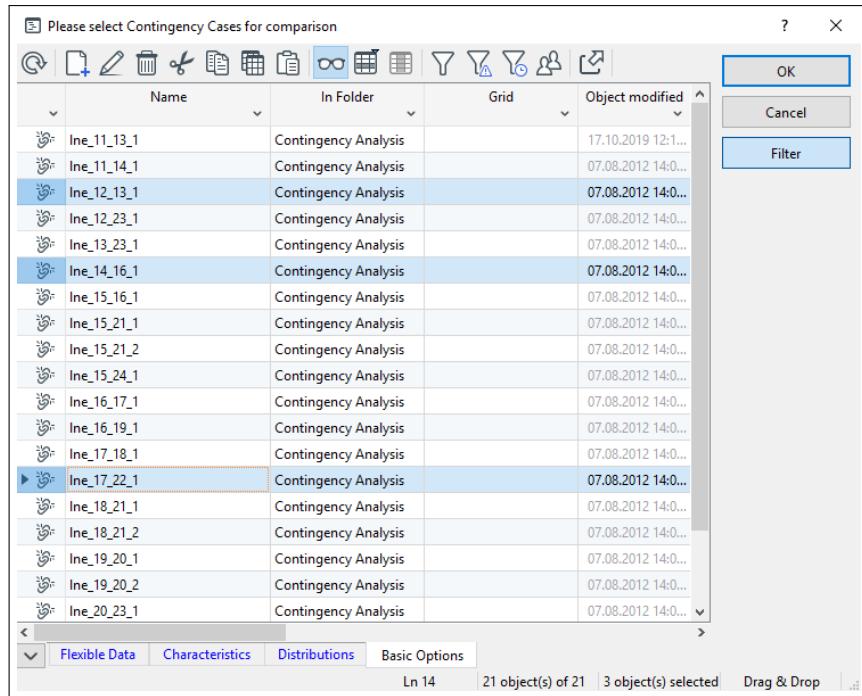


Figure 27.9.1: Selection of Contingency Cases for Comparison

3. By clicking on the **OK** button, the *Comparing of Results On/Off* button (Figure 27.9.2) is enabled and the selected contingency cases are automatically executed.

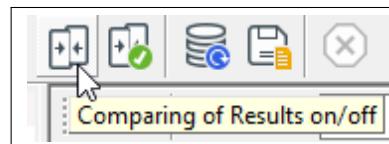


Figure 27.9.2: Comparing of Results Button

4. The single line graphic result boxes will display the results, based on the comparison mode and the two compared cases. By default, the comparison is made between the Base Case and the last selected contingency case in the list.
5. To change the comparison mode and/or the cases to be compared, click on the *Edit Comparing of Results* button (Figure 27.9.2). The *Compare* dialog will pop up displaying the current settings. To change the cases to be compared, click on the black arrow pointing down (▼) and select a different case (Figure 27.9.3).

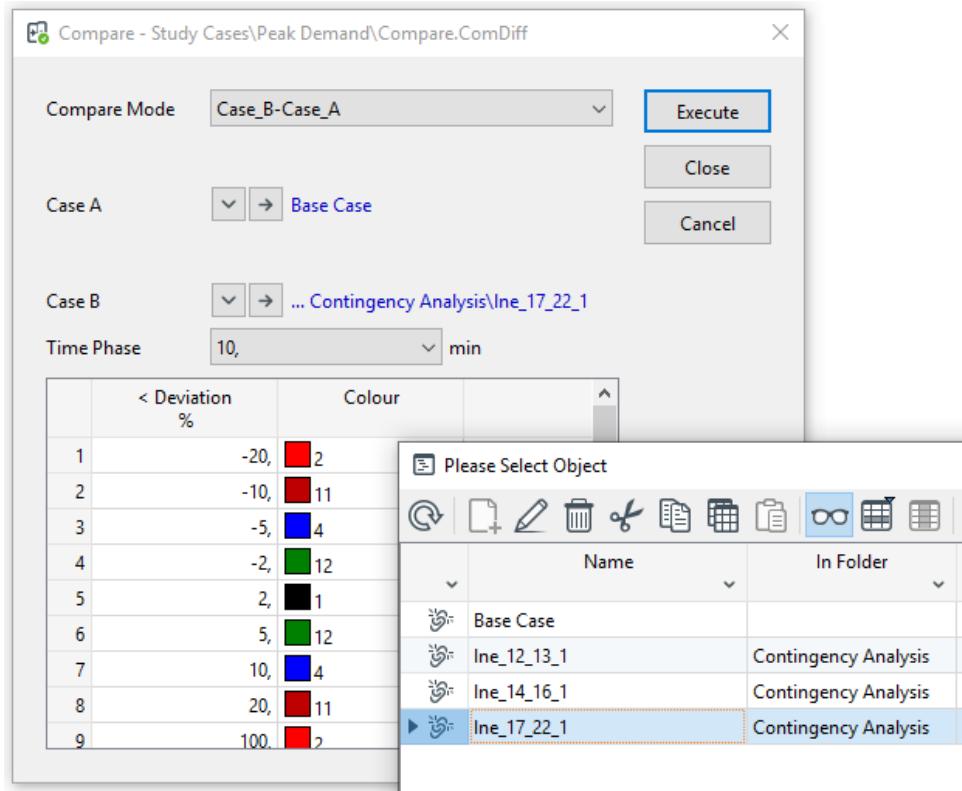


Figure 27.9.3: Selection of other Cases for Comparison

6. If the contingency analysis is defined with time phases, the compare dialog will have the option of selecting the time phase.
7. Once the calculation is reset (for example by either making changes in the model or by clicking on the *Reset Calculation* button), the comparison mode will be disabled.

Note that the comparison of contingency results feature is only available for static contingencies, not dynamic contingencies (see Section 27.2.1.2)

## 27.10 Managing variables to be recorded

In the Study Cases chapter, Section 13.11, there is a description of how results variables are managed, using the *ElmRes* object. For contingency analysis, a minimum set of variables for each element class is recorded, which is detailed in each relevant variable selection *IntMon* object, but it is possible to add additional variables to these *IntMon* objects if required.

The Variable Selection object is described in Section 19.3: [Variable Selection](#).

A further option to control the results being recorded is to use a user-defined filter within the *ElmRes* object, as described below.

### 27.10.1 Using filters to enable selective results recording

Sometimes the user may only be interested in results for part of the network being analysed and so wishes to record results selectively, based on the region of interest. For this purpose, it is possible to introduce a filter into the contingency analysis *ElmRes* object. An advantage of this approach, particularly with large networks, is that it reduces the size of the results file.

Results to be recorded may be filtered according to Grid, Area, Zone or Boundary. To create a filter, select the *ElmRes* object in the study case and then press the *New Object* icon. From the list, select *General Filter (SetFilt)*.

Within the new filter, the Object Filter should be set to all elements. If results are to be filtered for a specific grid, this can be selected directly using the down-arrow next to “Look in”. If an Area, Zone or Boundary is to be used, first select the Boundary, Zone or Area in a Network Model Filter or Data Manager, and copy it. Then within the filter, use the down-arrow next to “Look in” and select Paste.

Buttons are available on the *Recording of Results* page of the Contingency Command dialog, to allow easy access to the *ElmRes* object.

## 27.11 Remedial Action Schemes (RAS)

An important analysis requirement for transmission system operators in particular is to study the management of the network when a fault occurs and post-fault actions have to be taken. One way to do this is to create post-fault actions by adding events to the relevant fault cases, as described in section 27.8.4. Using this approach, however, has its limitations: firstly, the post-fault action will always be executed for these fault cases and secondly, if the same post fault action is appropriate for many contingencies, it must be modelled separately for each.

The Remedial Action Schemes functionality takes a different approach. The concept is that there is a library of Remedial Action Schemes (RAS), each of which consists of one or more trigger conditions and one or more events which model remedial actions. The RAS are selected as required in the Contingency Analysis command and then the remedial actions are executed for every fault case which meets the trigger conditions.

### 27.11.1 Creating a RAS object

RAS are stored in the “Remedial Action Schemes (RAS)” folder of the Operational Library. When a new RAS is created, the user will specify one or more trigger conditions and one or more events to represent the remedial action(s). All triggers and remedial actions which have been created for a particular RAS are stored as contents in the RAS and can be selected or not as required.

As an example, these are the steps required to create a simple RAS to model generation reduction for a post-fault overload on a line:

Select the Contingency Analysis toolbar then click on the  icon (Show Remedial Action Schemes). In the RAS subfolder, use the *New Object* icon to create a new *Remedial Action Scheme (IntRas)*. The RAS object is created and will automatically be given a unique Sequence number, used to determine the order of execution of RAS, should more than one be triggered for a particular contingency. These sequence numbers can be changed by the user.

On the left-hand side of the IntRas dialog box, click on the Create Condition button. Use the drop down arrow next to *Element selection* to select the line whose overloading is to act as a trigger. Using the drop-down menu, select the *Type of Condition* as *Loading continuous*. The percentage loading can be selected as required and the *Check condition* left as *Post-Fault*.

Now the Remedial action is created, for which the sequence of actions is slightly different. Firstly, on the RAS dialog, select the type of event. For this example, it will be *Dispatch Event*. Then press the Create button. In the new dialog box, select the generator which is to provide the remedial action and the required change in active power, then press OK.

### 27.11.2 Trigger Conditions

Below is a list of possible trigger conditions. The trigger condition must be appropriate for the element selected, otherwise an error message will be generated. The dialog box for the trigger also includes a Check button for validating the trigger condition which has been set up.

- Energising status
- Switch status
- Voltage
- Voltage step
- Voltage angle step
- Active power flow
- Loading (continuous)
- Boundary flow
- User-defined

The listed options are provided in order to enable users to easily set up triggers that are expected to be most commonly used, but the User-defined option allows for more customer-specific requirements.

For some of the triggers the user can select if the condition is for the pre-fault or the post-fault condition. In case of User-defined conditions also the difference between pre-fault and post-fault value can be used as a trigger.

### 27.11.3 Logical combinations of triggers

If more than one trigger is created, they can be combined using an *And* or an *Or* operator. For more complex combinations, the user can create logical gates using the *Create gate* button, and these also allow criteria to be negated. Logical gates can be nested.

### 27.11.4 Remedial actions

Below is the list of possible Remedial actions, defined in terms of events. These are the standard events allowed in Contingency Analysis fault cases. It is possible to define execution times for the events; this will be taken into account in conjunction with the time phase of the analysis.

- Switch event
- Dispatch event
- Tap event
- Power transfer event
- Load event

---

**Note:** A RAS may contain a number of triggers and remedial actions, but only those selected in the dialog box are active.

---

### 27.11.5 RAS groups

RAS objects are stored in the Remedial Action Schemes (RAS) folder of the Operational Library. If an existing project does not already have such a folder, it will be created automatically if the user creates a RAS using the process described above. There are two subfolders: the RAS subfolder where the RAS objects themselves are stored, and the RAS Groups folder, where groups of references to RAS objects may be created. These RAS groups are analogous to the Fault Groups in the Faults folder (see Section 27.8.5) and are handled in a similar way.

### 27.11.6 Using Remedial Action Schemes in Contingency Analysis

RAS objects are not fault-case specific, but will take effect for any contingency which meets the trigger conditions. Should the user wish a RAS to be fault-case-specific, this can be done by including a relevant trigger (such as circuit breaker operation or change in energising status).

Because the RAS are not fault-case-specific, users may want to be selective about which RAS are included in their calculations. One way to do this is to make use of the Out of Service flags on the RAS objects themselves; this flag will be observed in all study cases.

However, the more usual way to determine which RAS objects are used during the Contingency Analysis is to select the RAS and/or RAS Groups via the Contingency Analysis command dialog. This RAS selection is then defined for the particular study case.

The relevant parameters for executing RAS are:

**Consider Remedial Action Schemes (RAS)** (Basic Data page)

If this option is enabled, all selected RAS (unless Out of Service) will be applied during the Contingency Analysis calculation.

**Show triggered RAS for each contingency in the output window** (Output page)

If this option is enabled, a message will be output each time a RAS is triggered during the analysis. The message includes the name of the RAS as a hyperlink.

**Selection of RAS** works in exactly the same way as the selection of Fault cases, with options to remove, add and view the selected RAS.

### 27.11.7 Results and reporting

It is important to understand the way in which the Contingency Analysis handles the RAS and exactly what results are available to the user after the calculation, either via the standard reports or directly from the results files for customised reporting scripts.

#### 27.11.7.1 Single Time Phase

With the single time phase calculation, the results which are recorded in the results file are always the results of the contingency analysis after all triggered remedial action events have taken effect. This is the case regardless of whether the triggers can be determined prior to the fault case execution (topological criteria) or are dependent upon analysis results.

With the *Post-Contingency End of Time Phase* option selected, the remedial actions executed will take into account the execution times of the remedial action events: if this is longer than the specified Time Phase time, the remedial action event will not be taken into account. If a *Post-Contingency End of Time*

Phase time has not been specified, all events in a triggered RAS will take place regardless of their execution times.

In the results file, for each contingency executed, the triggered RAS objects are recorded (up to a maximum of 10 RAS per contingency).

### 27.11.7.2 Multiple Time Phase

Using Multiple Time Phase contingency Analysis in conjunction with RAS operation allows the user to evaluate in detail a sequence of events, for example if the entire remedial action scheme consists of a number of events which are expected to occur after various times after they have been triggered. The logic followed by the calculation can be illustrated by means of an example:

#### Use of two RAS to model post-fault generation changes

Consider a RAS called RAS1, which is intended to model the situation in which an overload on a line is alleviated by reductions in generation:

1. A fault occurs which overloads line L. This triggers reduction in generation from two generators, G1 and G2.
2. After 8 minutes, generator G1 reduces generation by 100 MW.
3. Five minutes later, 13 minutes after the fault, generator G2 reduces generation by 50 MW.

RAS1 will contain one trigger (1) and two remedial actions (2) and (3), which consist of Dispatch events occurring at 8 minutes and 13 minutes respectively.

The user also creates RAS2, to model the fact that if the output from G1 is reduced below a certain level, another generator G3 in a different part of the network should have its output increased to compensate.

1. The generation level at G1 goes below a prescribed threshold of, say 250 MW.
2. After 7 minutes, generator G3 increases generation by 100 MW.

RAS2 will contain one trigger (G1 generation drops below the specified level) and one remedial action, which consists of a Dispatch event on generator G3 with an Execution Time of 7 minutes.

Let us assume that the user sets up a Multiple Time Phase calculation, with the following time phases defined:

- 0 minutes
- 5 minutes
- 10 minutes
- 15 minutes
- 20 minutes

This will be the outcome:

**0 minutes** : Line L is overloaded. The trigger for RAS1 is activated, but nothing happens yet to reduce the load.

**5 minutes** : Line L is overloaded. The trigger for RAS1 is already activated, but still nothing happens yet to reduce the load because the first event only occurs 8 minutes after being triggered.

**10 minutes** : Now that more than 8 minutes have passed after RAS1 was triggered, the first generation change is taken into account and the active power on Line L is reduced. This drop in generation at G1 also triggers RAS2.

**15 minutes** : Now that more than 13 minutes have passed, the second generation change is also taken into account and the active power on the line is further reduced. However, it is not yet 7 minutes since RAS2 was triggered, so the associated event on G3 has not yet taken place.

**20 minutes** : Now that it is more than 7 minutes since RAS2 was triggered, the associated increase in generation at G3 will also be taken into account, resulting in the final state of the network for this sequence of events.

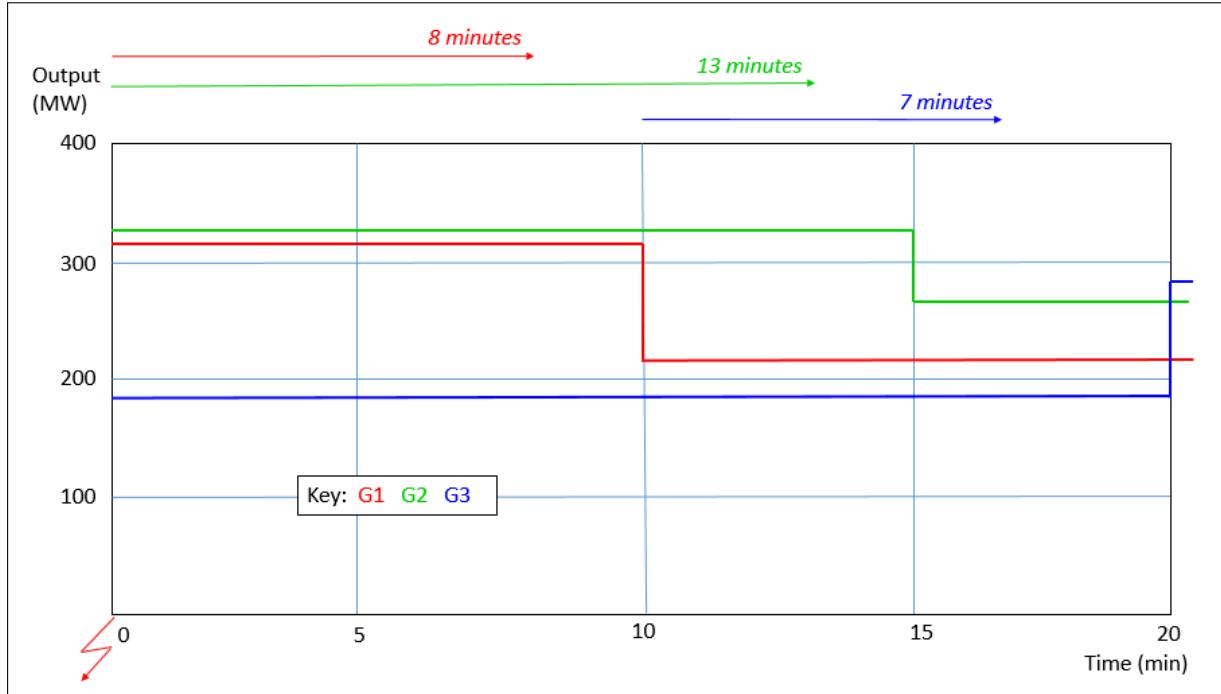


Figure 27.11.1: RAS timings in Multiple Time Phase Contingency Analysis

This example illustrates some important points relating to Multiple Time Phase calculations with RAS:

- In a multiple time phase contingency analysis, if a RAS is triggered in one time phase, the associated remedial action(s) will not have any effect until the next or later time phase, *even if the actions have zero Execution Time*.
- If remedial actions have a non-zero Execution Time, the timing starts when the RAS is triggered, *not* at time zero (unless of course, the first time phase studied is at 0s and the RAS is triggered at that point).
- RAS are only triggered at calculation points. Thus, in the example above RAS2 is triggered at the 10 minute point. If this were a real situation, the drop in generation at G1 would have happened after 8 minutes and therefore triggered the timer for the RAS2 at that time (resulting in G3 increasing output at 15 minutes), but we do not have a calculation point at 8 minutes, so the trigger is at 10 minutes and the increase is not observed until the 20 minute timephase.

### 27.11.8 Visualising RAS using the Trace Function

It may be helpful to visualise the effect of Remedial Action schemes on a particular contingency by using the Trace function (see [27.6](#)). When the use of RAS is enabled in the Contingency Analysis command, the Trace Function is available and the post-fault states can be seen first without and then with the triggered RAS(s) in a single time phase calculation, and similarly the complete sequence of events can be followed in a multiple time phase calculation.

## 27.12 Load Contingency Analysis Results

This option is accessed using the icon  on the Contingency Analysis toolbar. When a Contingency Analysis is executed, the results are stored in the associated results file. However, once the calculation has been reset, the results are no longer immediately available to be viewed on a diagram or in a Network Model Manager.

The Load Contingency Analysis Results option makes it possible for the results to be loaded back into memory, and summary quantities such as maximum loading viewed. To do this, use the icon to bring up the dialog box, select the required results file (if not already selected) and Execute.

Results which have been loaded back into memory are indicated with a grey background. This acts as a reminder to the user that these are previously-stored results and may no longer be consistent with the current network model if the model has been changed in the interim.

### 27.12.1 Load Individual Contingencies

The loading of results into memory also offers a further possibility: results from individual contingency calculations can be loaded into memory, which then allows the contingency to be examined in more detail on the graphic or in a Network Model Manager. To do this, use the icon to bring up the dialog box, select the required results file (if not already selected), select the required contingency and Execute.

The advantage of this process, compared with executing individual contingencies as single contingencies, is that various contingencies can be viewed without having to re-run the calculation,

# Chapter 28

## Quasi-Dynamic Simulation

### 28.1 Introduction

*PowerFactory* includes the *Quasi-Dynamic Simulation* toolbox, a dedicated time varying load flow calculation tool that can be used for medium to long term simulation studies. This tool completes a series of load flow simulations spaced in time, with the user given the flexibility to select the simulation period and the simulation step size. To achieve this, the *Quasi-Dynamic Simulation* makes use of time based parameter characteristics (refer to Chapter 18), variations and expansion stages (refer to Chapter 17), planned outages, simulation events and user defined time-dependent models.

The Quasi-Dynamic Simulation can be executed either sequentially (by default) or in parallel (via the parallel computing options).

The parallel computation (refer to Section 28.3.5) increases calculation performance by making full use of a machine's multi-core processor architecture.

For equipment and controls whose physical behaviour is known (e.g. via test measurements) the System Parameter Identification function can be used along with a non-parameterised simulation model in order to obtain an accurate representation of the system. The parameter identification dialog is opened by clicking . More details are available in Chapter 31.

This chapter is divided into several parts. Section 28.2 covers the technical background of the Quasi-Dynamic Simulation. Section 28.3 describes how to execute a Quasi-Dynamic simulation, Section 28.4 discusses how to analyse the output of the simulation, while Section 28.5 describes the modelling requirements and procedure for building user-defined *Quasi-Dynamic Simulation* models.

### 28.2 Technical background

The load flow calculation, detailed in Chapter 25 considers the network under a single set of operating conditions. In most electrical systems, engineers are interested in the performance of the system during worst case operational conditions. However, due to the complexity of the network, it might be difficult to intuitively understand which operating scenarios and network states cause such conditions. Consequently, to determine the worst case operating conditions, engineers often must run several different load-flow simulations with a range of operating conditions. This is usually achieved by modelling the network dependence on time because most operational parameters have an underlying dependence on time. For example:

- Load is dependent on time due to daily and seasonal cyclic load variation.
- Renewable sources such as solar and wind generation vary with solar insolation and wind speed

which are in turn functions of time.

- Network variations, maintenance outages, faults and unscheduled outages normally have some time dependence.
- Equipment ratings can also change due to the effects of wind and temperature.

Often when considering load flow variation over time, it is not the variations on a timescale of seconds (power system transients) that are of interest, but rather the behaviour of a network in timescales of minutes/hours up to months/years. It is, of course, possible to run a time domain simulation (RMS domain) with explicitly modelled dynamic controllers to simulate such a network (for more information refer to Chapter 29). Nevertheless, many of the time constants existing in stability models are much smaller than the simulation time steps being discussed here (e.g. the synchronous machine short-circuit time constants do not play any significant role in a medium- to long-term dynamic simulation, although they are represented in stability type simulations). These additional modelling considerations take a large computational effort and involve much unnecessary complexity if only the quasi-steady state load flow conditions are of interest. Consequently, a reasonable and pragmatic approach is to simulate so-called “Quasi-Dynamic” phenomena using a series of load-flow calculations with various model parameters being time dependent. Furthermore, to fit with real-world applications where control actions are executed based on the historical development (time dependence between consecutive time points), Quasi-Dynamic simulation uses the concept of state variables which are defined by their time derivative equations, as it is normally the case for time-domain simulations (refer to Section 28.5 for more information and examples).

Consider a simplified power system network consisting of four loads, two conventional synchronous machines and a solar photovoltaic power plant, linked by transmission lines. A single line diagram of the network is shown in Figure 28.2.1.

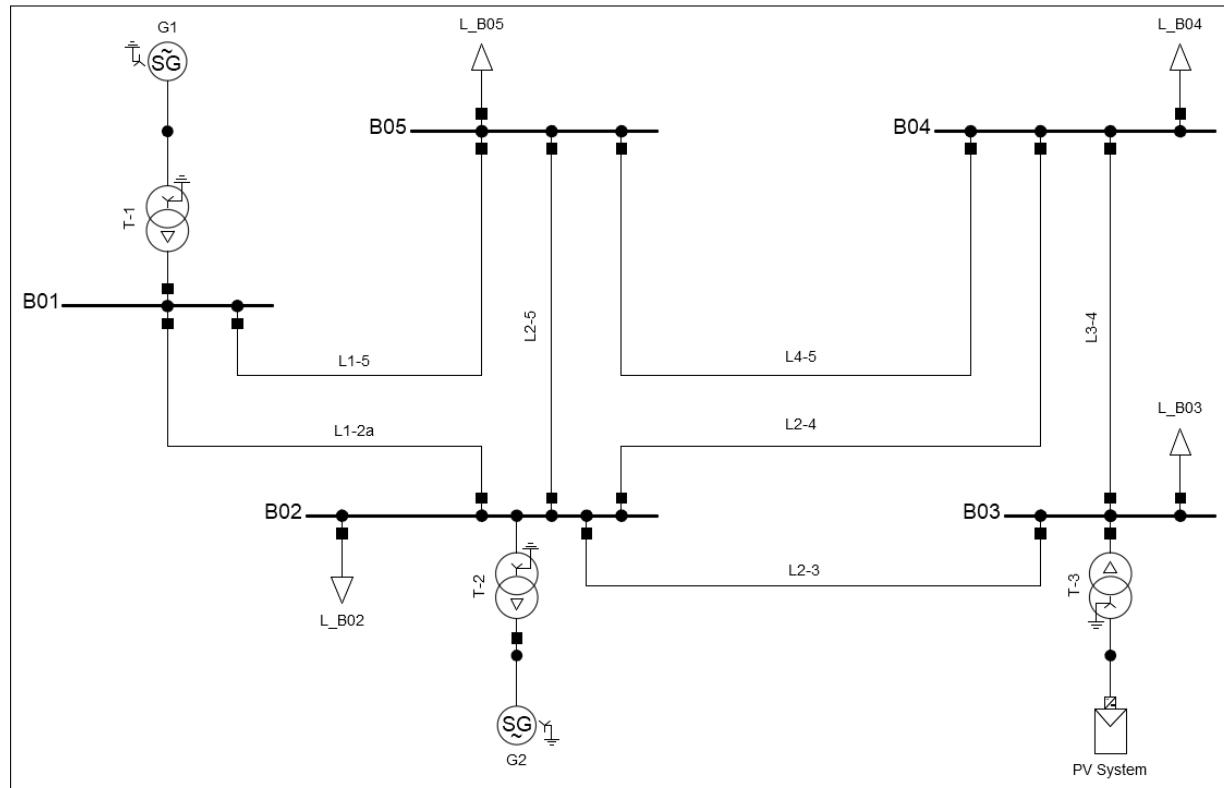


Figure 28.2.1: Single line diagram of example network

In this case, the load would vary depending on the time of day, the solar output would vary also depending on the insolation and consequently the conventional generators would be required to produce varying levels of output to balance the system load and generation. The engineer might be interested to

see the line thermal loading and voltage in the network over a period of say one week or perhaps even the seasonal variation over the course of one year. DPL or Python scripts could be written to achieve this, however by making use of the built-in parameter characteristics in *PowerFactory* and using the *Quasi-Dynamic* simulation tool, it is possible to complete such simulations very efficiently.

Figure 28.2.2 shows an example of the type of output that can be generated using this tool. The figure shows a clear cyclical pattern in the generation output, the line loading and the bus voltages. After determining the critical cases of such a simulation, the engineer might also like to complete more detailed RMS or EMT simulations on such cases to investigate potential short term issues. In this way, the Quasi-Dynamic simulation can be a useful screening tool.

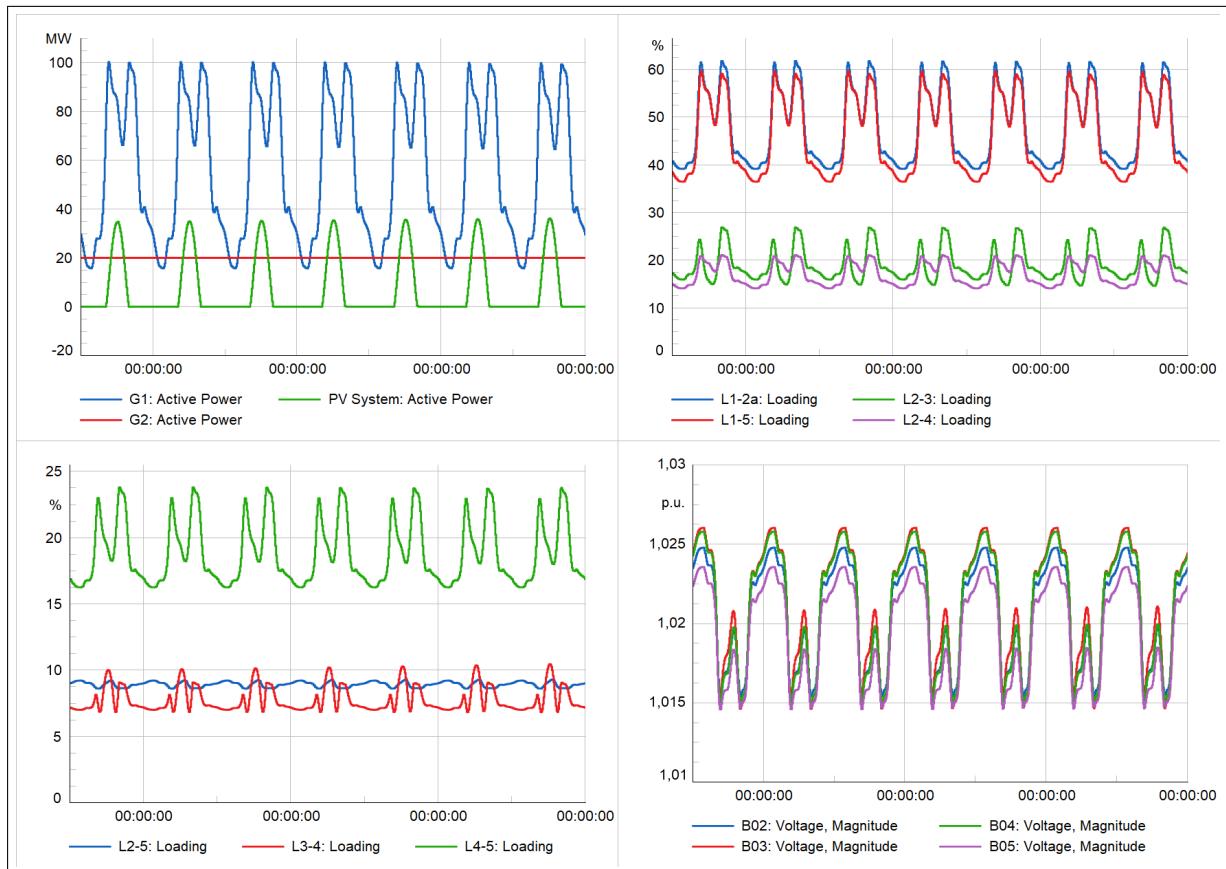


Figure 28.2.2: An example of weekly power flows in the example system calculated using the Quasi-Dynamic simulation tool

## 28.3 How to execute a Quasi-Dynamic Simulation

These are the steps taken when running a Quasi-Dynamic simulation:

- Defining the **Result Variables** to be monitored during the simulation. Refer to Section 28.3.1.
- Defining the **Maintenance Outages** and the **Simulation Events** to be applied during the simulation. Refer to Sections 28.3.2 and 28.3.3.
- **Running the simulation.** Refer to Section 28.3.4.
- **Plotting and analysing the results.** Refer to Section 28.4.

Optionally, the following preparatory steps may need be taken into consideration:

- The setup of time characteristics for quantities that are time varying in the network. Refer to Chapter 18 for more information on parameter characteristics.
- Configuring the network variations to model planned network changes. Refer to Chapter 17.
- Configuring the operation scenario to be applied during the simulation. Refer to Chapter 16.
- Configuring the model controllers (via user-defined models). Refer to Section 28.5.

The following sections explain these aspects in additional detail.

### 28.3.1 Defining the variables for monitoring in the Quasi-Dynamic simulation

Before running the Quasi-Dynamic simulation, it is necessary to tell *PowerFactory* which variables to record. To do this:

1. Click and select *Quasi-Dynamic Simulation*.
2. Click the button. A dialog will appear.
3. Select the type of results to be defined, either AC, AC unbalanced or DC. Note that the type of variables being monitored should match the type of load-flow calculation being used for the simulations.
4. Click **OK**. A tabular list showing the currently monitored variables will appear. By default *PowerFactory* will record some default variables for Areas, Feeders, Grids, Terminals, Zones and Branch elements.

**Note:** The default variable selection for Branch elements only contains transformer and line elements. Switches (ElmCoup and StaSwitch) are not part of this default selection and have to be added additionally.

---

5. Optional: Modify the default variables:
    - (a) Double-click the icon for the object. The variable selection dialog will appear.
    - (b) Select the variables that shall be stored. You can optionally filter them by *Group* or *Calculation*.
    - (c) Click **OK** to return to the tabular list of variables.
  6. Optional: Add recorded variables for another type of element/s:
    - (a) Click the icon. A blank dialog will appear.
    - (b) To add recorded variables for all objects of a specific class (for example all PV systems):
      - i. Enter the object class in the *Class Name* field. For example to record variables for all PV systems enter “ElmPvsys”.
      - ii. Press **tab** to update the dialog.
    - iii. Select the variables that shall be stored. You can optionally filter them by *Group* or *Calculation*.
    - iv. Click **OK** to return to the tabular list of monitored variables.
  - (c) To add recorded variables for a specific object:
    - i. Click the button next to *Object*.
    - ii. Choose *Select . . .*. A database navigation dialog will appear.
    - iii. Locate target object and select it.
    - iv. Press **OK** to open the variable selection dialog.
  - v. Select the variables that shall be stored. You can optionally filter them by *Group* or *Calculation*.
7. Click **Close** to close the list of monitored variables.

### 28.3.2 Considering maintenance outages

In the Quasi-Dynamic simulation it is possible to consider planned maintenance outages - this can also include planned generator derating. To do this the simulation uses the *PowerFactory* objects *IntPlannedout* and *IntOutage*. For more information about defining planned outages refer to Section 14.6.7.

To make the Quasi-Dynamic simulation consider the defined planned outages:

1. Select the *Maintenance & Events* page in the Quasi-Dynamic simulation command.
2. Check *Planned Outages* to automatically consider all outages during the simulation.
3. Optional: Click **Show Used Ones** to show a list of all currently considered outages. Only outages that occur during the defined simulation period on the basic data page are shown in this list.

**Note:** There must be at least one calculation time that lies within the execution time of the outage for it to be considered. For example, an outage at 23:30 is not executed in a *Complete Day* simulation with a *Step Size* of 1h, as the last calculation time would be 23:00.

4. Optional: Click **Show All** to open a data navigator on the planned outages folder. From here it is possible to see all outages that are defined within the project. Note, even those outages that would not occur during the simulation period are shown in this view.

### 28.3.3 Considering simulation events

#### 28.3.3.1 Overview

Various events are supported by Quasi-Dynamic Simulation either being pre-defined or generated via user defined *Quasi-Dynamic* models (QDSL models; see Section 28.5). This section provides an overview to the available simulation events and their usage, as they apply to Quasi-Dynamic Simulation. See Chapter 13, Section 13.9 for a detailed description of each type. Out of all possible events, the following are currently supported by Quasi-Dynamic Simulation:

- Dispatch event (EvtGen):  
From the firing time on, the active- and reactive power of the selected model (used in the calculation, i.e., incorporating characteristics and scaling, etc.) is incremented by the entered values. This means that, at any simulation point, the calculation value of the model is determined and if the simulation time has passed the execution time of a dispatch event, this calculation value is incremented again.
- External Measurement Event (EvtExtmea):  
Used to set and reset values and statuses as well as communication failure and restoration of external measurements.
- Load Event (EvtLod):  
A load step event increments the calculated load demand values at every point after the firing time. In particular, the absolute increment at a calculation time depends on the calculation value, since a proportional step change is entered in the event.
- Message Event (EvtMessage):  
If the execution time lies within the simulation period, the entered message is issued. Note that this event type is fired only once! Also note that the message is only issued when *Show detailed output* is selected in the QDS command.
- Parameter Event (EvtParam):  
This event enables changing signal parameters (*s:*) and calculation parameters (*c:*) during the simulation. It is also fired for every calculation point after its execution time.

- Stop Event (EvtStop):  
If this event is fired, the simulation will finish the current step and then stop. Results will be available until the stopping time and the calculation is valid.
- Switch Event (EvtSwitch):  
The switch event is executed once after the execution time. It is never fired again, if another event for the same switch exists that has a later execution time and the simulation time has passed this later event firing time.
- Tap Event (EvtTap):  
The calculated tap position of a transformer or shunt is determined for every calculation point. If a tap event has its execution time after the simulation time, a decrease/increase by one tap is executed or the tap is simply set to the selected value. Note that it is also fired for every calculation point after its execution time.
- Outage Event (EvtOutage):  
The outage event is executed once after the execution time. It is never fired again, if another event for the same element exists that has a later execution time and the simulation time has passed this later event firing time.
- Transfer Event (EvtTransfer):  
Portions of active and/or reactive power can be shifted from one load to another or even multiple other loads. It can also be used to shift power between static generators.

There are several ways to access *Event* objects:

- From the *Data Manager*, in the *Events of Quasi-Dynamic Simulation* object stored within the currently active *Study Case*;
- From the *Quasi-Dynamic Simulation* (ComStatsim) command, *Maintenance & Events* page, it is possible to *Select*; (▼), *Edit* (→) the currently selected *Events of Quasi-Dynamic Simulation*;
- From the Quasi-Dynamic Simulation toolbar by clicking the *Edit Events*  icon. A list of **all** pre-defined events will be displayed including the absolute execution time when the event will occur and the related object.

To create a new user defined event, do the following:

- Click the *Edit Events*  icon to open the *Events of Quasi-Dynamic Simulation*
- Use the *New Object*  icon in the toolbar to create a new event.
- The event type can be chosen from the list in the selection dialog which appears. Customisation of each event object is done by following the event description provided in Section 13.9.

#### Remarks:

- Events for QDS have a different GUI from that for dynamic simulation events. In particular, only events with a given (absolute) execution time within the calculation period will be fired.
- There is a difference between pre-defined events by the user and events that were fired by a QDSL model (28.5.5). If fired by a QDSL model, an event is also contained in the *Events of Quasi-Dynamic Simulation* folder until the calculation is reset. The firing is executed in the same fashion as for pre-defined QDS events.

#### 28.3.3.2 Use of events

- *Simulation events* flag - If the flag is set, *PowerFactory* considers and applies the relevant pre-defined simulation events during the simulation. The relevant events are those pre-defined events that occur in the specific time period (as defined in the *Time period* pane of the *Basic Data* page). If this flag is not set then neither pre-defined nor QDSL events are triggered during the simulation.

- **Show used ones** - Shows a list of all currently considered events. Only those events that occur in the defined simulation period are shown in this list.

**Note:** There must be at least one calculation time that is smaller or equal to the execution time of an event for it to be considered. For example, an event at 23:30 is not executed in a *Complete Day* simulation with a *Step Size* of 1h, as the last calculation time would be 23:00.

- **Show all** - Opens the *Data Manager* on the events folder. From here it is possible to see all events that are currently defined for this study case. Note, even those events that would not occur during the simulation period are shown in this view.
- **Remove all events** - Removes all events within the events folder.

#### 28.3.4 Running the Quasi-Dynamic simulation

To run a *Quasi-Dynamic Simulation* perform the following actions:

1. Click  and select *Quasi-Dynamic Simulation*.
2. Click  to open the *Quasi-Dynamic Simulation* dialog.
3. Edit the *Basic Options* → *Load Flow* pane:
  - Selection of *Load Flow* type to be used during each simulation time step:
    - *AC Load Flow, balanced*
    - *AC Load Flow, unbalanced, 3-phase (ABC)*
    - *DC Load Flow (linear)*.
  - *Settings*: Click  to edit and configure the *Load Flow Calculation* command used by the *Quasi-Dynamic Simulation*.
  - When the load flow results are approximated by a *Neural Network*, as described in Chapter 28.3.7, the selected neural network is linked here.
4. Edit the *Basic Options* → *Time Period* pane - the time period to run the simulation can be selected as follows:
  - *Complete Day* - a specific user defined day can be selected as simulation time range. The day is chosen in the corresponding field *Day*
  - *Complete Month* - a specific user defined month can be selected as simulation time range. The month is chosen in the corresponding field *Month*
  - *Complete Year* - a specific user defined year can be selected as simulation time range. The year is chosen in the corresponding field *Year*
  - *User defined calculation times* - specific user defined calculation times can be added for the simulation (using the time format *dd:mm:yyyy hh:mm:ss*). The simulation will only be executed at the defined calculation times. The User also has the option to ignore certain points in time by activating the associated option.
  - *User defined time range* - a customisable time period can be chosen for simulation by defining the *Begin* and *End* time points (using the time format *dd:mm:yyyy hh:mm:ss*).
5. Edit the *Basic Options* → *Step size* - Sets the time step size of the *Quasi-Dynamic Simulation*. A fixed step size simulation algorithm is used for time advance. The step size is defined by the *Step* (integer value) and the *Unit*. The *Unit* can be selected from the corresponding drop-down list (*Seconds, Minutes, Hours, Days, Months or Years*).
6. *Output* → *Results* - This field provides a link to the results object (*ElmRes*) that stores the calculation results. A different results object can be selected if required.
7. Edit the *Calculation Settings* page:
  - *Consider time-dependent states of models* - Set this flag to consider the time-dependency of *Quasi-Dynamic Simulation* models (*ElmQdsI*).

- *Max. number of control loops* - If the flag *Consider time-dependent states of models* is set, then the *Max. number of control loops* becomes relevant for the calculation. This number is used to limit the number of control iterations of a Quasi-Dynamic user-defined model. For more information refer to Section 28.5.4.

---

**Note:** The Quasi-Dynamic Simulation does not support the *Parallel computing method* if the *Consider time-dependent states of models* flag in the *Calculation settings* page is set.

- *Initialisation of load flow* - the initialisation of the load flow can either be *Based on previous load flow results* or a *Full initialisation* can be selected for each time step. If *Based on previous load flow results* is selected, the last valid load flow results will be used for the initialisation. If the calculation with these initial conditions fails, a *Full initialisation* is automatically executed as a fallback strategy.

8. Configure the *Output* report:

- *Off* - No output report is printed to the output window.
- *Short output* - A limited output report for each load flow calculation is shown.
- *Detailed output* - This option displays more detailed information and a *full load flow output per time step* in the output window.

9. Click **Execute** to execute the simulation.

10. If required, the simulation can be stopped before completion by clicking on the *Break* button (ⓧ) in the main toolbar.

---

**Note:** After starting the Quasi-Dynamic simulation, *PowerFactory* will determine the number of load-flows required based on the entered *Step size* settings and print this information to the output window. The total time it takes *PowerFactory* to complete the simulation varies depending on the time period being simulated and the step size. A progress bar will be displayed at the bottom of the *PowerFactory* graphical user interface.

---

### 28.3.5 Configuring the Quasi-Dynamic Simulation for parallel computation

*PowerFactory* supports the parallel execution of the Quasi-Dynamic Simulation. This is achieved via the *Parallel Computing* page of the *Quasi-Dynamic Simulation* command dialog. A number of options are provided in this page and described further below.

**Parallel computation:** This checkbox enables/disables the parallel computing feature. Tick this checkbox to enable parallel computing, un-tick it to disable the feature.

**Minimum number points in time:** The minimum number of time points needed for creating a parallel process. If the actual number of time points is less than this parameter, the calculation will be carried out sequentially (not in parallel).

**Package Size:** This parameter specifies how many time points will be distributed to a parallel process (this is called one package). When a process finished its assigned package, the main process will distribute the next package to this parallel process.

---

**Note:** Generally speaking, the package size is related to the calculation effort of one time point. If the network is rather large and one time point for the load flow calculation is computationally intensive, the package size should be small, otherwise in the end, it may happen that all other parallel processes have finished the calculation while one single process will still be executing for a long time to finish its whole package. On the other hand, if the calculation is fast for one time point, the package size should not be too small, otherwise lots of execution of time will be wasted in communication between the main process and parallel processes.

---

**Parallel Computing Manager:** The parallel computation settings are stored in a *Parallel Computing Manager* object (*SetParalman*). Further information on the particular configuration is found in Section [6.6.1](#).

---

**Note:** It is important to note that the Quasi-Dynamic Simulation does not support the *Parallel computing method* if the *Consider time-dependent states of models* flag in the *Calculation settings* page is set.

---

### 28.3.6 Configure the Quasi-Dynamic Simulation for real time simulation

QDS real time simulation can be executed if the option *Real-time simulation* is set to *Synchronised by system time*. This option can be found within the *Real time* settings of the QDS command. Furthermore the step duration can be adjusted, meaning that apart from the system time, one QDS time step will be executed every entered *time per step* or using a *Relative step duration* with a percentage *Ratio between real time and calculation time*. For proper real time simulations, it is recommended to set the *Step duration* to *relative*, entering the percentage value of 100%. The real time simulation capability can be applied if the OPC interface is used in QDS. Further information about the OPC Interface can be found in Chapter [24.14](#).

### 28.3.7 Use Neural Network approximation in the Quasi-Dynamic Simulation

In addition to the load flow calculation, *PowerFactory* also supports the use of neural networks for the *Quasi-Dynamic Simulation*. This is especially useful in larger power grids or when a lot of load flow calculations have to be executed, since neural networks can speed up the calculation immensely.

In comparison to the load flow calculation, a neural network does not have to solve equations based on physical laws (e.g. Kirchhoff's circuit law) to determine the steady-state load flow results, but it is able to predict the results purely by analysing available data. In order to approximate the results, it has to be trained in beforehand, so it can learn the behaviour of the network in different situations. To evaluate the training status of the neural network, the *Average validation error* can be assessed.

The neural network can be selected on the corresponding page in the QDS command. All information on how to set up and train a neural network can be found in Chapter [54](#).

If the option *Check consistency* is selected, a consistency check will be performed before starting the calculation to ensure that the current network model is consistent with the one used for the training.

---

**Note:** Time-dependent states like the State of Charge (SOC) of a battery can't be estimated by the neural network and are therefore not considered.

---

---

**Note:** All calculation settings and output variables are set by the neural network and can therefore not be changed in the QDS command anymore.

---

## 28.4 Analysing the Quasi-Dynamic simulation results

The Quasi-Dynamic simulation results can be presented in tabular form using the built-in reports, and in graphical form using the standard *PowerFactory* plot interface. Furthermore, *PowerFactory* also stores

summary statistics for every analysed variable. This section explains how to produce these three types of output.

### 28.4.1 Plotting

To produce an output plot of the Quasi-Dynamic simulation results follow these steps:

1. Click the  icon. A standard *PowerFactory* plot dialog will appear.
2. Select the desired variables in the curves section.
3. Optional: Adjust the plot options according to your preferences. Refer to Section 19.8 for more information on configuring standard plots in *PowerFactory*.
4. Click **OK** to create the plot in a new page.

### 28.4.2 Quasi-Dynamic simulation reports

The *Quasi-Dynamic Simulation Report* provides a means to summarise and examine system conditions over the simulated time period. There are three different reports available:

- Loading Ranges;
- Voltage Ranges; and
- Non-convergent cases.

The loading ranges report shows the maximum and minimum loading of each monitored branch element and also the time that each of these occurred.

The voltage ranges report shows the minimum and maximum observed voltages at each monitored terminal and the times that each of these occurred.

The non-convergent cases report shows a list of all the cases that did not converge and the time that they occurred.

To show the reports:

1. Click the  icon. A dialog for configuring the reports will appear.
2. Choose the reports to be produced.
3. Optional: Enable a loading range filter. You can use this to only show branch elements that exceeded a certain loading during the simulation. Note it is possible to alter the value of this filter, in the report later on.
4. Optional: Enable a voltage range filter. Here you can enter a lower limit and an upper limit. Only those terminals that had voltages recorded outside this range will show up in the report. It is also possible to alter these values in the report subsequently.
5. Click **OK** to show the reports. They will be displayed in separate windows.
6. Optional: Click  and choose *Export to Excel* or *Export to HTML* to export the results to Excel or in an HTML format.

### 28.4.3 Statistical summary of monitored variables

Conveniently, *PowerFactory* also calculates summary statistics for every monitored variable in the Quasi-Dynamic simulation. The following quantities are determined automatically:

- Average (mean)
- Maximum
- Minimum
- Time of maximum
- Time of minimum
- Range
- Standard Deviation. Note this is population standard deviation calculated according to:

$$\sigma = \sqrt{\frac{\sum (x - \bar{x})^2}{n}} \quad (28.1)$$

- Variance =  $\sigma^2$
- Only for monitored variables of active power: Energy over the entire period

To show these results:

1. Click the  icon and select the target object type of interest.
2. Click the *Flexible data* tab.
3. Click the  icon to define the shown variables.
4. Ensure the *AC Load Flow Sweep* page is selected.
5. Select the desired variables. For example “avg” is the mean value of the variable during the simulation and “std” is the population standard deviation.

#### 28.4.4 Loading Results

This option is accessed using the icon  on the Quasi Dynamic Simulation toolbar. When a Quasi Dynamic Simulation is run, the results are stored in the associated results file. However, once the calculation has been reset, the results are no longer immediately available to be viewed on a diagram or in a Network Model Manager.

This Load Time Sweep Results option makes it possible for the results to be loaded back into memory, and summary quantities such as maximum loading viewed, together with the individual load flow results for a selected time. To do this, use the icon to bring up the dialog box, select the required date and time and Execute.

Results which have been loaded back into memory are indicated with a grey background. This acts as a reminder to the user that these are previously-stored results and may no longer be consistent with the current network model if the model has been changed in the interim.

## 28.5 Developing QDSL models

*PowerFactory* allows the creation of user defined load flow and Quasi-Dynamic models (referred throughout the text as *QDSL models*) such to obtain customisable steady state behaviour of various power system equipment. This is done using the *PowerFactory* class objects *TypQdsi* and *ElmQdsi*.

Furthermore, to fit real-world applications where control actions are executed based on historical development, Quasi-Dynamic simulation extends the functionality of user defined models with the concept of

states. These state variables are defined by their time derivative equations, as it is normally the case for time-domain simulations. There are several situations where the time dependency between consecutive steady states must be considered. Examples include:

- Slow reacting equipment (e.g. slow varying generating units belonging to secondary control equipment reacting in the range of hours on large setpoint steps);
- Slow reacting controllers - Control systems with time constants comparable to or greater than the simulation time step (e.g. Transformer/shunt tap changers, etc.);
- Certain quantities might be time dependent but cannot be provided as a time characteristic, i.e. the values over the whole simulated time range are not known apriori (prior to execution of the simulation) but only a starting value is available. Consider for example the state of charge of a battery system, fuel remaining/consumed in a diesel generator set, etc.

### 28.5.1 PowerFactory objects for implementing user defined models (*TypQdsI* and *ElmQdsI*)

Two objects are available in *PowerFactory* for creating user defined models:

- **Quasi-Dynamic model type** (*TypQdsI*, library type object). Represents the type definition of a Quasi-Dynamic model. It is referred throughout this Section as *QDSL type*;
- **Quasi-Dynamic model** (*ElmQdsI*, network element object). Represents the network model instance of a certain user defined model (of type *TypQdsI*). It is referred throughout this Section as *QDSL element*.

#### 28.5.1.1 Creating the Quasi-Dynamic model type *TypQdsI*

A new *QDSL type* object *TypQdsI* may be created in the project library (default location is *Library* → *Dynamic Models*). The *QDSL type* must be edited and all the necessary scripting code must be introduced.

To create a *QDSL type* (*TypQdsI*) do the following:

- Using the Data Manager, navigate to *Library* → *Dynamic Models* folder in the active project. Use the *New Object*  toolbar icon and select the object *Quasi-Dynamic Model Type*;
- Alternatively, using the Data Manager, right-click inside the *Library* → *Dynamic Models* folder in the active project and select *New... → Quasi-Dynamic Model Type* from the context menu;
- Alternatively, if a *QDSL element* (*ElmQdsI*) is already available, one can open the element's dialog and create a new *QDSL type* by clicking on *Select (▼) → New Project Type*. This action will create a new *QDSL type* within the project's *Library* → *Dynamic Models* folder.

The *QDSL type* (*TypQdsI*) dialog is divided in several sections with the following properties:

**Basic Data** page options:

- *Use of scripts in calculation method* - By default, the same scripts are used for balanced AC, unbalanced AC and DC load flow calculations. If separate scripts are to be used depending on the calculation method, the options *Separate scripts for unbalanced AC Load Flow* and *Separate scripts for DC Load Flow* can be selected.
- *Variables* table - Two variable types are available: state variables and parameters. Parameters should be constants that define the particular specification of a model. States determine the time-dependent behaviour. They are integrated with respect to their differential equations during Quasi-Dynamic simulations. All variables defined within this table are accessible from all model scripts.

- *Connected network elements* - Specific *PowerFactory* objects can be used as referenced objects. This table defines the names to identify them in the model scripts of a *QDSL type* (*TypQds*). Member variables can be accessed within all model scripts as it is typically done within a DPL script. The input/output flag defines the type of reference object. Set it to input if the referenced object is used as measurement point (reading variables). Set it to output if the referenced object is being controlled. This flag is particularly useful for dealing with the initialisation order of inter-referenced user defined models (e.g. one *ElmQds* model *QDSL1* may be controlled by a second one, named *QDSL2*; in this case *QDSL2* will be provided as a *Connected network elements* item for *QDSL1* and be set to “Input” type). Alternatively, *QDSL1* could be set as an output of *QDSL2*.

#### **Results** page options:

- *Results table* - The list of result variables can be defined. All result variables are of type double. Like state variables or parameters, result variables can be recorded in the results file of a *Quasi-Dynamic Simulation*. In particular, their time-dependent character can be captured.

**General Scripts** page: This page contains all scripts that are used for the balanced AC load flow. If no separate scripts are selected for unbalanced AC and/or DC load flow on the *Basic Data* page, these calculation methods will also use these scripts. If separate scripts have been selected, the pages *Unbalanced AC Scripts* and/or *DC scripts* also appear. The tabs in these pages are identical to the *General Scripts* page. The different script tabs are explained below.

#### **Initialisation** tab:

- *Initialisation* script - The initialisation script defines part of the model behaviour. The script is used to initialise the model state variables and other parameters (for details refer to Section 28.5.4).

#### **LDF: Equations** tab:

- All options within this page are used for all load flow based calculations (and are not specific to Quasi-Dynamic Simulation).
- *Linear Model* flag - Check this flag if the load flow *Equations* script is representing a linear model in its inputs and outputs (for further details refer to Section 28.5.4).
- *Restrict shown variables to those supported in all selected methods* - if selected, only the input/output variables supported by all the selected calculation methods will be shown. This simplifies the creation of scripts that are valid for several calculation methods(i.e. AC balanced and unbalanced and DC load flow) .
- *Inputs/Outputs for load flow equations* - As described in Section 28.5.4, the *Equations* script of the load flow part does not allow direct access to any monitoring (**m:**) and calculation variable (**c:**). A limited set of variables which can be accessed within the *Equations* script is provided for selection within this table. Each class of network elements (*PowerFactory* class objects starting with “Elm”) includes an extensive number of variables from which the user may choose the set of interest. To insert one variable do the following:
  - Append an empty row in the table (if not there already)
  - In the first column, type in the variable name/identifier to be used in the *Equations* script. This name is arbitrary (but must be unique in the model) and is chosen by the user.
  - Select the usage (second column) *Input* if the variable is a measurement (e.g. a remote voltage measurement on a busbar) or *Output* if it is a controlled variable (e.g. the reactive power setpoint of a static generator).
  - In the *Class name* column, double-click in the field to open a selection with available class names. If different element classes are to be supported (e.g. *ElmGenstat* and *ElmPvsys*), several class names can be selected.
  - Double-click the empty field in the *Variable name* column and select from the list of available variables the one which is of interest. If several classes are selected, only the variables available for all the classes are displayed.
  - If an input variable is selected, then the *Bus/Phase name* must also be selected. Double-click the corresponding empty field and select the appropriate bus/phase.

- *Load flow equations* script - The *Equations* script is one of the model scripts that define the model behaviour. The script must contain the model's inner loop load flow equations and during execution of any given arbitrary inner loop it must be differentiable and the set of equations should be fixed. (for further details refer to Section [28.5.4](#)).

**LDF: Control tab:**

- *Control* script - The *Control* script is one of the model scripts that define the model behaviour. The script must contain the model's outer loop load flow control equations (for further details refer to Section [28.5.4](#)).

**QDS: Equations tab:**

- *Quasi-dynamic equations* script - This script is one of the model scripts that define the model behaviour. The script is executed once per time step and must contain the model's time derivative equations of all time-dependent quantities (state integration, etc.). For further details refer to Section [28.5.4](#).

**QDS: Control tab:**

- *Control* script - The *Control actions for Quasi-Dynamic Simulation* script is one of the model scripts that define the model behaviour. It can be executed multiple times in order to achieve a steady state control output at a certain time step. The script must contain the model's outer loop Quasi-Dynamic control actions, e.g. triggering of events or limitation of states. (for further details refer to Section [28.5.4](#)).

**Note:** The *Quasi-Dynamic Simulation* command (*ComStatsim*) does not consider by default the time dependency of *QDSL models* (i.e. by default, the model scripts within the *Quasi-Dynamic Simulation* page of the *QDSL type TypQdsI* are not executed). The initialisation and load flow scripts of the *QDSL model* will still be included in the calculation. This simplification brings the advantage that, by default, the *Quasi-Dynamic* simulation contains a set of decoupled load flows which can be independently executed (e.g. within a parallel computation algorithm). To consider the time dependency of models, make sure that the flag *Consider time-dependent states of models* in the *Calculation Settings* page of the *Quasi-Dynamic Simulation* command dialog is set.

---

**Version page:**

A number of version control fields are available, as follows:

- Company
- Author
- Version
- Last Modified
- Change Log

## 28.5.2 QDSL Model Encryption

*PowerFactory* offers the possibility to encrypt the script code of a QDSL type (Quasi-Dynamic Model Type). The encryption action can be initiated by pressing the corresponding button in the edit dialog of the QDSL Type object. The encryption process will ask in a dialog for a password. The password will be required for later decryption and is not needed for running simulations using the QDSL type. After successful encryption, the code is hidden and only the name of the QDSL type itself can be changed.

The encryption is reversible; an encrypted script can be decrypted using the corresponding button in the edit dialog of the encrypted TypQdsI-object. After entering the password and confirming with **OK**, the script returns to its original status, where all properties may be changed and the script code is shown.

---

**Note:** The user should be aware that encryption can never guarantee complete security. The chosen technology balances the requirements for security with the usability and performance of encrypted models. Generally, users are advised to share models only with trusted partners.

---

### 28.5.2.1 Creating the Quasi-Dynamic Model *ElmQdsI* (*QDSL element*)

Based on an already created *QDSL type* object *TypQdsI* (located in the project/global library folder) any number of new *QDSL elements* can be created in the network folder, representing instances in the power system of the type definition object. Similarly with dynamic models (DSL based), parameters in the *QDSL type* object (*TypQdsI*) are only used as literals while in the *QDSL element* *ElmQdsI* actual values of these parameters are assigned.

To create a new *QDSL element*, do the following:

- Using the Data Manager, navigate in the active project to *Network Model* → *Network Data* → [*Target Grid*] folder (where [*Target Grid*] represents the network where the model is intended to be deployed). Use the *New Object*  toolbar icon and select the object *Quasi-Dynamic Model*;
- Alternatively, from the single line diagram, right-click on any network element that shall be controlled and select *Network Models* → *Quasi-Dynamic Model* → *New...*. Then a *QDSL type* can be selected or defined and the model is automatically created in the respective grid folder.

The *QDSL element* (*ElmQdsI*) dialog is divided in several sections with the following properties:

#### Basic Data page options:

- *Supported calculation methods* - Selection of the types of load flow calculations for which the model is considered:
  - AC Load Flow, balanced, positive sequence
  - AC Load Flow, balanced, 3-phase (ABC)
  - DC Load Flow (linear)
- *Type* - The intended *QDSL type* can be assigned here via the selection icons.
- *Out of service* flag - Set this flag in order to disable the model. To enable the model this flag must be unset.
- *Parameters table* - Out of the *Variables* table declared in the model definition *Variables* table, all the parameters are listed here. Values for all parameters defined in the *Variables* table of the *QDSL type* (*TypQdsI*) are assigned here.
- *Connected network elements* - Within this table, specific *PowerFactory* objects can be used as referenced objects. Addressing a specific referenced object is done in the scripts via the corresponding object name (declared in the *QDSL type*).

#### Load Flow page options:

- A separate tab is displayed for each calculation method (Balanced AC, Unbalanced AC, DC), on which network elements defined in the *Inputs/Outputs for load flow equations* table on the *LDF: Equations* tab of the *QDSL type* object scripts can be selected.
- Only network elements of the declared class (as defined in the *QDSL type*) can be selected for a specific row of the table *Network elements of inputs/outputs*.
- If the same script is used for several calculation methods, the same connected network elements are also used.

**Description** page - A standard *Description* page is made available for version control purposes.

### 28.5.3 Overview of modelling approach

The two *PowerFactory* objects described above, *QDSL type* (*TypQdsI*) and element (*ElmQdsI*) provide the framework of integrating a user defined *Quasi-Dynamic* model *QDSL model*.

The scripting language for all the model definition (*TypQdsI*) scripts is *DlgsILENT Programming Language* (DPL). Refer to Chapter 23 for a comprehensive description of DPL. Further documentation is available within the *DPL Reference* which provides a full description of available functions. The *DPL Reference* can be accessed in the *PowerFactory* program from the main menu at *Help → Scripting References → DPL*. Functions that can be used within the *QDSL model* scripts are marked appropriately in the *DPL Reference* documentation by a star (\*) suffix (e.g. *GetFromStudyCase\**). There are functions (without a star (\*) marking) that are not available within the model scripts, e.g. the execution of a short circuit *ComShc* command is not allowed. Besides the available DPL functions, there exist several functions which are specific to *QDSL model* scripts **only**. They are listed and described in Section 28.5.5.

*PowerFactory* provides seamless integration of *QDSL models* within the Load Flow calculation and the *Quasi-Dynamic* Simulation engines. This means that once a *QDSL model* is implemented it will be active for both Load Flow (*ComLdf*) and *Quasi-Dynamic* Simulation (*ComStatSim*) commands and any other command which subsequently makes use of any of the two. A high level overview of the integration of the user defined models within a *PowerFactory* project is shown in Figure 28.5.1. The user is provided a high degree of flexibility in terms of interaction with the power system model via the comprehensive scripting functionality. Any number of *QDSL models* can be created. Each model may control one or several network elements. Moreover, each model may measure power system quantities of one or several network elements. This allows development of complex control schemes applicable for all load flow based calculations.

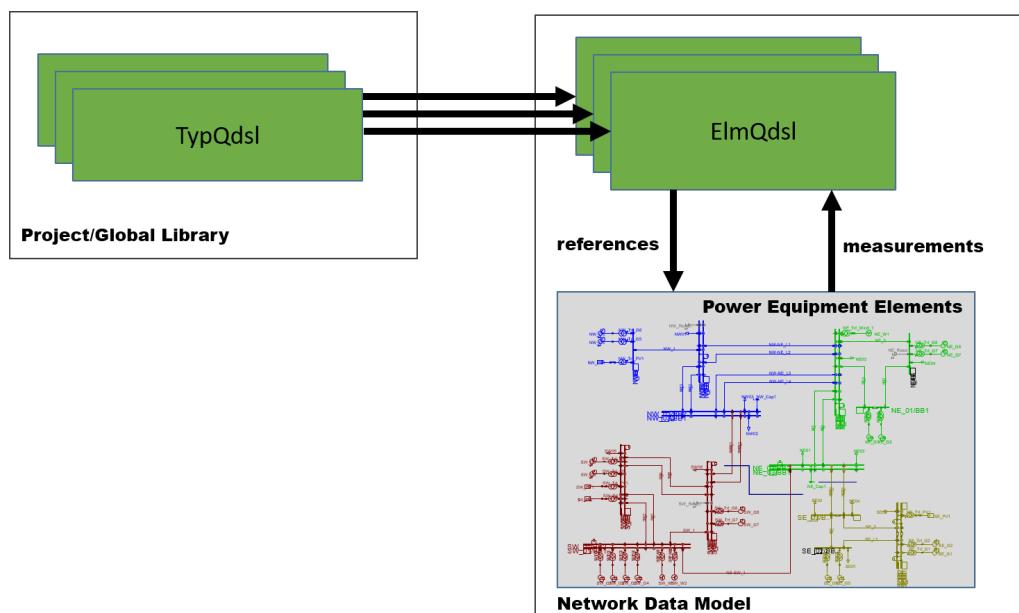


Figure 28.5.1: Integration of *QDSL models* within the *PowerFactory* project

### 28.5.4 Algorithm flow of user defined Quasi-Dynamic models

The simulation procedure of a *QDSL model* that includes time-dependent state variables is shown in Figure 28.5.2. Here, the time advance from a generic discrete absolute time point ( $t_k$ ) to the next time point ( $t_{k+1}$ ) is exemplified, with  $t_{k+1}$  depending on  $t_k$  and the parameter *Step* provided in the *Basic Options* page:

$$t_{k+1} = t_k + Step$$

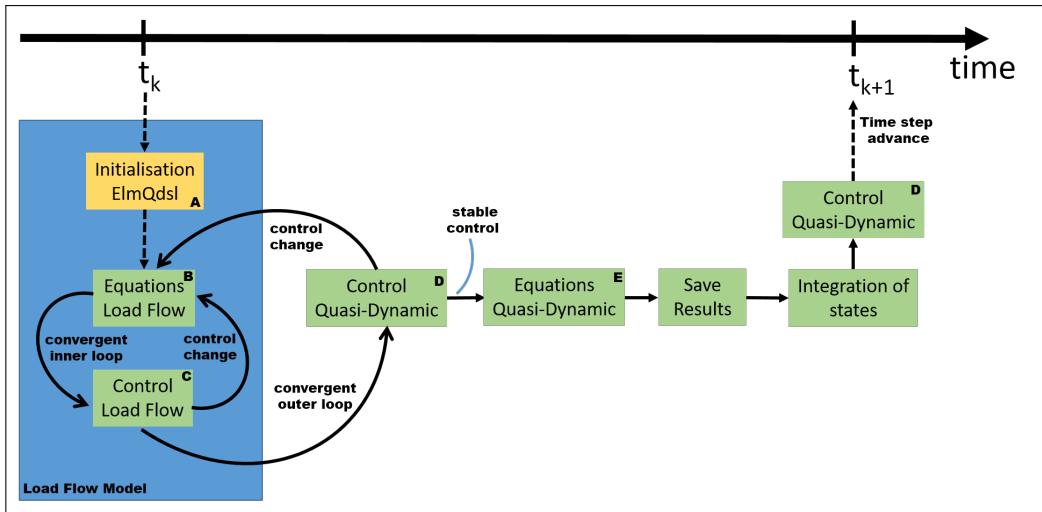


Figure 28.5.2: QDSL models - Simulation Procedure

With reference to Figure 28.5.2, within a *QDSL model* the following blocks of user defined code may be programmed in the *QDSL type*:

1. Initialisation block (**A**)
2. Load Flow (LDF) Equations block (**B**)
3. Load Flow (LDF) Control block (**C**)
4. Quasi-Dynamic Simulation (QDS) Control block (**D**)
5. Quasi-Dynamic Simulation (QDS) Equations block (**E**)

In the following, a short description of the intended functionality of each block is given:

**A. - Initialisation script:**

- This script is used for the purpose of initialising the model state variables, the result variables and parameters. Initialisation can be done with a constant value or via a calculation based on certain network element parameters. Therefore, the user can even overwrite parameterisation of parameters set in the *QDSL element* tables);
- This script is relevant for *Load Flow* calculation and *Quasi-Dynamic* simulation;
- When executing a *Quasi-Dynamic* Simulation (*ComStatsim*), this block is executed once for each *QDSL element* only the first time the respective model is inserted into the calculation. Insertion may happen at the beginning of the simulation or, if added within an expansion stage, at the activation time of the expansion stage containing it. If a *QDSL element* is outaged at a certain moment in time then, upon re-insertion at a subsequent instance, it will be re-initialised. Any subsequent time steps occurring after the model insertion will have the effect of bypassing the execution of the Initialisation block (i.e. will not be executed);
- When executing a simple *Load Flow* calculation (*ComLdf*) this block is executed once before running the load flow iterations;
- Within this script the user has access to a limited list of environment variables, i.e. element parameters (e.g. *battery:e:sgn*) of elements being provided as *Connected network elements* in the *Basic Data* page of the *QDSL type TypQdsI*. Neither monitoring variables (e.g. *m:P:bus1* of *ElmGenstat*) nor calculation variables (e.g. *c:loading* of *ElmGenstat*) are available within this script;
- Signals of controlled models (e.g. *s:id\_ref* of *ElmGenstat*) can also be initialised here.

**B. - Load Flow (LDF) Equations script:**

- This script is used for solving the model load flow equations of a fixed state (e.g. determination of the battery power injection such that the load flow reaches steady state; neither the charging mode nor the state of charge should change within this script);
- This script is relevant for *Load Flow* calculation and *Quasi-Dynamic* simulation;
- The equations must be differentiable in its inputs/outputs and fixed until the outer loop control script is called;
- Monitoring variables are available here only via the *Inputs/Outputs for load flow equations* table available in the *Load Flow* page of the *QDSL type TypQdsl*. A comprehensive number of variables are made available for each build-in model type. Refer to the technical reference of each power system model for a specific listing of available variables;
- Input and output variables shouldn't be used within if-conditions in the Load Flow Equations (e.g. *if(u > 1){...}*), as the variables can vary significantly during the inner loop until convergence is reached. It is recommended to use modes (e.g. *if(mode\_overvoltage = 1){...}*) instead and determine the mode of operation in the *Load Flow Control* (C).
- The DPL function **SetEquation** is used to formulate the load flow equation of a certain load flow model output. For each load flow model output (as explicitly defined in the *Inputs/Outputs for load flow equations* table available in the *Load Flow* page of the *QDSL type TypQdsl*) there must be exactly one *SetEquation* command that will characterise the output. Refer to the specific function **description** for further information (e.g. Should a model contain five output signals, then five *SetEquation* equations must be formulated - one for each output);
- *Linear Model* flag: the load flow *Equations* script is assumed to be linear in **all** inputs and outputs. For such models, if set, there is a certain performance improvement that can be obtained. Convergence behaviour must be checked on individual basis. Do not set this flag if the model is not linear.

**Note:** For example, assuming that the load flow equation function of a certain model is defined by:

$$f(P_{\text{set}}) = P_{\text{set}} - 2 = 0$$
$$\frac{df}{dP_{\text{set}}} = 1 = \text{constant} \implies \text{model is linear, in the only output } P_{\text{set}}$$

Assuming the load flow equation function of a certain model is defined by equation below (where  $P_{\text{ini}}$  is a parameter):

$$f(P_{\text{set}}, u_i, u_r) = P_{\text{set}} - (u_r^2 + u_i^2) \cdot P_{\text{ini}} = 0$$

Calculating the first order derivative of the function with respect to its inputs:

$$\frac{df}{dP_{\text{set}}} = 1 = \text{constant} \implies \text{model is linear in } P_{\text{set}}$$
$$\frac{df}{du_i} = -2 \cdot P_{\text{ini}} \cdot u_i = \text{not constant} \implies \text{model is non-linear in } u_i$$
$$\frac{df}{du_r} = -2 \cdot P_{\text{ini}} \cdot u_r = \text{not constant} \implies \text{model is non-linear in } u_r$$

---

#### C. - Load Flow (LDF) **Control** script:

- This script checks for convergence of the load flow outer-loop control algorithm. The machine state of the model may be changed here in order to reach a satisfactory operating point (e.g. determination whether a battery is in charging or discharging operation modes depending on measured network conditions, i.e. measured busbar voltage, measured power transfer via a branch, etc.);
- This script is relevant for *Load Flow* calculation and *Quasi-Dynamic* simulation;
- Within this script, all monitoring variables and calculation quantities of referenced models or DPL-retrieved models are available;

- Convergence criterion: by default, the algorithm decides internally whether the model triggers an additional control loop by observing changes in result variables or states of the *QDSL model* or the signals of the connected network elements controlled by the model. The threshold triggering outer loops is defined in the load flow command. Alternatively, the user can take control over the convergence decision by calling the DPL function [SetControlLoopFinished](#).

D. - Quasi-Dynamic Simulation (QDS) **Control** script:

- This script has a similar purpose as its Load Flow counterpart, i.e. to check for convergence in the Quasi-Dynamic Simulation solution at time step  $t_k$ ;
- This script is relevant for *Quasi-Dynamic* simulation only;
- With reference to the algorithm flow (see Figure 28.5.2), the script may be executed multiple times until a steady state condition is reached within the Quasi-Dynamic loop (defined by transitions “control change”, “convergent inner loop” and “convergent outer loop”). The script is run once more at the end of the time step iteration, after the results have been saved for time  $t_k$  and the integration has been performed for time  $t_k + 1$  for the purpose of limiting the newly integrated state variables, thereby ensuring feasible conditions for the next time step;
- Simulation events can only be generated within this script. Typically, after a simulation event has been applied to the network one further iteration of the Load Flow loop should be executed. Therefore, events are triggered immediately. It is not possible to create events in the *QDS control block* which is executed between time steps  $t_k$  and  $t_{k+1}$ . To distinguish between the different *QDS control blocks*, use the function [IsDuringFinalControlQdsScript](#);
- Within one time step, a single event of a given type can be applied on a single target network element by one *QDSL element*. Multiple events can be generated on different elements;
- Within this script, all monitoring variables and calculation quantities of referenced models or DPL-retrieved models are available;
- Convergence criterion: by default, the algorithm decides internally whether the model triggers an additional control loop by observing changes in result variables, state variables, triggering of events etc. Alternatively, the user can take control over the convergence decision by calling the function [SetControlLoopFinished](#). Non-convergence is triggered when the maximum number of outer loops is exceeded. This threshold is a *Quasi-Dynamic* Simulation command parameter, available in the *Calculation Settings* page. (parameter “Max. number of control loops”).

E. - Quasi-Dynamic Simulation (QDS) **Equations** script:

- This script is executed once per simulation time step and contains statements of all state variables differential equations. It is the only location where state derivative equations may be defined;
- This script is relevant for *Quasi-Dynamic* simulation only;
- Formulation of state derivatives is similar to the Dynamic Simulation Language (DSL) of the Stability/EMT user defined models, i.e. each dot suffixed state variable represents the time derivative of the corresponding state variable and must be assigned a single user defined equation. For example, assuming that *SOC* is defined as a state variable and represents the percent state of charge of a battery, *Eini* is the battery’s capacity in MWh, then:

```
SOC. = -Pset * 100. / (Eini * 3600.); ! slope of SOC
```

- It is possible to use derivatives of state variables in all QDSL scripts. Derivatives can be accessed via a *statename : dt* statement (e.g. given a generic state variable *x*, the derivative value is called using *x : dt* or alternatively *this : s : x : dt*). The format *statename.* is only allowed when defining the state derivative equation and never when retrieving the actual value for further calculation;
- Within this script, all monitoring variables and calculation quantities of referenced models or DPL-retrieved models are available;

### 28.5.5 Scripting Functions for Quasi-Dynamic Simulation

function	description
CreateEvent	Creates a simulation event with a certain execution time
CreateMultiLoadEvent	Creates a simulation load event defined for multiple loads
GetSimulationTime	Retrieves the current simulation time
GetStepSizeSeconds	Returns simulation step size in seconds
SetEventParameter	Function used to set a certain event parameter
SetEquation	Defines the load flow equation of one load flow output variable
GetEquationMismatch	Retrieves the current value of the SetEquation of a certain index
SetControlLoopFinished	Enables the user to govern the outer loop behaviour of the <i>Control</i> scripts
IsOutaged()	Checks if element is outaged
GetControlLdfIteration()	Retrieves the outer loop iteration number of the current load flow calculation
IsDuringFinalControlQdsScript()	Enables to distinguish between the QDS Control between time steps and the <i>QDS Control block</i> in the outer loop within a time step, see Figure 28.5.2

Table 28.5.1: Overview of DPL functions for Quasi-Dynamic simulation

#### CreateEvent

Creates an event of a given type for the Quasi-Dynamic simulation. This event exists only temporarily in the study case folder of the Quasi-Dynamic simulation and will be deleted when the calculation is reset. Note that events can only be created in the Control script (for the Quasi-Dynamic simulation) of *QDSL models*. For further details about QDS-Events, please refer to Section 28.3.3. **object CreateEvent(string eventType, [double executionTime])**

#### Arguments:

*eventType (obligatory)* : Type of event to be created, e.g., EvtGen, EvtParam, EvtSwitch, etc.  
*executionTime (optional)* : Execution time for the event (in seconds since 00:00 01.01.1970GMT). If not set, the current simulation time is used.

#### Return value:

If successful, the event that was created is returned.

#### Example:

The following example shows how a Dispatch event for a generator is created and executed at the next time step:

```
double time;
object event, gen1;
set setGens;

time = GetSimulationTime();
event = CreateEvent('EvtGen', time+1);

setGens = GetCalcRelevantObjects('ElmSym');
gen1 = setGens.First();

if ({event}<>NULL).and.{gen1}<>NULL) {
    SetEventParameter(event, 'loc_name', 'GenEvtQDSL');
```

```

    SetEventParameter(event, 'p_target', gen1);
    SetEventParameter(event, 'dP', 1.0);
    SetEventParameter(event, 'dQ', 2.0);
}

```

### CreateMultiLoadEvent

Creates a load event for multiple loads during Quasi-Dynamic simulation. This event exists only temporarily in the study case folder of the Quasi-Dynamic simulation and will be deleted when the calculation is reset. Note that events can only be created in the Control script (for the Quasi-Dynamic simulation) of *QDSL models*.

**object CreateMultiLoadEvent(set setOfLoads, [double executionTime])**

**Arguments:**

*setOfLoads* : Set of all loads to be considered by the event.

*executionTime (optional)*: Execution time for the event (in seconds since 00:00 01.01.1970GMT). If not set, the current simulation time is used.

**Return value:**

If successful, the event that was created is returned.

**Example:**

The following example shows how a multi-load event is created and scheduled for the current time step:

```

object load,
event;
set setLoads,
allLoads;

allLoads = GetCalcRelevantObjects('ElmLod');
load = allLoads.First();
setLoads.Add(load);
load = allLoads.Next();
setLoads.Add(load);
event = CreateMultiLoadEvent (setLoads);

if (event<>NULL) {
    SetEventParameter(event, 'loc_name', 'MultiLoadEvent1');
    SetEventParameter(event, 'iopt_type', 0);
    SetEventParameter(event, 'dP', 1.0);
    SetEventParameter(event, 'dQ', 2.0);
}

```

### GetSimulationTime

Get the current simulation time during Quasi-Dynamic simulation (in seconds since 00:00 01.01.1970GMT).

**double GetSimulationTime ()**

**Arguments:** -

**Return value:**

Returns the current simulation time (in seconds since 00:00 01.01.1970GMT).

### **GetStepSizeSeconds**

Get the Quasi-Dynamic simulation step size in seconds. If user-defined calculation times are used, the step size to the next point in time is returned.

**double GetStepSizeSeconds ()**

**Arguments:** -

**Return value:**

Returns the step size in seconds.

### **SetEventParameter**

Setup the parameters of a (temporary) event created during Quasi-Dynamic simulation by a QDSL model.

**bool SetEventParameter(Object event, string paramName, int|double|string|object value)**

**Arguments:**

*event*: Event whose parameter shall be modified.

*paramName*: Parameter name to be modified for event object.

*value*: New value to set for parameter *paramName* of event object.

**Return value:**

Returns non-zero if setting the parameter failed.

**Example:**

Refer to example of [CreateMultiLoadEvent](#).

### **SetEquation**

Equations are formulated and solved in the form  $f(x)=0$ . The number of equations for a QDSL model is equal to the number of outputs defined. Here the currently calculated value for the equation of a certain index is set. This function must be called for every index  $0, \dots, (\text{number of outputs}-1)$ . Note that this function can only be called in the load flow *Equations* script of QDSL models.

**bool SetEquation (int eqIdx, double eqValue)**

**Arguments:**

*eqIdx*:The index of the equation to set the value (zero-based).

*eqValue*:The currently calculated value to set for the equation.

**Return value:**

Returns non-zero if setting the equation value failed.

**Example:**

The following example shows how an equation  $f(\text{in}, \text{out}) := \text{out} - \text{in} = 0$  can be formulated and solved for one input in and one output out.

```
SetEquation(0, in-out);
```

The variable *in* represents in the example above an *Output* signal as declared within the *LDF: Equations* tab of the *QDSL type*, table *Inputs/Outputs for load flow equations*. The variable *out* represents the setpoint value to be obtained within the load flow inner loop by the variable *in*. The variable *out* is typically calculated within the *Equations* script. Example, where *u* and *in* are defined in Table 28.5.2 and  $P_{\text{rated}}$  is a parameter:

```
double out;
out = Prated * u ;
SetEquation(0, in-out);
```

Name	Usage	Class name	Variable name	Bus/phase name
in	Output	ElmGenstat	pset	
u	Input	ElmGenstat	u1	bus1

Table 28.5.2: Inputs/Outputs for load flow equations of *QDSL type*

### GetEquationMismatch

Equations are formulated and solved in the form  $f(x)=0$ . The number of equations for a QDSL model is equal to the number of outputs defined. Here the currently calculated value for the equation of a certain index is obtained.

**double GetEquationMismatch(int eqIdx)**

**Arguments:**

*eqIdx*: The index of the equation to get the value (zero-based).

**Return value:**

Returns the value of the equation at index *eqIdx* on success. If not successful, this function returns -1.

**Example:**

The following example shows how the value of equation 0 can be obtained:

```
double mismatch;
mismatch = GetEquationMismatch(0);
printf('The error in equation 0 is %f', mismatch);
```

### SetControlLoopFinished

Function enabling the user to govern the outer loop behaviour of the control scripts for the Load Flow or Quasi-Dynamic simulation. If the function is not called the algorithm decides internally whether the model triggers an additional control loop. If the function is called, the user can decide when the model has converged.

**bool SetControlLoopFinished (bool isFinished)**

**Arguments:**

***isFinished:***

- *isFinished!* = 0 - This model will not trigger another outer loop.
- *isFinished* = 0 - The model has not converged satisfactorily yet. Another outer loop is demanded by the user.

**Return value:**

Returns zero, if the control statement could be successfully set.

**Example:**

The following example shows how the user can decide whether another loop is necessary (in the control script):

```
if (gen:m:P:bus1 > 1.0) {  
    gen:s:pset = 0.5;  
    SetControlLoopFinished(0);  
}  
else {  
    SetControlLoopFinished(1);  
}
```

**IsOutaged()**

Function to determine whether or not an object is currently calculation-relevant, in-service and not outaged (e.g., by a contingency).

**bool object.IsOutaged ()**

**Arguments:** -**Return value:**

Returns one, if object is currently outaged.

**GetControlLdfIteration()**

Function to determine the current outer loop iteration number for the control actions of the current Load Flow calculation (starting with 0). The control script on the Load Flow tab of a Quasi-Dynamic Model Type is executed once per outer loop.

**int GetControlLdfIteration()**

**Arguments:** -**Return value:**

The current outer loop iteration number for the control actions of the current Load Flow calculation. Otherwise -1.

**IsDuringFinalControlQdsScript()**

Function to distinguish whether the current *Quasi-Dynamic Simulation Control block* is executed between time steps as part of the outer loop within a time step, see Figure 28.5.2.

**int IsDuringFinalControlQdsScript()**

**Arguments:** -**Return value:**

Returns one, if called in the *QDS Control block* between time steps. Otherwise 0.

### 28.5.6 Example: Modelling a battery as a *Quasi-Dynamic* user defined model

This section does not intend to give a specification on modelling a battery system, but rather a high level overview of a Battery model which can help the reader to understand the modelling requirements within *Quasi-Dynamic* Simulation.

With reference to Figure 28.5.3 a very simple AC power system that contains a PV unit, an AC load and an AC/DC converter interfaced battery unit is considered. In this arrangement, one possible battery control strategy would be to measure the AC power flow through the supply line such that the contribution of the battery system at a certain moment in time can be correlated with the generated photovoltaic and the consumed load powers.

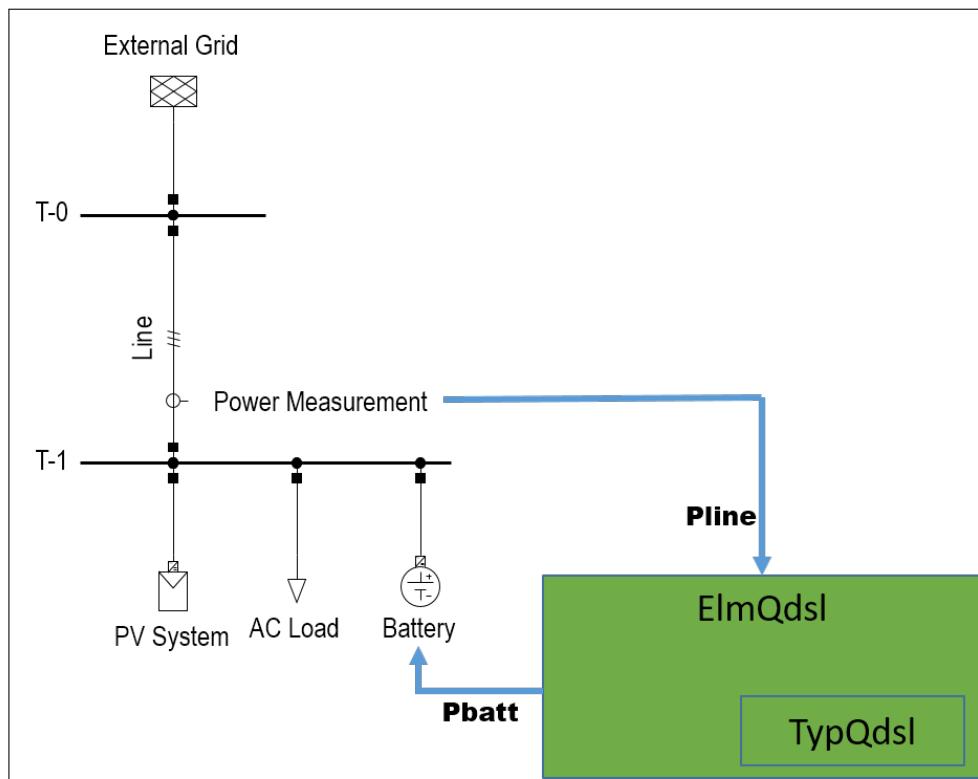


Figure 28.5.3: Example of Battery System considering branch flow measurement

Having knowledge of the power flow through the supply line, the actual power balance between the PV system and the AC load can be easily calculated as:

$$P_{pv} + P_{load} = P_{line} - P_{batt} = P_{meas}$$

A valid battery control strategy (refer to Figure 28.5.4) would be to verify the power throughput  $P_{meas}$ . For the periods when  $P_{meas}$  is positive (hence the PV system is generating more power than the load can consume) the battery system may charge to replenish the state of charge. For the periods when  $P_{meas}$  is negative (hence the load consumes more power than the PV can generate) the battery system may discharge to minimise the power net import from the supply network. A deadband may also be introduced in order to avoid unwanted behaviour (using  $P_{StartFeed}$  and  $P_{StartStore}$  thresholds). Hence, a charging/discharging operation mode can be identified based on the power flow through the supply line.

This operation mode,  $chargeP$  is defined as below:

$$chargeP = \begin{cases} 1, & \text{battery charging} \\ 2, & \text{battery inactive} \\ 3, & \text{battery discharging} \end{cases} \quad (28.2)$$

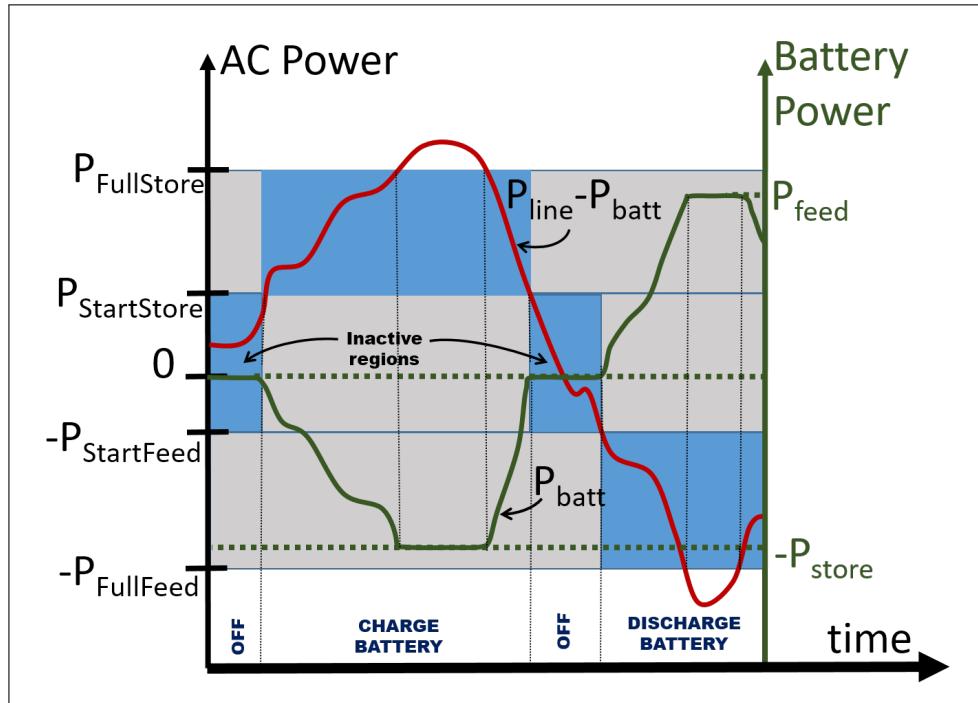


Figure 28.5.4: Example of a generic battery control strategy

An important parameter of a battery model is the battery's state of charge (expressed in %) and defined as the percentual fraction of the energy still available in the battery (in  $MWh$ ) over the total energy when the battery is fully charged (denoted by  $C$  and expressed in  $MWh$ ). The state of charge is hence a state variable of a battery model, with  $0 \leq SOC \leq 100$ . In its most simple representation of a battery, the time dependence of the battery's state of charge  $SOC$  can be defined by the following differential equation:

$$\frac{d}{dt} SOC = \frac{-P_{batt} \cdot 100}{C \cdot 3600}$$

where  $P_{batt}$  is the AC power (in MW) flowing through the battery branch, under the assumption of a unity transfer efficiency between the AC and DC side of the battery converter system.

Further considerations must be taken in the *QDSL model* in order to limit the charging/discharging modes of the battery depending on the current  $SOC$  at any given moment in time. This operation mode,  $chargeE$  is defined as below:

$$chargeE = \begin{cases} 1, & SOC \leq SOC_{\min} \\ 2, & SOC_{\min} \leq SOC \leq SOC_{\max} \\ 3, & SOC \geq SOC_{\max} \end{cases} \quad (28.3)$$

where  $SOC_{\max}$  and  $SOC_{\min}$  are the maximum and the minimum allowed state of charge setpoints respectively, and  $0 \leq SOC_{\min} \leq SOC_{\max} \leq 100$ .

The model state variables and parameters can be defined as provided in Table 28.5.3. These parameters are accessible from all model scripts.

<b>state variable</b>	<b>SOC</b>	<b>%</b>	<b>State of charge</b>
parameter	Eini	MWh	Storage Energy Size
parameter	SOCini	%	Initial state of charge
parameter	SOCmin	%	Minimal state of charge
parameter	SOCmax	%	Maximal state of charge
parameter	Pstore	MW	Rated charging power
parameter	PFULLStore	MW	Pmeas where maximum charging is reached
parameter	PStartStore	MW	Pmeas where charging starts
parameter	Pfeed	MW	Rated discharging power
parameter	PStartFeed	MW	Pmeas where discharging starts
parameter	PFULLFeed	MW	Pmeas where maximum discharging is reached
parameter	orientation		1=terminal j is closest, otherwise -1

Table 28.5.3: Parameters and state variables of user defined *Quasi-Dynamic* model

The model results can be defined as below. The results include the operation modes *chargeP* and *chargeE*, which can be used in the model scripts as well (they can be updated over time, depending on the grid situation):

<b>Pmeas</b>	<b>MW</b>	<b>Measured power (PV and load)</b>
chargeE		Operation area w.r.t. energy
chargeP		Operation area w.r.t. power
iniSOCob		Initial SOC out of bounds flag

Table 28.5.4: Results of user defined *Quasi-Dynamic* model

The **Initialisation** script (block A in Figure 28.5.2) must deal with initialising all the operation modes and the initial state of charge of the battery, as shown here:

```
double pmeas;
SOC = SOCini;
pmeas = 0.; ! arbitrary value being provided; load flow has not been yet executed
! measured power operation area
chargeP = 0.;
if ({PFullStore <= PStartStore}.or.{-PStartFeed <= -PFullFeed}) {
    chargeP = 0; ! Error
    Warn('PFullStore must be > than PStartStore and PFullFeed > than PStartFeed');
} else if (pmeas < PStartFeed) chargeP = 3;
else if (pmeas > PStartStore) chargeP = 1;
else chargeP = 2;
! energy operation area
iniSOCob = 0; ! Inside bounds
if (SOCmin >= SOCmax) {
    chargeE = 0; ! Error
    Warn('SOCmin must be < than SOCmax.');
}
else if (SOC > SOCmax) {
    chargeE = 3;
    iniSOCob = 1;
}
else if (SOC = SOCmax) chargeE = 3;
else if (SOC = SOCmin) chargeE = 1;
else if (SOC < SOCmin) {
```

```

chargeE = 1;
iniSOCob = 1;
else chargeE = 2;

```

The load flow outer loop equations ([LDF Control](#) script, i.e. block **C** in Figure [28.5.2](#)) are responsible with changing the charging/discharging operation mode *chargeP* such that the battery system is operating on the correct region as defined in Figure [28.5.4](#). Additionally, the verification of the current value of the state of charge *SOC* must be considered by appropriately updating the *chargeE* operation mode. Based on the previous considerations, the load flow control DPL script can be programmed as shown here:

```

Pmeas = Pline*orientation - Pbatt; ! negative=load
! measured power operation area
if (chargeP > 0) { ! Not initial error
    if (Pmeas < -PStartFeed)      chargeP = 3;
    else if (Pmeas > PStartStore) chargeP = 1;
    else                           chargeP = 2;
}
! energy operation area
if (chargeE > 0) { ! Not initial error
    if (SOC >= SOCmax) {
        chargeE = 3;
        if ({iniSOCob = 0}.and.{SOC > SOCmax}) {
            SOC = SOCmax;
        }
    }
    else if (SOC <= SOCmin) {
        chargeE = 1;
        if ({iniSOCob = 0}.and.{SOC < SOCmin}) {
            SOC = SOCmin;
        }
    }
    else {
        chargeE = 2;
        iniSOCob = 0; ! Inside limits now
    }
}

```

The load flow inner loop equations ([LDF Equations](#)) must consider a fixed state of the battery model (i.e. the charge operation modes, *chargeP* and *chargeE*, the state of charge of the battery *SOC* and any other operational flags do not change within this script). The amount of power being consumed/generated by the battery in the discharge/charge operation modes can be defined as below using a linear function that considers a reduction factor depending on the value of *P<sub>meas</sub>* with respect to four other thresholds (*P<sub>FullFeed</sub>*, *P<sub>StartFeed</sub>*, *P<sub>FullStore</sub>* and *P<sub>StartStore</sub>*):

$$P_{\text{batt}} = \begin{cases} P_{\text{feed}} \cdot \left(1 - \frac{P_{\text{meas}} + P_{\text{FullFeed}}}{P_{\text{StartFeed}} + P_{\text{FullFeed}}}\right), & \text{discharging} \\ -P_{\text{store}} \cdot \left(1 - \frac{P_{\text{FullStore}} - P_{\text{meas}}}{P_{\text{FullStore}} - P_{\text{StartStore}}}\right), & \text{charging} \\ 0, & \text{battery inactive} \end{cases}$$

The measured and the controlled quantities of the *QDSL model* are, as shown in Figure [28.5.3](#), *P<sub>line</sub>* and *P<sub>batt</sub>*. They are declared within the *Load Flow* page of the *QDSL type* as below. Further to that, within the *Load Flow* page of the *QDSL element* the actual network elements (the battery element and the supply line) will need to be assigned.

Name	Usage	Class name	Variable name	Bus/phase name
Pbatt	Output	ElmGenstat	pset	
Pline	Input	ElmLne	Psum	bus2

Table 28.5.5: Inputs/Outputs for load flow equations of *QDSL type*

Considering the previous observations, the [LDF Equations](#) script (block **B** in Figure 28.5.2) can be defined using DPL as shown below. Note the use of the DPL function [SetEquation](#), which provides the statement for controlling the output power of the battery  $P_{\text{batt}}$  (static generator element in *PowerFactory*) to the calculated setpoint  $P_{\text{gen}}$ . Note also the generator sign convention chosen when computing the setpoint  $P_{\text{gen}}$  such to comply with the requirements of the *Static Generator* element (positive  $P_{\text{gen}}$  means discharging, negative  $P_{\text{gen}}$  means charging, as in Figure 28.5.4).

```
double Pgen,
      redFac;
Pmeas = Pline*orientation - Pbatt; ! negative=load
redFac = 1.0;
if ({chargeP = 3}.and.{chargeE >= 2}.and.{chargeE > 0}) {
    if (Pmeas > -PFullFeed)
        {redFac = 1 - ((Pmeas + PFullFeed)/(-PStartFeed + PFullFeed));}
    Pgen = Pfeed * redFac; ! discharge = GEN, feeding
}
else if ({chargeP = 1}.and.{chargeE <= 2}.and.{chargeE > 0}) {
    if (Pmeas < PFullStore)
        {redFac = 1 - ((PFullStore - Pmeas)/(PFullStore - PStartStore));}
    Pgen = -Pstore * redFac; ! charge = LOAD, storing
}
else {
    Pgen = 0.;
}
SetEquation(0, Pbatt - Pgen);
```

The [QDS Equations](#) script (block **E** in Figure 28.5.2) must contain the statements for the time derivatives and other time-dependent variables. In this case, the only time-dependent quantity is the state of charge variable *SOC*. Hence, the *Quasi-Dynamic Equations* script contains this DPL code:

```
SOC. = -Pbatt * 100. / (Eini * 3600.); ! slope of charge/discharge in %
```

The [QDS Control](#) script (block **D** in Figure 28.5.2) is only a variation of the load flow *Control* script (since no additional simulation events are generated), as shown below:

```
! energy operation area
if (chargeE > 0) { ! Not initial error
    if (SOC >= SOCmax) {
        chargeE = 3;
        if ({iniSOCob = 0}.and.{SOC > SOCmax}) {
            SOC = SOCmax;
        }
    }
    else if (SOC <= SOCmin) {
        chargeE = 1;
        if ({iniSOCob = 0}.and.{SOC < SOCmin}) {
            SOC = SOCmin;
        }
    }
    else {
        chargeE = 2;
        iniSOCob = 0; ! Inside limits now
    }
}
```

Implementing the model and running one simulation for a generic load curve, the results shown in Figure 28.5.5 and 28.5.6 are obtained.

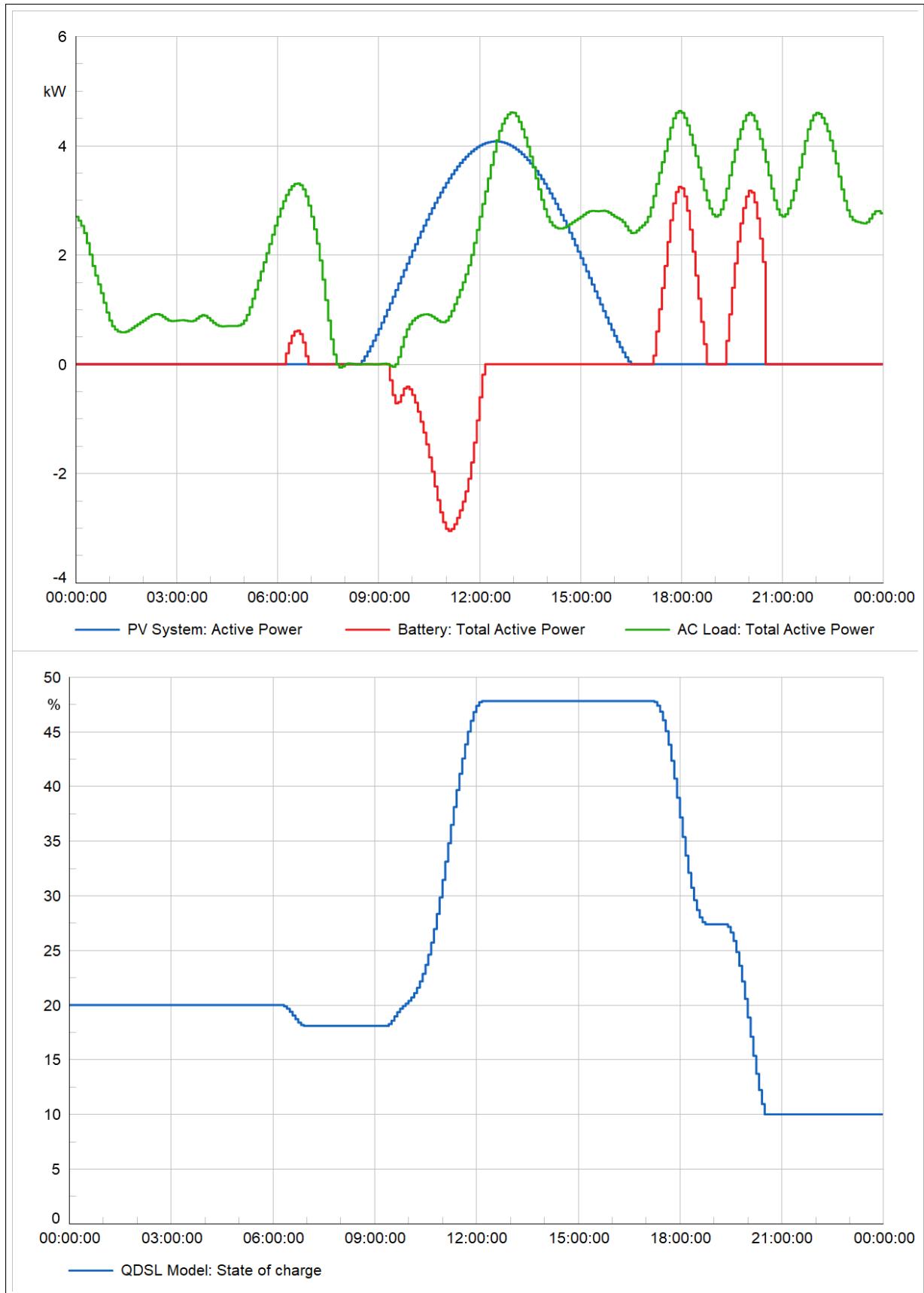


Figure 28.5.5: Power Flow and state of charge SOC behaviour during a one day simulation

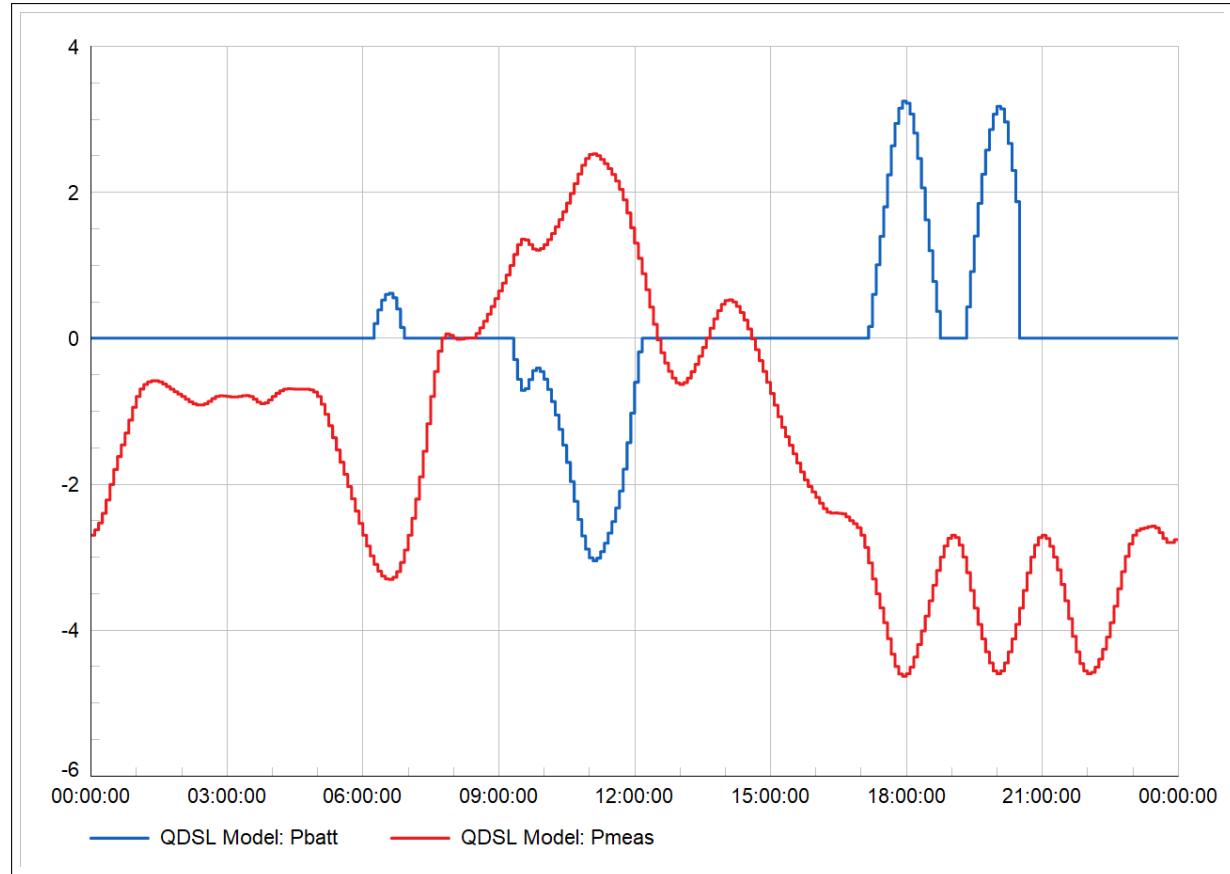


Figure 28.5.6:  $P_{\text{meas}}$  - sum of PV and Load power in kW;  $P_{\text{batt}}$  - battery power output in kW

# Chapter 29

## RMS/EMT Simulations

### 29.1 Introduction

The RMS/EMT simulation functions available in *PowerFactory* are able to analyse the dynamic behaviour of small and large power systems. The underlying framework makes it possible to model a wide range of complex systems, such as large transmission grids, renewable generation plants or industrial networks, while taking into account electrical, mechanical and control parameters.

Transients, stability analyses and dynamic control problems are important during the planning, design and operation of modern power systems. Such studies can be conducted using time-domain simulations for varying time periods. A wide range of either ac, dc or ac/dc based systems may be necessary to be analysed (e.g. transmission systems with detailed models of power plants, HVDC systems, power electronics based generation, railway systems, variable speed drives motor start-up).

Transients in electrical power systems can generally be classified according to three possible time frames of interest:

- short-term, or electromagnetic transients;
- mid-term, or electromechanical transients;
- long-term transients.

*PowerFactory* supports the following time-domain simulation methods (described in Section 29.2):

- *RMS balanced* - A basic function which uses a symmetrical steady-state (RMS) network model for mid-term and long-term transients under balanced network conditions (refer to Section 29.2.1);
- *RMS unbalanced* - A three-phase function which uses a steady-state (RMS) network model for mid-term and long-term transients under balanced and unbalanced network conditions, i.e. for analysing dynamic behaviour after unsymmetrical faults (refer to Section 29.2.2);
- *Instantaneous values* - An electromagnetic transient (EMT) simulation function using a dynamic network model for electromagnetic and electromechanical transients under balanced and unbalanced network conditions. This function is particularly suited to the analysis of short-term transients (refer to Section 29.2.3).

The simulation methods use an iterative procedure to solve AC and DC load flows at any given time point (algebraic equations), along with the solution for dynamic model state variables (differential equations).

Before the start of a simulation, the user may configure what type of network representation should be used in the simulation, what step sizes to use, which events to handle and where to store the results.

Upon start of a simulation, *PowerFactory* determines the initial conditions for all power system elements (including all controller units and mechanical components) based on a valid load flow calculation

(described in Chapter 25).

The process of performing a dynamic simulation typically involves the following steps:

- Calculation of initial values, including a load flow calculation (refer to Section 29.3)
- Definition of result variables (refer to Section 29.5)
- Definition of simulation events (refer to Section 29.6)
- Execution of simulation (refer to Section 29.7)
- Creation of simulation plots (refer to Section 29.8)
- Exporting results (optional, refer to Section 19.7.1).

For all simulation methods, *PowerFactory* provides a *Simulation Scan* function which greatly simplifies the task of monitoring power system wide variables (e.g. voltage, frequency, generator loss of synchronism) and issuing corresponding actions. This function is documented in Section 29.9.

Based on the aforementioned sequential time domain simulation methods (RMS balanced, RMS unbalanced and EMT), *PowerFactory* supports the following co-simulation functions:

- Single/Multiple Domain co-simulation (described in Section 29.11)
- Co-simulation with external solver (described in Section 29.12)

Whenever dynamic models need to be analysed in frequency domain, the *Frequency Response Analysis* function can be employed, enabling users to create Bode or Nyquist plots. Refer to Section 29.13 for more information.

If time domain curves (e.g. simulation results) need to be analysed in frequency domain, *PowerFactory* provides two signal frequency analysis functions, as documented in Section 29.14:

- Fourier Analysis, using Fast Fourier Transform (FFT), and
- Prony Analysis.

User defined dynamic models can be developed, configured and included in a dynamic simulation. These models are based on the dynamic modelling framework, described in Chapter 30.

For equipment and controls whose physical dynamic behaviour is known (e.g. via test measurements) the *System Parameter Identification* function can be used along with a non-parameterised simulation model in order to obtain an accurate representation of the system. More details are available in Chapter 31.

*PowerFactory* provides a dedicated toolbar for accessing the different dynamic simulation commands. The toolbar can be shown by clicking the *Change Toolbox* button and selecting *Simulation RMS/EMT*. It is also possible to execute some of the fore mentioned functions via the main menu: *Calculation* → *Simulation RMS/EMT*. The following functions are available in the toolbar:

- *Calculate Initial Conditions*
- *Start Simulation*
- *Stop Simulation*
- *Create Simulation Plot*
- *Initial Conditions for co-simulation*
- *Prepare co-simulation with ext. solver*
- *Save Snapshot*

- *Load Snapshot* 
- *Edit Result Variables* 
- *Edit Simulation Events* 
- *Edit Simulation Scan* 
- *Calculation of Frequency Response* 
- *System Parameter Identification* 

## 29.2 Calculation Methods

### 29.2.1 Balanced RMS Simulation

The balanced RMS simulation function considers dynamics in electromechanical, control and thermal devices. It uses a symmetrical, steady-state representation of the passive electrical network. Using this representation, only the fundamental components of voltages and currents are taken into account.

Depending on the models of generators, motors, controllers, power plants and motor driven machines used, the following studies may be carried out:

- transient stability (e.g. determination of critical fault clearing times);
- mid-term stability (e.g. optimisation of spinning reserve and load shedding);
- oscillatory stability (e.g. optimisation of control device to improve system damping);
- motor start-up (e.g. determination of start-up times and voltage drops);

Various events can be included in the simulation, for example:

- start-up and/or loss of generators or motors;
- stepwise variation of loads;
- load-shedding;
- line and transformer switching/tripping;
- symmetrical short-circuit events;
- insertion of network elements;
- power plant shut down;
- variations of controller setpoint;
- change of any system parameter.

Because of the symmetrical network representation, the basic simulation function allows the insertion of symmetrical faults only.

### 29.2.2 Three-Phase RMS Simulation

If asymmetrical faults or unbalanced networks have to be analysed, the three phase RMS simulation function must be used. This simulation function uses a steady-state, three-phase representation of the passive electrical network and can therefore compute unbalanced network conditions, either due to unbalanced network elements or due to asymmetrical faults. Dynamics in electromechanical, control and thermal devices are represented in the same way as in the basic RMS simulation function.

Asymmetrical electromechanical devices can be modelled, and single-phase and two-phase networks can also be analysed using this analysis function.

In addition to the balanced RMS simulation events, unbalanced fault events can be simulated, such as:

- single-phase and two-phase (to ground) short-circuits;
- phase to phase short-circuits;
- inter-circuit faults between different lines;
- single- and double-phase line interruptions.

All of these events can be modelled to occur simultaneously or separately, hence any combination of symmetrical and asymmetrical faults can be modelled.

### 29.2.3 Three-Phase EMT Simulation

Voltages and currents are represented in the EMT simulation by their instantaneous values, so that the dynamic behaviour of passive network elements is also taken into account. This is necessary for the following applications:

- DC and harmonic components of currents and voltages;
- Exact behaviour of inverter-driven machines;
- Exact behaviour of HVDC transmission systems;
- Non-linear behaviour of passive network elements such as transformer saturation;
- Over-voltage phenomena in switching devices;
- Lightning strikes and travelling waves;
- Analysis of the exact behaviour of protection devices during faults.

The high level of detail used to represent the modelled network means that all phases and all defined events (symmetrical and asymmetrical) can be simulated. The EMT function can also be used for the simulation of longer-term transients. However, due to the passive network elements being represented dynamically, the integration step size has to be significantly smaller than in the case of a steady-state representation and as a result, the calculation time increases.

## 29.3 Calculation of Initial Conditions command

The *Calculation of initial Conditions* command (*ComInc*) dialog allows the configuration of simulation solver settings, such as the simulation type (i.e. RMS or EMT, balanced or unbalanced) and simulation step size.

The available configuration pages of the *Calculation of initial Conditions* command dialog are:

- **Basic Options:** selection of simulation type (RMS, EMT; balanced, unbalanced), load flow command, results object, event list and reference system.
- **Step Size:** start time, integration step sizes and output step size can be specified here, along with step size adaptation parameters.
- **Solver Options:** includes various iteration, integration, algorithm and event control parameters.
- **Simulation Scan:** simulation scan options.
- **Noise Generation:** defines parameters of the noise generation for stochastic applications.
- **Real Time:** defines parameters for real-time applications.
- **Snapshot:** save snapshot options.

### 29.3.1 Initial Conditions - Basic Options

The *Basic Options* are used to select the simulation type and the network representation. References to the results object (described in Section 29.5), the event list (described in Section 29.6) and the load flow command are available for inspecting or editing these objects, by clicking on the respective → icon.

#### 29.3.1.1 Basic Options - General

##### Verify initial conditions

If the initial conditions can be fulfilled, the power system will be in a steady-state condition. When the *Verify initial conditions* option is enabled, then the condition  $dx/dt = 0$  is checked for all state variables. If one or more of the state variable derivatives does not equal zero, the power system may start 'moving' from the very beginning of the simulation, even without the application of an external event. In this case the user should carefully analyse the relevant controller or model and its defined initial conditions.

All warnings or error messages issued in the output window should be checked. Typical problems include devices which are overloaded or operate above or below signal limits from the beginning of the simulation.

An error message displayed in the output window may appear as follows:

```
Some models could not be initialised.  
Please check the following models:  
'Simple Grid AVR Common Model.ElmDsl':  
Initial conditions not valid !
```

##### Automatic step size adaptation

This option enables the step size adaptation algorithm, and can be used to considerably speed-up the simulation. *PowerFactory* adjusts the step size to the actual course of the dynamic simulation. Based on the local truncation error, *PowerFactory* calculates an optimal step size that keeps the numerical error within the specified limit. A step size controller adjusts the integration step size. As a result, when fast transients have decayed, *PowerFactory* automatically increases the step size and speeds up the simulation process considerably.

In the case of events (external or internal), the step size is always set back to the minimum step size. This way, the behaviour of the system during a transient event is represented with the best accuracy.

If the error exceeds the specified limit or the simulation does not converge with an integration step larger than the minimum step, the simulation algorithm steps back and repeats the integration with a smaller integration step in order to reduce the error or to achieve convergence.

Further parameters to adapt this algorithm are defined on the *Step Size* page (see Section 29.3.2).

### Reuse previous load flow results

When the calculation of initial conditions is carried out at the start of a simulation, a load flow needs to be run. If many simulations are to be performed and therefore the initial conditions must be repeatedly recalculated, the repeated load flows could cause a considerable time overhead. If the option is selected, the load flow results from the calculation of initial conditions are retained and used in subsequent initial conditions calculations.

#### 29.3.1.2 Basic Options - Reference system

The reference defines the reference frequency for RMS simulation. Table 29.3.1 gives an overview of the resulting reference frequency, depending on the settings for *reference* and *reference system area*. The *isolated areas* refer to the isolated areas detected by the topology check of the load flow calculation.

##### Reference

- Element: the reference of the system is a single element.
- Centre of inertia: the reference is the calculated centre of inertia.
- Nominal frequency: the reference is constant throughout the simulation and equal to 1 p.u.

##### Reference system area

- Global: one single reference is used when executing the simulation. It should be used if the separated areas are re-synchronised again later in the simulation.
- Local (individual in each isolated area): this is the default option. A local reference is selected for each isolated area.

Reference	Reference System Area	Isolated Areas of the same nominal frequency	Isolated Areas of different nominal frequency
Element	Global	The reference frequency is taken from one reference element for the whole power system.	Reference frequencies are calculated for isolated areas of the same nominal frequency. Isolated areas of different nominal frequency have a different reference frequency. Isolated areas of the same nominal frequency have the same reference frequency.
Element	Local	The reference frequency is calculated for each isolated area.	The reference frequency is calculated for each isolated area.
Centre of inertia	Global	The reference frequency is calculated according to the centre of inertia of all machines in the whole power system.	The reference frequency is calculated according to the centre of inertia of all machines in the whole power system (irrespective of the individual nominal frequencies of isolated areas, as the global reference frequency is expressed in p.u.).
Centre of inertia	Local	The reference frequency is calculated individually for each isolated area, according to the centre of inertia of the machines of the isolated area.	The reference frequency is calculated individually for each isolated area, according to the centre of inertia of the machines of the isolated area.
Nominal frequency	N/A	The reference is always 1 p.u. nominal frequency.	The reference is always 1 p.u. nominal frequency.

Table 29.3.1: Reference systems for simulation

### Reference system calculation method

If the option *Element* is selected from the *Reference* frame, the additional options are available:

- Implicit: this calculates the angle in every iteration, i.e. introduces additional dependencies in the system equations.
- Explicit: this uses the angle value calculated in the previous time step to avoid additional dependencies.

After running initial conditions, the reference element(s) is (are) displayed in the output window (only if the option *Element* was selected as reference).

### Synchronous machine out of step detection

The out of step detection is based on the angle  $f_{irel}$  (rotor angle of the synchronous machine with respect to the rotor angle of the local reference). It is functional only for RMS type simulations. Two options are available:

- Out of step is detected when the rotor angle  $f_{irel}$  reaches the detection angle.
- Out of step is detected when the rotor angle  $f_{irel}$  changes by the detection angle from its initial operating point.

The detection angle can be changed by the user and its default value is 360 degrees.

### Calculate maximum rotor angle deviation

*PowerFactory* can also calculate the maximum deviation between the rotor angles of the synchronous machines in the system. This variable, called *dfrotx*, can then be selected for display from the variables of all synchronous generators in the system. It can be used as an indicator for the synchronous operation of a large transmission system.

## 29.3.2 Initial Conditions - Step Size

In this page additional options are defined depending on whether the option *Automatic step size adaptation* on the *Basic Options* page is activated or deactivated.

### 29.3.2.1 Step Size - General

The dynamic simulation is based on a variable step size simulation solver. During periods of no discrete simulation events, the simulation step size will be chosen based on the configuration provided in the *Integration step size* pane, as detailed below.

#### Integration step size

If the *Automatic step size adaptation* flag is not set, then a single reference value can be configured for the integration step size.

- Electromechanical transients (typical value: 10 ms)
- Electromagnetic transients (typical value: 0.1 ms)

If the *Automatic step size adaptation* flag is set, then a reference value range can be configured for the integration step size.

- *Electromagnetic transients / Electromechanical transients*: Minimum step size for the simulation.
- *Maximum step size*: Maximum step size for the simulation.

#### Start time

The start time of the simulation. This is typically negative, allowing the first event to be analysed to take place at t=0 s.

---

**Note:** When setting up time-domain simulations, it is very important to use the correct time steps in order to be able to observe phenomena in the results. For the RMS simulation the minimum time step should always be smaller than the time constants in the system. In controllers one must consider both the open-loop and the closed-loop time constants. For electromagnetic transients, e.g. when analysing travelling waves, the smallest travelling time would be the upper limit for the minimum time step.

---

In addition to the Newton-Raphson based algorithm for the solution of “weak” non-linearities (i.e. saturation effects in synchronous and asynchronous machines), the simulation function allows interrupts for the simulation of “strong” non-linearities (i.e. switches, two-slope transformer saturation or thyristors). These interrupts can also occur between time steps. In the case of this kind of interrupt, all time-dependent variables are interpolated to the instant of interrupt and the simulation restarts at that point. This prevents numerical oscillations.

### Enforced synchronisation

This option can be used to get simulation results at every specified time. E.g. if the *period* of the enforced synchronisation is set to 1 s there will additional results at every second regardless of step size.

### Record results

It is often unnecessary to plot every single calculated time step, and this reduction in plotted data can also result in a reduced simulation time. For this purpose the output sampling step for the output graphs can be set, so that not every point in time throughout the simulation time will be plotted. By selecting a larger output sampling step, the simulation process will speed up without influencing the calculation process. It should be noted, however, that fast changes may not be seen in the reported results. The following options are available:

- After interruption or lapse of output step: the results will be recorded at every *output sampling step* which is defined by entering a *sampling ratio*.
- At synchronised point in time: the *period* defined in the enforced synchronisation field will be used to determine the sampling step.

#### 29.3.2.2 Step Size - Automatic Adaptation

If option *Automatic step size adaptation* is enabled on the *Basic Options* page, further step size options are available on the *Automatic Adaptation* tab. These options are:

- Reset automatic step size at interruption: the step size is set to the minimum after any interruption
- Use maximum step size at start: this option can be selected to speed up the beginning of the simulation.
- Advanced step size algorithm selected: if the option is selected, the step size algorithm is based on the integration of the prediction error. In addition, the algorithm is looking for an optimal step size. The options of this method are:
  - Maximum prediction error (typical value: 0.01)
  - Time constant for RMS/EMT Simulation
- Advanced step size algorithm not selected: if the option is un-selected, the step size algorithm is based on the increase and decrease by speed factor and prediction error. The options of this method are:
  - Maximum prediction error (typical value: 0.01)
  - Minimum prediction error (typical value: 0.001)
  - Delay for step size increase (number of steps) (typical value: 10)
  - Speed factor: increase (default value: 1.5)
  - Speed factor: decrease (default value: 2)
  - Maximum increase of step size for RMS (typical value: 0.05 s)
  - Maximum increase of step size for EMT (typical value: 0.001 s)

---

**Note:** The simulation time can be very sensitive to some of these parameters. For example, when the maximum time step is increased, the duration of calculating transients may not always decrease. If this time step is increased over an “optimal” time step the simulation time may increase as well. It is strongly recommended to critically observe the simulation time and the results for different simulation parameters.

---

### 29.3.3 Initial Conditions - Solver Options

The solver options may be used to tune the performance of the simulation algorithm. Less experienced users are recommended to use the default values.

#### 29.3.3.1 Solver Options - General

##### Integration control

- Maximum error for dynamic model equations (typical value: 0.1 %)
- *Damping factor* for RMS (typical value: 1)
- *Damping factor* for EMT (typical value: 0.99)

---

**Note:** The *damping factor* range is between 0 and 1. A value of 0 corresponds to use as numerical integration method the *Backward Euler* (implicit) method. If set to 1 then the trapezoidal integration (implicit) method is applied. A value between these two results in a combination of the two methods.

---

If the simulation method is set to EMT, and the *automatic step size adaptation* option is selected, the integration factors can be adapted by using the additional option *Apply AC-adaptation*.

##### Iteration control

- Maximum error for bus equations: the iteration error for bus equations depends on the rated power of the machines and the voltage levels. As an adequate starting value, should be set to: errsm = 10\*errlf, where errlf is the maximum acceptable load flow error for each node. Checking is best done by plotting some voltages at generator busbars. If voltage steps are observed, the value of errsm should be reduced.
- Maximum error for network model equations:  
this error can be entered separately for the high, medium and low voltage level. The thresholds of these levels can be defined in the project settings.  
(typical value: 1 %)
- Maximum number of iterations: specifies the maximum number of iterations at each integration step which are allowed to reach the maximum tolerable bus-error errsm. During the transient simulation process, the typical number of iterations required is between 1 and 5. Under certain conditions, i.e. after switching operations, up to 25 iterations may be observed. (typical value: 25)
- Iteration limit to recompute Jacobian matrix (typical value: 5)

##### Simplifications

The use of these options will result in a faster simulation.

- Fast connection of A-stable models outputs: optimises the output equations formulation for A-stable models.
- Fast convergence check: determines convergence via advanced heuristics instead of checking each single equation.
- Fast computation of outputs: uses the last available output from the solution process instead of recomputing each single quantity.

- Fast independent solution of network and dynamic models: determines convergence sequentially and independently on network and dynamic models rather than simultaneously.

---

**Note:** A requirement for using the independent solution of network and dynamic models algorithm is that “small” integration steps are used with respect to the dynamics involved.

---

### 29.3.3.2 Solver Options - Models

#### Initialisation

- Solve dynamic model equations at initialisation: the dynamic equations of the non-A-stable models are solved at initialisation together with network and A-stable models which are always solved.
- Issue warnings for multiple initialisation of signals: a warning is issued when a DSL model input signal is initialised by two different models.

#### A-stable integration algorithm

The A-stable integration algorithm is only used in RMS-simulation.

If this option is enabled, *PowerFactory* uses an A-stable numerical integration algorithm for models to solve the simulation. In this case the dynamic model equations and network equations are solved simultaneously. This algorithm is (slightly) slower for small step sizes but convergence is improved for large step sizes. Typical applications are long-term simulations, in which the simulation step size is increased considerably after fast transients have decayed. Another typical application is systems with power electronics. Even if power electronics devices are usually equipped with very fast controls, the A-stable algorithm still allows reasonable step sizes, at which the relaxation method would fail.

When using a conventional partitioned method (not an A-stable algorithm), the integration step size must be adjusted to the eigenvalues of a system. Such a method means a mutual solution of dynamic model equations and network equations until convergence is reached: this algorithm is fast for small step sizes but fails to converge when the step size is increased. This is the best choice for classical transient stability applications, but if excessively large step sizes are used, the numerical solution becomes unstable, even if fast modes have fully decayed and are no longer apparent in the system.

With the *PowerFactory* A-stable algorithm, the step size can be adjusted to the actual course of all state variables without considering numerical stability. When fast transients have decayed, the step size can be adjusted to the speed of slower transients, etc.

If some very fast modes are not of interest, a large step size can be selected from the beginning, and the algorithm will automatically smooth fast variations. A typical application of this type of algorithm is the simulation of long-term phenomena, where it is necessary to increase the simulation step size to the range of minutes, even if fast modes are present in the system.

However, if power electronics are involved, characteristic time constants can be extremely short (i.e. 1 ms), even if a stability model with steady-state equations for the electrical network is used. Hence, using a classical integration algorithm would require the use of step sizes significantly smaller than the smallest time constant of the system, otherwise it would be numerically unstable.

---

**Note:** A requirement for using the A-stable integration algorithm is that only “true” input and output signal variables are used for exchanging information between different models.

---

It is also possible to specify the usage of an A-stable algorithm for some element models only (i.e. not for all models), so that it is possible to run only a portion of the models with the A-stable algorithm (for example the power electronic converters or fast controllers). This option is available in the dialogs of the elements.

With the A-stable algorithm, these systems can be analysed with reasonable step sizes. Hence, the A-stable algorithm cannot be described as using simplified models but as a different type of numerical integration algorithm.

There are three options available:

- Apply per element: the algorithm is applied only in the elements with the *A-stable* flag selected.
- Apply to all elements: the algorithm is applied to all the elements.
- Apply per element and composite model: the algorithm is applied to all the elements inside the composite model if the *A-stable* flag is selected in at least one of those elements.

## DSL

- Direct application of events: all internal DSL events stemming from `select()`, `lim()` and `limstate()` functions are applied directly within one step.
- Fast direct interpolation of buffers: allows for faster interpolation of buffers (delay and movingavg) which leads to better performance and convergence in simulations that may use integration steps which are larger than the delay or the movingavg constants of the model.
- Automatic compilation: for increased simulation performance, automatically compile all currently non-compiled DSL models of level 5 or higher. *PowerFactory* automatically monitors the state of relevant DSL models, so any subsequent runs on an unchanged DSL model will not trigger a re-compilation. Conversely, if any modification has been applied to a DSL model then a re-compilation is triggered for that specific model. If this option is set, but due to various reasons the compilation process fails either partly or entirely (e.g. no compiler is available, some DSL models cannot be compiled, etc.), then those DSL models which failed to compile will be run as *Interpreted models*. Refer to Section 5.2.4 for information on C-Compiler settings and Section 30.1.5 on general compilation functionality of *PowerFactory* native dynamic models.

## Parallelisation of model equations

- Enable: Enable this function to parallelise the computation of model equations across multiple processor cores. Improved simulation performance is expected for complex systems.
- Disable: This function is disabled.

### 29.3.3.3 Solver Options - Advanced

#### Event control

- *Maximum number of repeat event loops*: maximum number of times that a given integration step can be repeated in order to schedule upcoming events. Once the maximum number is reached, any further events will not be scheduled until the next step.
- *Maximum number of zero-length interruptions*: maximum number of times that an integration step can be restarted after a zero-length interruption. Once the maximum number is reached, any further zero-length interruptions will not be scheduled until the next step.
- *Maximum number of reschedule event loops*: maximum number of times that a rescheduling of upcoming events can be called. Once the maximum number is reached, no further rescheduling is allowed until the next step.
- *Resolution factor*: this parameter (*kres*) determines the time interval used to synchronise events. *PowerFactory* executes all events that occur within a time interval of duration *kres \* dtmin* at the same instant in time. A value of 0 implies that events are executed precisely in time, but may lead to slow simulations.

- *Reset integration formula after reinitialisation of algebraic equations.* If an event occurs, the integration formula order is temporarily reinitialised to 1 to avoid numerical oscillations, before being gradually restored to the specified value;
- *Integration formula restoration steps after reset:* number of steps used to gradually restore the specified integration formula after a reinitialisation of its order to 1.
- *Accelerated solution of equation system:* The solution of dynamic equations applies advanced techniques to accelerate the simulation performance, especially in the case of high-frequent clocked systems or in the presence of high-frequent switching devices. In addition, it applies non-precise time point evaluation for clocked models defined via Modelica Model (*EImMdl*).

### Reinitialise algebraic equations at interruption

- *Disable:* if an interruption event occurs, the algebraic equations are not re-initialised at interruption time;
- *Enable for systems containing only AC elements:* if an interruption event occurs, the algebraic equations are reinitialised at interruption time if and only if the power system model contains exclusively AC elements. If this is the case, it leads the solver to the calculation of  $v(t^-)$  and  $v(t^+)$ , hence two values at the same time instant, one before the occurrence of the event, and one after. The two generated value sets are stored in the result file at the same time point.
- *Enable:* if an interruption event occurs, the algebraic equations are reinitialised at interruption time for all power system configurations, with or without DC elements. If enabled, it leads to the calculation of  $v(t^-)$  and  $v(t^+)$ , hence two values at the same time instant, one before the occurrence of the event, and one after. The two generated value sets are stored in the result file at the same time point. If DC elements are existing in the system then results during the interruption event may be inaccurate.

### Behaviour at user-defined events

- *Postprocessing:* wait for the integration step to end, then apply event.
- *Interruption:* stop instantly and interpolate, then apply event.
- *Repetition:* apply event during step, then repeat the integration step.

### Signal buffer

Number of additional signals that can be allocated during the simulation.

### 29.3.4 Initial Conditions - Simulation Scan

Different variables can be monitored during the simulation and events triggered accordingly by the use of the simulation scan.

There are several types of simulation scan modules, described in detailed in Section 29.9. For the modules to be considered, the *Active* flag should be selected on this page.

New modules can be created by clicking on the **Show** button and then on the *New Object* ( icon). The button **Remove all** will delete all the existing simulation scan modules.

### 29.3.5 Initial Conditions - Noise Generation

The *Noise Generator* element (*EImNoise*) can be used in a transient simulation to produce a noise signal based on random numbers. On the *Noise Generation* page of the *ComInc* dialog, the random

number generation method can be selected. The random number generator can be selected to *Automatic*, which is the default value and the most commonly used.

Alternatively, the option *User defined* may be selected, in which case the random seed of the noise generator can be entered manually. The information about the last used seed is also shown, so the results of a former simulation can be reproduced exactly.

### 29.3.6 Initial Conditions - Real Time

The Real-time simulation options can be configured here, as follows.

#### Real-time simulation

- *Off*: No synchronisation with system time is performed. Use this setting for an offline (non real-time) simulation.
- *Synchronised by system time*: Attempts to synchronise the simulation with the system time by periodically pausing the simulation. This setting can not speed up the simulation if it is slower than real time to begin with and thus should only be used if it is known that the simulation can run at or above Real-Time speed. If the simulation effort temporarily increases above real-time (e.g. by an event causing a lot of switching actions), this option will attempt to catch up until it is back in sync with respect to the start of the simulation.
- *Synchronised by system time, no catch-up*: Same as the previous option, attempts to synchronise the simulation with the system time. However this option will attempt to maintain Real-Time behaviour for each synchronisation interval and will not catch up after the simulation effort temporarily exceeds Real-Time.

#### Synchronisation with system time

- *Ratio between real time and calculation time*: The ratio between the elapsed simulation time (within PowerFactory) and the elapsed system time targeted by the Real-Time simulation. A value above 100 % will cause the simulation to proceed faster than real time (e.g. with a value of 200 %, 10 s of simulation time will take 5 s of system time) while a value lower than 100 % will cause the simulation to run slower (e.g. at 50 %, 10 s of simulation time will take 20 s of system time). As above, this setting can not speed up the simulation.
- *Time interval used for synchronisation*: The interval (in simulation time) at which the simulation waits to synchronise with the system time. Setting a value such that interval / factor < 20 ms (where factor is the previous setting) is ineffective.

#### Interface

These options control how communication via protocols such as OPC is handled during Real-Time simulation.

- *Update - at each step*: Communication will happen at every simulation step.
- *Update - use interval*: Communication will happen at the interval set below.
- *Update - Interval*: Interval for communication (in system time)
- *Send and receive after initialisation*: Communication will happen once after the initial conditions have been calculated.

### 29.3.7 Initial Conditions - Snapshot

This page provides further configuration options for the *Save/Load Snapshot* functionality of the time domain simulation. The *Save/Load Snapshot* is described in more detail in Section [29.10](#).

#### 29.3.7.1 Save snapshot via event pane

During the simulation it is possible to define events that can trigger saving a simulation snapshot. The event required is a *Save Snapshot* event type (*EvtSave*).

This pane provides the user the possibility to choose the *PowerFactory* behaviour in the case of such an event, by saving the snapshot:

- *In non-persistent memory slot* or
- *In file*.

If the option *In non-persistent memory slot* is selected then the simulation is saved only temporary in the local memory. As such, the snapshot is lost upon closing the *PowerFactory* application. If the snapshot is intended to be permanently saved, then the option *In file* needs to be chosen, along with a valid directory path to be provided in the subsequent *Directory* field.

#### 29.3.7.2 Load snapshot at initialisation

Activate this checkbox in order to load a snapshot at initialisation. The options are:

- *From non-persistent memory slot* or
- *From file*.

If the option *From non-persistent memory slot* is selected then the snapshot is loaded from memory, as the last saved snapshot of the active study case. The snapshot is lost upon closing the *PowerFactory* application. If the option *From file* is selected, then the snapshot is loaded from the particular snapshot file.

The used snapshot stores and applies the following simulation settings (hence, the parameters within the *Calculation of initial conditions* are ignored):

- Simulation method
- Network representation
- Integration step size parameters

An *Info* pane is available, displaying specific information regarding the used snapshot.

### 29.3.8 Advanced Simulation Options - Load Flow

There are further options which can influence the simulation process and its results. In the load flow command dialog (*ComLdf*, see also Chapter [25: Load Flow](#)) on the *Advanced Options* page, *Simulation* tab, the influence of protection devices or various controller models can be selected to be ignored, in which case the chosen models or protection devices will be ignored during the simulation as well as in load flow and other calculations. This is illustrated in Figure [29.3.1](#).

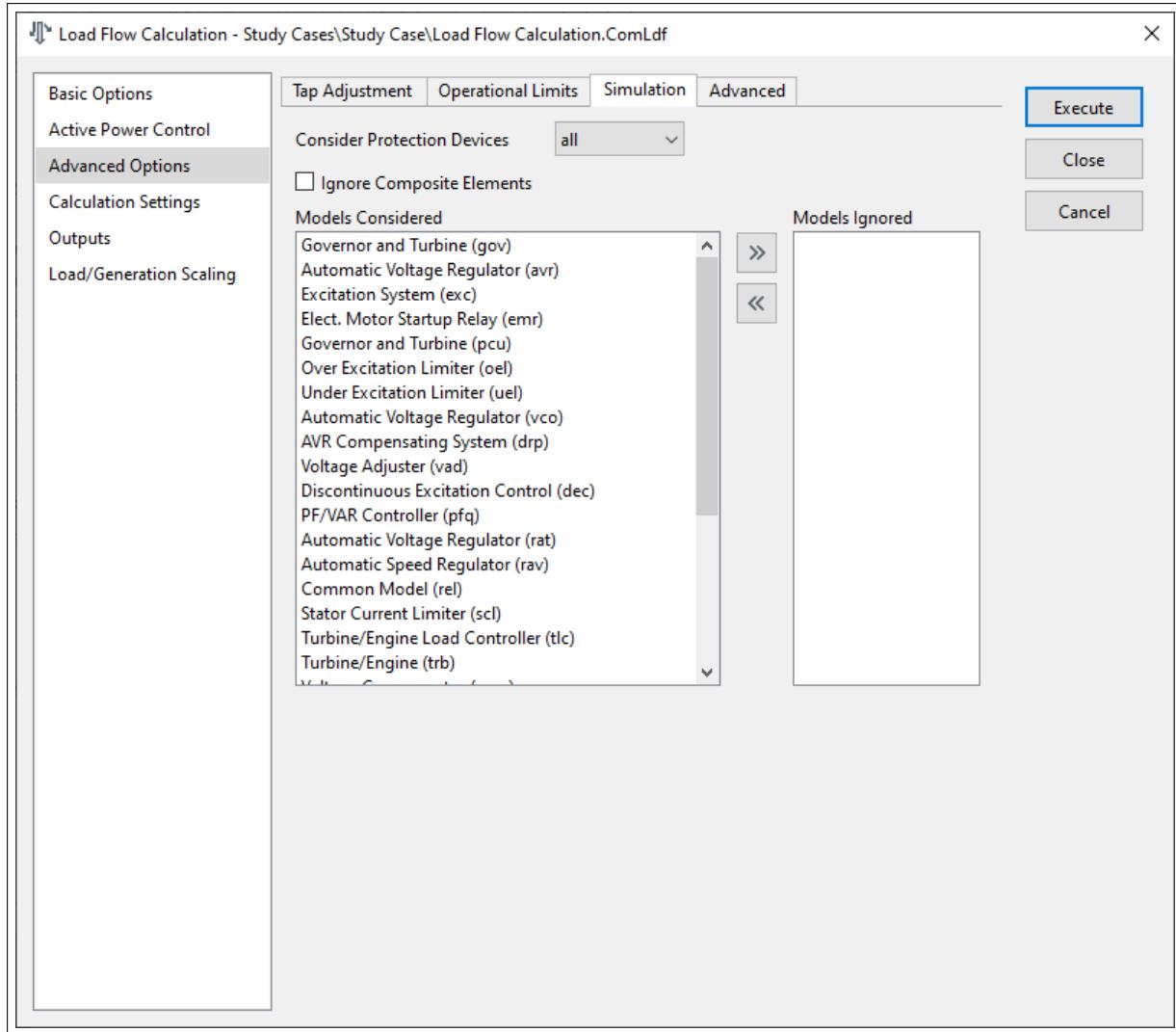


Figure 29.3.1: Advanced Simulation Options in the Load Flow Calculation command dialog

The options available for the consideration of protection devices are:

- **none**: no protection devices are considered in the calculations
- **all**: all protection devices are considered
- **main**: only protection devices in operation which are defined as 'main' devices are considered.
- **backup**: only 'backup' protection devices are considered.

For the controller models, there is the possibility to ignore all controllers and mechanical elements with the option *Ignore Composite Elements*. If only some specific model types should be ignored during the simulation, they can be moved from the left window *Models Considered* to the right window, *Models Ignored*.

## 29.4 Run Simulation command

The *Run Simulation* command (*ComSim*) dialog enables users to run and configure a dynamic simulation, based on a number of configuration options.

The command dialog supports the following actions:

- Run simulation and save changes: Click on the button **Execute** to run the simulation using the chosen settings. Any configuration options that have been currently modified are saved.
- Close dialog and save changes: Click on the button **Close** to close this dialog and save any applied changes to the command options. The dynamic simulation is not executed in this case.
- Close dialog and discard changes: Click on the button **Cancel** to close this dialog and discard any applied changes to the command options. The dynamic simulation is not executed in this case.

The available command options are described below:

#### Stop time options:

- **Start time:** This field reports the selected simulation start time, as configured in the *Calculation of Initial Conditions* command. This value cannot be directly changed in this command dialog.
- **Absolute:** Set here the absolute stop time of this simulation. This value must always be greater than the *Start time*.

---

**Note:** The absolute stop time value can be entered in seconds (by default) or using a derived unit by clicking in the text field corresponding to the unit and then changing the exponent setting corresponding to this unit.

---

**Stop time** options when the simulation has already advanced past the initially configured *Start time* (i.e. at least one simulation run has been previously executed and the calculation has not been reset):

- **Current time:** This field reports the currently achieved simulation time, corresponding to the last time instant of the most recently executed *Run Simulation* command. This value cannot be directly changed in this command dialog.
- **Absolute:** Set here the absolute stop time of this simulation, referenced to the simulation *Start time*. This value must be greater than the *Current time*. Upon changing this value, the *Relative time* is automatically updated.
- **Relative:** Set here the relative stop time of this simulation, referenced to the simulation's *Current time*. This value must be greater than 0. Upon changing this value, the *Absolute time* is automatically updated.

#### Display in output window options:

- **Display result variables:** Set this checkbox in order to display to the output window all the recorded result variables at each simulation time step. Note, for improving the simulation performance, make sure to not use this option.
- **Display internal DSL events:** Set this checkbox in order to display to the output window all internally executed DSL events. Note, for improving the simulation performance, make sure to not use this option.
- **Display automatic step size adaptation events:** Set this checkbox in order to display to the output window all internally applied step size adaptation events. Note, for improving the simulation performance, make sure to not use this option.

#### Behaviour in case of Internal Dynamic Model warnings:

- **Ignore and continue simulation:** If this option is used, any issued Internal Dynamic Model warnings will be ignored and the simulation will be continued.
- **Display in output window and continue simulation:** If this option is used, any issued Internal Dynamic Model warnings will be reported to the output window. Furthermore, the simulation will be continued.
- **Display in output window and stop simulation:** If this option is used, any issued Internal Dynamic Model warnings will be reported to the output window. Furthermore, the simulation will be stopped immediately.

**Initial conditions:** This field reports the relevant *Calculation of Initial Conditions* command which is applied for this simulation run.

## 29.5 Results Objects

During an EMT or RMS simulation, a large number of available variables are changing over time. Time dependent variables which can be monitored within a dynamic simulation fall into the following categories:

- Currents, Voltages and Powers
- Bus Results
- Signals
  - Input signals - subcategory “IN” (e.g. for a synchronous generator, variable “ve” - Excitation voltage)
  - Output signals - subcategory “OUT” (e.g. for a synchronous generator, variable “ie” - Excitation current)
  - State variables - subcategory “STATE” (e.g. for a synchronous generator, variable “phi” - Rotor Angle)
  - Derivatives of state variables - subcategory “d/dt” and denoted by “state:dt”, where “state” is the name of a state variable (e.g. for a synchronous generator, variable “phi:dt” - derivative of Rotor Angle, i.e. rotor speed)
- Calculation Parameter
- Element Parameter
- Reference Parameter

To reduce the available data and to narrow down the number of variables to those necessary for the analysis of each particular case, a selection of these signals for later use has to be defined.

Therefore, one or more results objects containing the result variables can be configured. The simulation function needs the reference to a results object to store the results.

The command dialogs for calculation functions, that produce signals, have results object references, as depicted in Figure 29.5.1 for the *Initial Conditions* dialog.

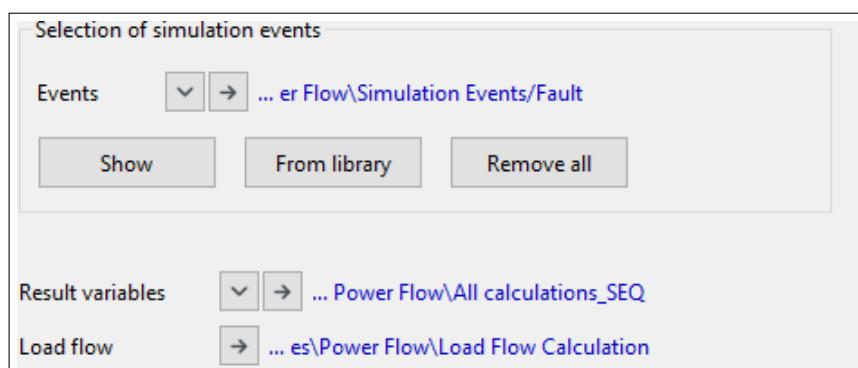


Figure 29.5.1: Results Object Reference

Such a results object reference refers to the currently used results object. The downward arrow button ( ) is used to select or reset the reference, or to edit the contents of the referenced results object.

The right-arrow button ( ) is used to edit the results object itself. When the button is pressed, the *ElmRes* dialog is opened. It is possible to access to the list of variables stored inside the results object by pressing the **Variables** button.

An easier way to edit or inspect the results object is to press the *Edit Result Variables* icon on the main toolbar ( ), or to select the *Calculation → Simulation RMS/EMT → Result Variables...* option from the main menu. This will enable the user to edit the contents of the currently selected results object in the *Initial Conditions* command dialog. Results objects (*ElmRes*) are covered in detail in Chapter 19 (Reporting and Visualising Results).

### 29.5.1 Monitoring variables of an element

It is many times useful to record variables of an element for various purposes e.g. plotting a variable versus simulation time. To record one or several element variables, these need to be explicitly defined as *monitored variables* belonging to a *Results* object (*ElmRes*).

**To add a variable to a *Results* object**, do the following:

- Make sure that any previously executed calculation (if any) is reset . Otherwise, it is not possible to add new variables to a *Results* object. One exception: it is possible to add monitored variables if the previously executed calculation is the *Calculation of Initial Conditions*.
- If the element is reached using the *Data Manager* or the *Network Model Manager*:
  - navigate and identify the element whose variables are to be added.
  - Right-click the element and choose *Results Variables → Simulation RMS/EMT...*
- If the element is reachable directly from the single line diagram:
  - Right-click the element and choose *Results Variables → Simulation RMS/EMT...*
- If there exist multiple *Results* objects in the active study case, then a selection window is shown. Choose the *Results* object in which the variables will be added.
- The contents of the *Results* object is shown in a new window. The element of interest is shown in this list as well. Double click the element's icon.
- The *Variable Selection* window is shown. Select the corresponding variables.

---

**Note:** The Variable Selection object is described in Section 19.3: [Variable Selection](#).

---

#### 29.5.1.1 Displaying current values of scalar/array variables in the *Data Manager*/*Network Model Manager*

The current value of a scalar/array variable can be shown using the *Network Model Manager*, as follows:

- Make sure that either the *Calculation of Initial Conditions* or a *Run Simulation* has been previously executed (such that results are available)
- Open the *Network Model Manager* and go to the element category of interest (e.g. *Modelica Models*).
- Select the element of interest (e.g. a specific *Modelica Model*) and click the *Variable Selection* ( ) button
- The *Variable Selection* window is shown. Select the corresponding variables.

---

**Note:** The Variable Selection object is described in Section 19.3: [Variable Selection](#).

---

- After adding the variables of interest, click **OK**.

- New columns are added to the *Flexible Data* page, one for each added variable.
- If the variable is a scalar, then its current value is displayed in the relevant field (column of the variable, row of the element).
- If the variable is an array, then the value of the first cell in the array is displayed in the relevant field along with trailing dots. Double click the relevant field in order to show the current values of the entire array variable.

A similar procedure can be applied while using the *Data Manager*.

Further description of displaying multidimensional attributes is provided in Chapter 11, Section 11.2.10.

### 29.5.2 Saving Results from Previous Simulations

The variables to be monitored are stored (by default) in the results object called “All calculations” within the Study Case. The results of the variables in the current simulation are stored in this file also. If the results of two different simulations are to be displayed, e.g. in one plot, there is the possibility to save the results object of a previous simulation simply by copying the results object “All calculations” and renaming it.

This can be done easily in the Data Manager, by copying and pasting the results object “All calculations” into the active Study Case folder. A second results object will be created with the name “All calculations(1)”.

In the next simulation, the default results object “All calculations” will be overwritten with the new results, but the copied results will not be modified and can be displayed together with the new simulation results in one plot. For further information see Chapter 19: Reporting and Visualising Results, Section 19.8 (Plots).

## 29.6 Simulation Events

This section provides a general description of Events, as they apply to time-domain simulations. See Chapter 13: Study Cases, Section [Events](#) for a detailed description of the event types.

There are several ways to access events objects:

- From the *Data Manager*, in the *Simulation Events/Faults* object stored within the *Study Case*.
- From the *Calculation of Initial Conditions* command, using the *Show* button on the *Selection of simulation events*.
- From the Simulation RMS/EMT toolbar by pressing the *Edit Simulation Events*  icon. A list of the currently defined events will be displayed, including the set simulation time, when the event will occur, and the related object. Note that a duration for a 3-Phase Short-circuit is not specified, rather, another event is created to clear the fault.

When creating a new event, use the *New Object*  icon in the toolbar. The event type can be chosen from the displayed list, as shown in Figure 29.6.1. Some events can also be modified during a simulation by stopping the calculation, editing the events and continuing the simulation.

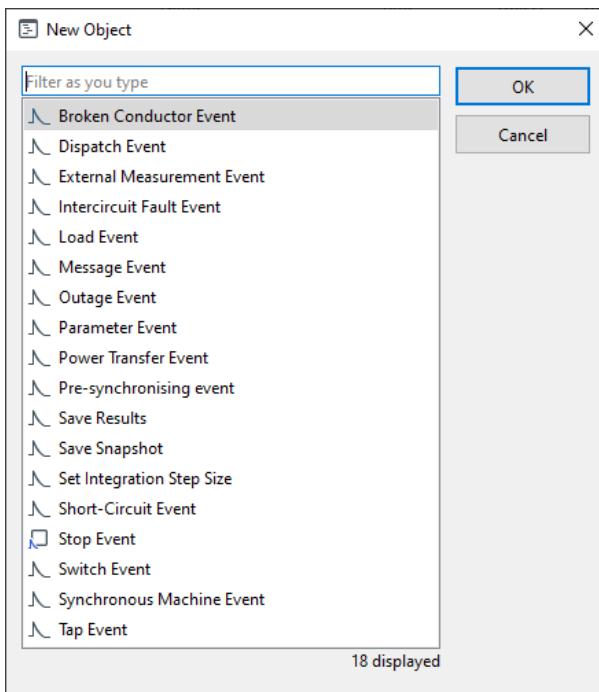


Figure 29.6.1: Defining a New Simulation Event

An alternative way of defining events is as follows: upon calculation of the initial conditions (⌚), or when the simulation is already running, double-click on the relevant cubicles to create switch events. Additionally, the user can right-click on an element and then select an element-related event such as *Simulation Events* → *Switch Event...* or *Simulation Events* → *Load Event...*.

During a simulation all previous events (i.e. events which have already occurred), are shown in grey and can no longer be edited or changed. When the simulation is finished or is stopped manually, the events which are still to come in the simulation can be altered and new events can be created.

The following events, where the fault location is defined in the event, cannot be defined when the simulation is running:

- Broken Conductor Event
- Intercircuit Fault Event
- Short-Circuit Event

The reason for this is that when calculating the initial conditions, a dummy terminal is internally defined on the line to represent the fault location.

**Note:** At the end of a simulation the event list shows all events, in grey. They can no longer be modified for this simulation, because the simulation could be restarted from this point on. To change the events for a new simulation one must first initialise the calculation again (⌚), so that the simulation time is reset to the beginning.

#### EMT Simulation: Various options of triggering breaker close events

The breaker switching event (*EvtSwitch*) enables a circuit breaker to be closed:

- depending on execution time
- at voltage zero crossing

- on minimum absolute voltage across contacts
- on maximum absolute voltage across contacts
- on maximum positive voltage across contacts
- on maximum negative voltage across contacts

Further customisation options are available when closing a breaker at voltage zero crossing or on minimum absolute voltage across contacts, as shown in Figure 29.6.2.

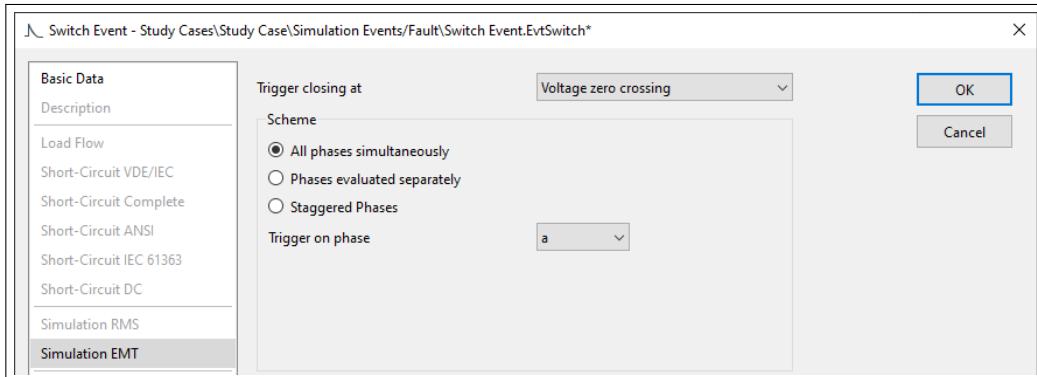


Figure 29.6.2: Additional options for breaker close events

### Load Event used on a selection of load elements

It is possible to apply a single load event (*EvtLod*) to multiple load elements by selecting the option *Event for → Multiple Loads*.

### Parameter Event used on a selection of elements

It is possible to apply a single parameter event (*EvtParam*) to multiple elements by referencing a selection object (*SetSelect*).

## 29.7 Executing the Simulation

Upon successful calculation of the initial conditions (i.e. execution of *ComInc* ), the simulation can be started via *Start Simulation* on the Simulation RMS/EMT toolbar. In the case that the calculation of the initial conditions has not yet been performed, *Start Simulation* will automatically execute the initialisation.

The *Run Simulation (ComSim)* dialog has a link to the used initial conditions command and additional options for the displayed messages can be set in the *Display in output window* and *Internal Dynamic Model warnings* fields.

---

**Note:** The *Start Simulation* button is deactivated and can't be used if the previous simulation lead to an error. In this case, either the command *Reset Calculation* or *Calculation of Initial Conditions* must be executed first.

---

The simulation is performed for the time interval between the start time defined in the initial conditions command (*ComInc*), and the stop time, which is specified in the simulation (*ComSim*) dialog. After a

simulation has finished, it may be continued by pressing *Start Simulation*  again and entering a new stop time. In this case, the stop time can also be entered relative to the current simulation time.

A running simulation can be interrupted by one of the following options. *Stop Simulation*  pauses the simulation after the current simulation time step. In this stage, some types of events can be created, results may be viewed or the simulation end time can be adapted. The simulation can then be continued by pressing *Start Simulation*  again. Pausing and continuing the simulation may be done as often as required. In contrast, using *Break*  on the main toolbar stops the simulation immediately. If this occurs during an iteration or while convergence issues occur, an error is thrown. In any case, the current results of the simulation are saved, however, the simulation can not be continued afterwards if the break leads to an error.

During each simulation, a progress bar will be displayed at the bottom of the *PowerFactory* graphical user interface. The progress bar will disappear once the simulation is finished or interrupted.

## 29.8 Creating Simulation Plots

After a simulation is executed, the results can be visualised in a plot. Pressing the *Create Simulation Plot* icon  from the Simulations RMS/EMT toolbar opens the Insert Plot dialog where the typical plots used for RMS/EMT simulation are displayed.

Further information about plot types and handling is available in Chapter 19: Reporting and Visualising Results, subSection 19.8 (Plots).

## 29.9 Simulation Scan

Simulation scan modules can be defined and accessed as explained in Section 29.3.4 or by using the *Edit Simulation Scan* icon  from the Simulations RMS/EMT toolbar. The available simulation scan objects are described in the following subsections.

---

**Note:** After creating a new simulation scan module, the initial conditions should be executed with the option *Active* on the *Simulation Scan* page enabled. Otherwise the modules will not be considered.

---

### 29.9.1 Fault Ride Through Scan Module

The Fault Ride Through (FRT) scan module (*ScnFrt*) monitors variables of various elements (e.g. the voltage on a busbar) in the power system and continuously verifies these signals for validity against a user defined FRT characteristic. Validity is defined by comparing the waveform of the measured signal with the FRT characteristic and requiring the measured signal not to be below the characteristic longer than a specific user defined period (which can be set to zero). The triggering of the comparison (i.e. the scan start time) is done by comparing the measured signal with a constant threshold parameter. The first time that the signal is lower than the threshold, the FRT scan module is triggered and the two waveforms (measured signal and FRT characteristic) start to be compared one against the other. Should the FRT scan module detect a FRT characteristic violation of the measured signal(s) then, depending on user choice, the simulation can be stopped, or a message can be printed to the output window (without simulation interruption).

The Fault Ride Through Scan Module options are described in the following sections:

### 29.9.1.1 Fault Ride Through Scan - Basic Options

#### Ignored

If this flag is set, then this scan module is not active.

#### Scan Location

- *Whole System*: applies the FRT scan module to all calculation relevant elements of the class defined in *Class name*.
- *User defined*: applies the FRT scan module to a set of elements or to a single element. The set or the single element can be selected using the associated drop down menu ( *Select...* )
- *Class name*: elements matching this class name will be added to the monitoring list. If a single element is chosen in the *User defined* field then the class is automatically selected.

#### Variable

- *Voltage*: the voltage of the scan location will be monitored
- *Other*: a variable different than voltage should be monitored. The name of the variable has to be entered and it must correspond to a valid variable name of the element class being monitored.

#### Limit Curve Type

- *Lower limit*: the curve will be set as the lower limit of the monitored variable
- *Upper limit*: the curve will be set as the upper limit of the monitored variable

#### Action

- *Display message*: if the limit is violated, a corresponding message is displayed in the output window.
- *Stop simulation*: the dynamic simulation is stopped if the limit is violated and a corresponding message is displayed in the output window.
- *Stop with error*: this option is only visible if the *Stop simulation* option is selected. A simulation error is thrown in addition to stopping the simulation.
- *Trigger*: if the limit is violated, a trigger is activated and if an action is triggered, a corresponding message is displayed in the output window. The trigger can be selected or edited with the and buttons respectively.

#### Display Messages

- *All*: all the messages, including activation of the scan are printed in the output window.
- *Violations only*: only the violations of the FRT curve are printed.

### 29.9.1.2 Fault Ride Through Scan - Limit Curve

#### FRT Table

This table allows the user to insert the FRT characteristic using a set of points in format (time,value). Each row corresponds to a single point on the FRT time characteristic. The characteristic is shown under the table.

### Activation threshold

This value sets the fault detection threshold. During a simulation (and provided that the FRT scan is “active”), the first time that any of the monitored variables goes below the *activation threshold*, the FRT scan module will issue a fault start event and report it to the output window.

### Duration

Once a fault is detected, the value of the monitored signal(s) is compared to the FRT characteristic. If the *Duration* is zero and once any of the monitored values is below the characteristic, the case is treated as a FRT characteristic violation. If the *Duration* is not zero, the specific monitored value (that has been triggering the fault detection) must remain below the characteristic for *Duration* seconds in order to treat the case as a violation. The purpose of using a *Duration* parameter is to avoid spurious actions.

### Multiple fault detection

By selecting this option, an additional threshold can be defined. This threshold will “reset” the fault detections if the curve stays over the *Reignition threshold*, for a time defined in the *Minimum duration* field.

#### 29.9.1.3 Fault Ride Through Scan - Unbalanced Options

##### Unbalanced network representation

The voltage to be measured is defined in this field, the following representations are available:

- PH-PH
- PH-N
- PH-E
- Pos. Seq.
- PH. Tech.

*Phase Technology* will use the representation defined in the busbar element. If a representation different than the one recorded in the results file is selected, a warning will be issued and the recorded one will be automatically selected.

##### Unbalanced fault detection

- *Any phase crosses threshold*: all phases are monitored using the same curve, as soon as one phase crosses the threshold.
- *All phases cross threshold*: all phases are monitored using the same curve, as soon as all phases cross the threshold.
- *Each phase crosses threshold independently*: each individual phase is monitored independently as soon as the corresponding phase crosses threshold, i.e. three different fault detection messages will be displayed (if the *Display Messages* option in the *Basic Option* page is set to *All*)

##### Unbalanced violation

- *Any phase violates limit*: the violation is detected as soon as the first phase violates the limit.
- *All phases violate limit*: violation message is triggered when all the phases violate the limit.
- *Each phase violates limit independently*: each phase is evaluated independently, when the *Action* is set to *Display Message*, messages for each phase will be printed in the output window.

### 29.9.1.4 Fault Ride Through Scan - Time Settings

#### Activation time

- The activation time, defined in *Hours*, *Minutes* and *Seconds*, represents the time in simulation when the FRT Scan module should start the monitoring of variables. The values of monitored variables prior to the activation time are ignored and not considered in the assessment.
- *Time step*: defines the sample time with which the scan is performed.

## 29.9.2 Frequency Scan Module

The frequency scan module (*ScnFreq*) monitors bus frequency.

### 29.9.2.1 Frequency Scan Module - Basic Options

#### Ignored

If this flag is set, then this scan module is not active.

#### Scan Location

- *All busbars*: applies the frequency scan module to all the busbars (*ElmTerm*) relevant for calculation.
- *User defined*: only a set of elements or a single element is monitored.

#### Scan measurement

This module can be defined to measure either the *Frequency* or the *Frequency gradient*. The *Nominal frequency* can be set to any frequency value.

#### Detection of multiple violations

If the simulation is not stopped, all the frequency violations will be detected, i.e. the curve is continuously scanned.

#### Action

- *Display message*: if the limit is violated, a corresponding message is displayed in the output window.
- *Stop simulation*: the dynamic simulation is stopped if the limit is violated and a corresponding message is displayed in the output window.
- *Stop with error*: this option is only visible if the *Stop simulation* option is selected. A simulation error is thrown in addition to stopping the simulation.
- *Trigger*: if the limit is violated, a trigger is activated and if an action is triggered a corresponding message is displayed in the output window. The trigger can be selected or edited with the and buttons respectively.

### 29.9.2.2 Frequency Scan Module - Limits

The limits for the frequency violation detection are set in this page. The *Maximum* and *Minimum* limits should be defined.

### 29.9.2.3 Frequency Scan Module - Time Settings

#### Activation time

- The activation time, defined in *Hours*, *Minutes* and *Seconds*, represents the time in simulation when the frequency scan module should start the monitoring of variables. The values of monitored variables prior to the activation time are ignored and not considered in the assessment.
- *Time step*: defines the sample time with which the scan is performed.

## 29.9.3 Loss of Synchronism Scan Module

The loss of synchronism scan module (*ScnSync*) monitors the internal generator model signal “Out of Step” of synchronous machines (*ElmSym*). The out of step detection can be defined on the *Reference System* tab of the *Basic Options* page on the Initial Conditions command.

#### Scan location

Defines whether the scan has to be applied to the whole system, a selection of objects or a single object.

#### Activation Time

Defines the time at which monitoring should start (till end) in *Hours*, *Minutes* and *Seconds*. Set the *Time step* to define the intervals at which the scan should be performed.

#### Action

- *Display message*: if the limit is violated, a corresponding message is displayed in the output window.
- *Stop simulation*: the dynamic simulation is stopped if the out of step signal is detected. The corresponding message is displayed in the output window.
- *Stop with error*: this option is only visible if the *Stop simulation* option is selected. A simulation error is thrown in addition to stopping the simulation.
- *Trip generator*: if the signal *out of step* is detected, the generator that has lost synchronism is disconnected, a corresponding message is displayed in the output window. After this event is triggered, it is still possible to reconnect the generator.
- *Trigger*: a trigger is activated and if an action is triggered a corresponding message is displayed in the output window. The trigger can be selected or edited with the  $\downarrow$  and  $\rightarrow$  buttons respectively.
- *Trip generator and set out of service*: if the signal *out of step* is detected, the generator that has lost synchronism is disconnected and set to out of service along with all associated controls. Associated controls are all DSL elements (and only those ones) which are referenced into a slot of a composite model whose *Main slot* is occupied by the target generator. A corresponding message is displayed in the output window. After execution of this event, it is not possible to reconnect the generator or any of the associated controls.

### 29.9.4 Synchronous Machine Speed Scan Module

The synchronous machine speed scan module (*ScnSpeed*) monitors speed of synchronous machines. If a limit is violated, it displays message, stops the simulation, trips the generators that have violated the limit or activates a predefined trigger.

#### Scan location

Define whether the scan has to be applied to the whole system, a selection of objects or a single object.

#### Speed settings

*Maximum* and *Minimum* speed limits are defined.

#### Activation time

- The activation time, defined in *Hours*, *Minutes* and *Seconds*, represents the time in simulation when the variable scan module should start the monitoring of variables. The values of monitored variables prior to the activation time are ignored and not considered in the assessment.
- *Time step*: defines the sample time with which the scan is performed.

#### Action

- *Display message*: if the limits are violated, a corresponding message is displayed in the output window.
- *Stop simulation*: the dynamic simulation is stopped if the scan module triggers. A corresponding message is displayed in the output window.
- *Stop with error*: this option is only visible if the *Stop simulation* option is selected. A simulation error is thrown in addition to stopping the simulation.
- *Trip synchronous machine*: if the limits are violated, the corresponding generator is disconnected and a message is displayed in the output window. After this event is triggered, it is still possible to reconnect the generator.
- *Trigger*: a trigger can be assigned using this option. If the limits are violated, the trigger is automatically activated. A corresponding message is displayed in the output window.
- *Trip synchronous machine and set out of service*: if the limits are violated, the corresponding generator is disconnected and set to out of service. This option is particularly useful if disabling associated DSL models is needed. A corresponding message is displayed in the output window. After triggering this event, there is no possibility of reconnecting the generator.

### 29.9.5 Variable Scan Module

The variable scan module (*ScnVar*) monitors a selected element variable and triggers a display message or stops the simulation if a defined limit is violated.

#### Scan Location

- *Whole System*: scans all calculation relevant elements of the class defined in *Class name*.
- *User defined*: applies the scan module to a set of elements or to a single element. The set or the single element can be selected using the associated drop down menu ( *Select...* )
- *Class name*: elements matching this class name will be added to the monitoring list. If a single element is chosen in the *User defined* field then the class is automatically selected.

### Detection of multiple violations

If the simulation is not stopped, all limit violations of the variable are detected, i.e. the value is continuously scanned and the selected action started in case of new violations.

### Variable

The name of the variable has to be entered and it must correspond to a valid variable name of the element class being monitored.

### Settings

Define the variables *Maximum limit* and *Minimum limit*.

### Activation Time.

Define the time at which monitoring should start (till end) in *Hours*, *Minutes* and *Seconds*. Set the *Time step* to define the intervals at which the scan should be performed.

### Action

- *Display message*: if the limit is violated, a corresponding message is displayed in the output window.
- *Stop simulation*: the dynamic simulation is stopped if the limit is violated and a corresponding message is displayed in the output window.
- *Stop with error*: this option is only visible if the *Stop simulation* option is selected. A simulation error is thrown in addition to stopping the simulation.
- *Trigger*: if the limit is violated, a trigger is activated and if an action is triggered a corresponding message is displayed in the output window. The trigger can be selected or edited with the and buttons respectively.

## 29.9.6 Voltage Scan Module

The voltage scan module (*ScnVolt*) monitors bus voltage and triggers a display message or stops the simulation if a defined limit is violated.

### 29.9.6.1 Voltage Scan Module - Basic Options

#### Ignored

If this flag is set, then this scan module is not active.

#### Scan Location

- *All busbars*: applies the voltage scan module to all the busbars (*ElmTerm*) relevant for calculation.
- *User defined*: only a set of busbar or a single one is monitored.

#### Scan measurement

This module can be defined to measure either the *Voltage* or the *Voltage gradient* during a specified *Time window*.

### Detection of multiple violations

If the simulation is not stopped, all the voltage violations will be detected, i.e. the curve is continuously scanned.

### Unbalanced network representation

The voltage to be measured is defined in this field, the following representations are available:

- PH-PH
- PH-N
- PH-E
- Pos. Seq.
- PH. Tech.

*Phase Technology* will use the representation defined in the busbar element. If a representation different than the one recorded in the results file is selected, a warning will be issued and the recorded one will be automatically selected.

### Action

- *Display message*: if the limit is violated, a corresponding message is displayed in the output window.
- *Stop simulation*: the dynamic simulation is stopped if the limit is violated and a corresponding message is displayed in the output window.
- *Stop with error*: this option is only visible if the *Stop simulation* option is selected. A simulation error is thrown in addition to stopping the simulation.
- *Trigger*: if the limit is violated, a trigger is activated and if an action is triggered a corresponding message is displayed in the output window. The trigger can be selected or edited with the and buttons respectively.

#### 29.9.6.2 Voltage Scan Module - Limits

##### Voltage

Define the *Maximum limit* and *Duration (max. limit)*, and the *Minimum limit* and *Duration (min. limit)*.

##### Recovery thresholds

- *No thresholds*: no recovery thresholds are considered.
- *Individual thresholds for violation and recovery detection*: different thresholds are defined for the maximum and minimum voltage limits. The parameter *Duration below/above the threshold* defines the minimum time the value has to be within the limits to be recognised as a recovery.
- *Common threshold for violation and recovery detection*: one unique value is defined as recovery threshold.

### 29.9.6.3 Voltage Scan Module - Time Settings

#### Activation time

- The activation time, defined in *Hours*, *Minutes* and *Seconds*, represents the time in simulation when the voltage scan module should start the monitoring of variables. The values of monitored variables prior to the activation time are ignored and not considered in the assessment.
- *Time step*: defines the sample time with which the scan is performed.

## 29.10 Save/Load Snapshot

When carrying out simulations in *PowerFactory*, it is possible to save the current simulation state for later use. This can greatly increase productivity, especially if the simulation state has been obtained as a result of a time-consuming simulation.

The *Save Snapshot* as well as the *Load Snapshot* actions can be performed easily from the *Simulation RMS/EMT* toolbar. The snapshot can either be saved in memory, in which case the information is lost once *PowerFactory* is closed, or in an external file, such that the simulation state can always be recovered at a later date.

---

**Note:** The *Load/Save Snapshot* supports all built-in and DSL based models. Furthermore, all DLL models generated via the *C-Interface* (as documented in Section 30.14.3) are also supported.

---

### 29.10.1 Saving a snapshot

Saving a snapshot can be done either manually or automatically.

In order to manually save a snapshot, do the following:

- Prepare a dynamic simulation and make sure that it initialises without errors;
- Run the simulation up to the time point which should be saved in the snapshot;
- Click on  *Save Snapshot* icon in the *RMS/EMT Simulation* toolbar. Choose between the following options:
  - *In non-persistent memory slot* - the snapshot is saved in a temporary location in memory; as such, the snapshot is lost upon closing the *PowerFactory* application;
  - *In file* - the snapshot is permanently saved on the disk (or similar permanent storage device); in this situation make sure to provide a valid directory path and corresponding file name for the snapshot within the subsequent *File* field; using this option will enable users to load the snapshot even after restarting *PowerFactory*.

In order to automatically save a snapshot, do the following:

- Define *Save Snapshot* events (*EvtSave*) in the simulation queue and set the event trigger time to the simulation time instant at which the simulation will be saved. *Save Snapshot* events are detailed in Section 13.9.10;
- Make sure that you have accordingly configured the options in the *Calculation of Initial Conditions (ComInc)* dialog, page *Snapshot*. Further information on these options are available in Section 29.3.7;
- Prepare a dynamic simulation and make sure that it initialises without errors;

- Run the simulation. The *Save Snapshot* event is triggered automatically and a snapshot is consequently saved.

### 29.10.2 Loading a snapshot

Loading a snapshot can be done by executing the following steps:

- Make sure that you have activated in *PowerFactory* the same project with the same study case that has been used for saving the snapshot.
- Execute the *Calculation of initial conditions* and make sure that it has run successfully.
- Click on the  *Load Snapshot* icon in the *RMS/EMT Simulation* toolbar and point to the snapshot that is to be loaded. The snapshot can be available either in the memory (if the  *Save Snapshot* command has been previously executed and saved a snapshot in the non-persistent memory) or on the disk. After selecting the snapshot click on Execute. The simulation state will be loaded and the simulation initialised at snapshot time.
- Continue running the simulation from that point.
- All queued events that are defined after the saved execution time will be applied during the simulation. All events defined for execution prior to the snapshot time will be disregarded.
- Result files are written with the starting time point equal to the snapshot time.

---

**Note:** Snapshots are only compatible within the same major *PowerFactory* version. As such, snapshots cannot be used for a different major version than the one they have been exported from. It should also be noted, that their functionality can be compromised if network elements are changed (e.g. modification of line parameters, DSL or DLL equations)

---

## 29.11 Single/Multiple Domain Co-simulation

This section provides guidance on the basic usage and configuration of the *Single/Multiple domain co-simulation* tool. The terms and definitions used in co-simulation are detailed in Section 29.11.5. An overview of the co-simulation concepts is given in Section 29.11.1. For information on how to configure the co-simulation tool, Section 29.11.2 is to be followed. Information regarding the execution and plotting of co-simulation results is provided in Section 29.11.3.

### 29.11.1 Overview of the Single/Multiple Domain Co-simulation

*PowerFactory* provides an integrated co-simulation tool for executing time domain analysis of power systems. This *Single/Multiple domain co-simulation* tool is a fully integrated *PowerFactory* solution which does not require any third party interfaces or software. From this perspective, the *Single/Multiple domain co-simulation* is fundamentally different from the external solver based co-simulation tool described in Section 29.12, where *PowerFactory* is intended to run in a co-simulation environment composed of one or multiple third-party solvers.

Based on the standard dynamic simulation tool (presented in Section 29.2), *PowerFactory* supports the following *co-simulation domains*:

- RMS balanced;
- RMS unbalanced;

- EMT.

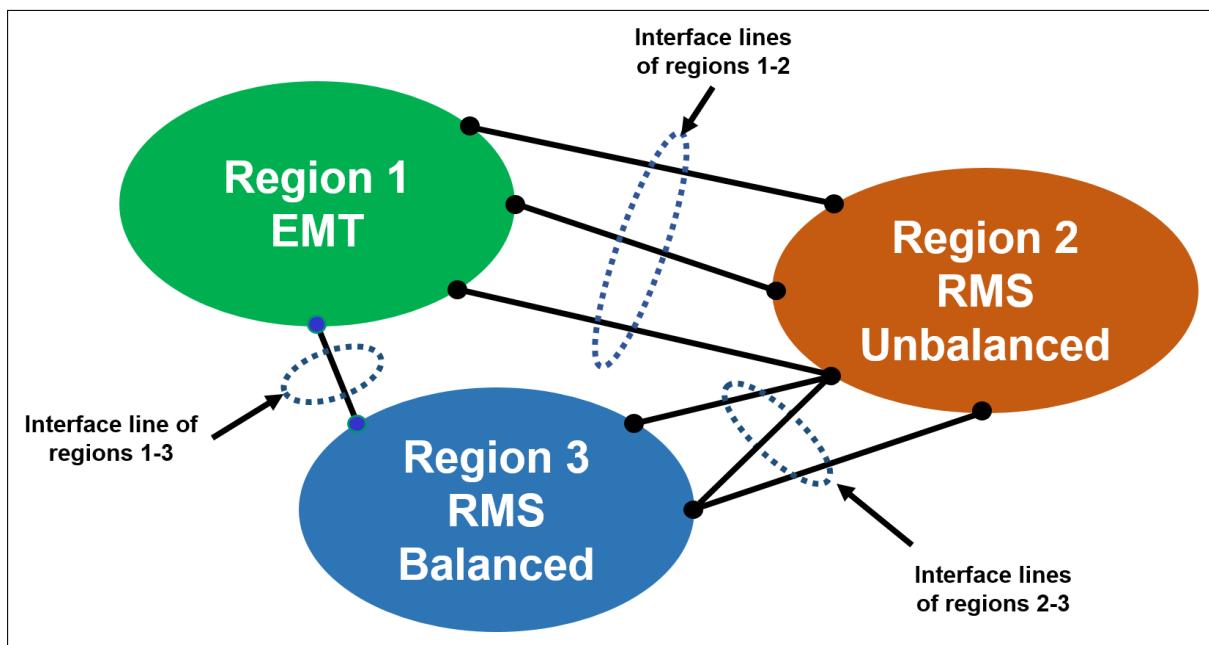


Figure 29.11.1: Example of co-simulation *domains* and corresponding *interface elements*

In terms of possible *domain* configurations, the *Single/Multiple domain co-simulation* supports the following choices:

- for single *domain* co-simulation:
  - RMS balanced - RMS balanced **or**
  - RMS unbalanced - RMS unbalanced **or**
  - EMT - EMT;
- for multiple *domain* co-simulation:
  - RMS balanced - RMS unbalanced - EMT (as illustrated in Figure 29.11.1) **or**
  - RMS balanced - RMS unbalanced **or**
  - RMS balanced - EMT **or**
  - RMS unbalanced - EMT.

**Note:** None of the configurations above imposes a specific restriction on the number of definable *regions* e.g. a RMS balanced - RMS balanced single *domain* co-simulation may have more than two co-simulation *regions*. Each *region* is assigned to a single logical processor core; the number of logical cores in a machine defines the maximum number of *regions* that can be used in a co-simulation. As such, in terms of performance and flexibility, machines with a high number of cores are better suited for execution of co-simulation runs.

Two co-simulation *Methods* are available, as follows:

- *Implicit (exact approach)*: - this *method* provides accurate results compared with a normal sequential run. The method requires that the *boundary* between *regions* is defined on long enough AC power lines such that it can account for the associated transmission delays. The AC lines must be manually set to use the “Distributed parameter” model (available in the *Basic Data* page of the line element dialog).

**Note:** For implicit co-simulation, boundary branches using a “Distributed parameter” model with *loss model* and set to *R/2 at both ends* are supported.

- *Explicit (approximate approach)*: this *method* delivers reliable results (approximate, but not exact in comparison with a sequential simulation) while providing further flexibility in the choice of the *boundary* definition. The co-simulation interface can be defined on any set of AC lines without a particular requirement on line length and the associated transmission delays.

### 29.11.2 Configuring the *Internal Co-simulation*

The calculation function for setting up a co-simulation is called from the *Initial conditions for co-simulation* (*ComCosim*  ) command.

This command must be successfully executed before the actual co-simulation can be run (done via the *Run simulation* (*ComSim*  ) command). Practically, the co-simulation procedure is as follows:

- Set up boundary objects  for each *region*;
- Assign *boundaries* and other configuration settings within the *Initial conditions for co-simulation* command *ComCosim*  (refer to Section 29.11.4 for more information);
- Configure simulation options for each *region* based on calculation of initial conditions (*ComInc*  ) object(s);
- Define variable results for elements of interest;
- Execute the *Initial conditions for co-simulation* (*ComCosim*  ).
- Execute the *Run simulation* (*ComSim*  ) command for the intended simulation duration.

An example of a *Single domain* co-simulation configuration is shown in Figure 29.11.2.

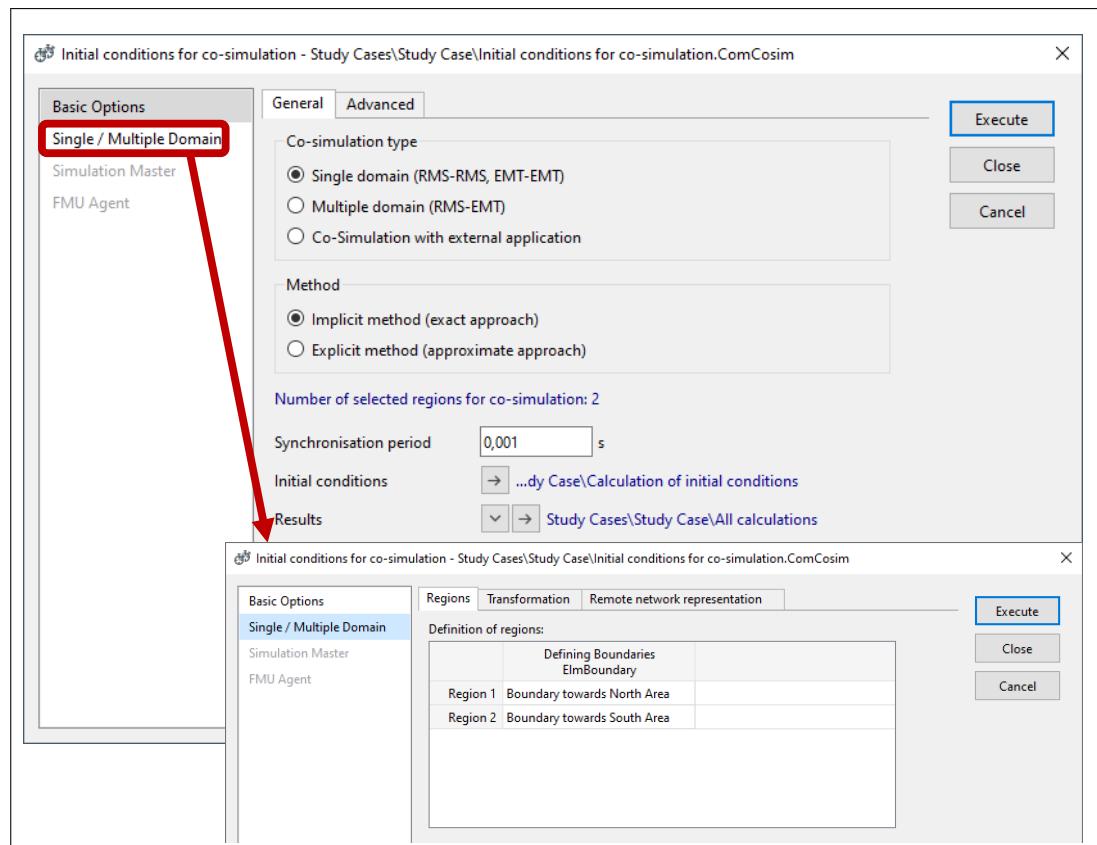


Figure 29.11.2: Co-simulation configuration options (example)

Further guiding notes:

- **Boundary objects (*ElmBoundary*)** are used to define the co-simulated *regions* and thereby the co-simulation interface. The interior region of each *boundary* needs to contain all elements belonging to the specific co-simulation *region*. Boundaries need to overlap either on the line element(s) which represent the co-simulation border (so-called “Branch-based splitting”) or on the common bus elements at the co-simulation border (so-called “Node based splitting”). An example for a *Branch-based splitting* is shown in Figure 29.11.3, whereas a *Node-based splitting* is shown in Figure 29.11.4. Both figures show a simple two area power system interconnected via two AC lines, a candidate for a co-simulation split on two *regions*. For the *Branch-based splitting*, the two AC lines need to be contained within both boundary objects (as overlapping elements), in order to correctly identify the co-simulation interface. In this example, two boundary objects need to be defined; then the *boundaries* are assigned to individual *regions* within the *Initial conditions for co-simulation*. Alternatively (Figure 29.11.4) a *Node-based splitting* can be used, where two common AC buses need to be contained within both boundary objects (as overlapping elements), in order to correctly identify the co-simulation interface.
- **Configuration of simulation options** is done via the *Initial conditions for co-simulation* by editing the default *Calculation of Initial Conditions (ComInc)* command (in the case of *Single domain* co-simulation) or by assigning individual *Calculation of Initial Conditions (ComInc)* command objects for each *region* (in the case of *Multiple domain* co-simulation). A user-defined common synchronisation period can be configured in the *ComCosim* command which will define the rate by which *regions* are synchronised across the co-simulation interface(s). Based on the two area example above, a multiple *domain* co-simulation could be configured as shown in Figure 29.11.2
- **Definition of results** follows the standard procedure adopted for a typical sequential dynamic simulation. *PowerFactory* will automatically manage and organise the storage location of results for a given element, depending on the *region* to which it belongs. At the end of a co-simulation run, there will be one results file (*ElmRes*) for each *region*.

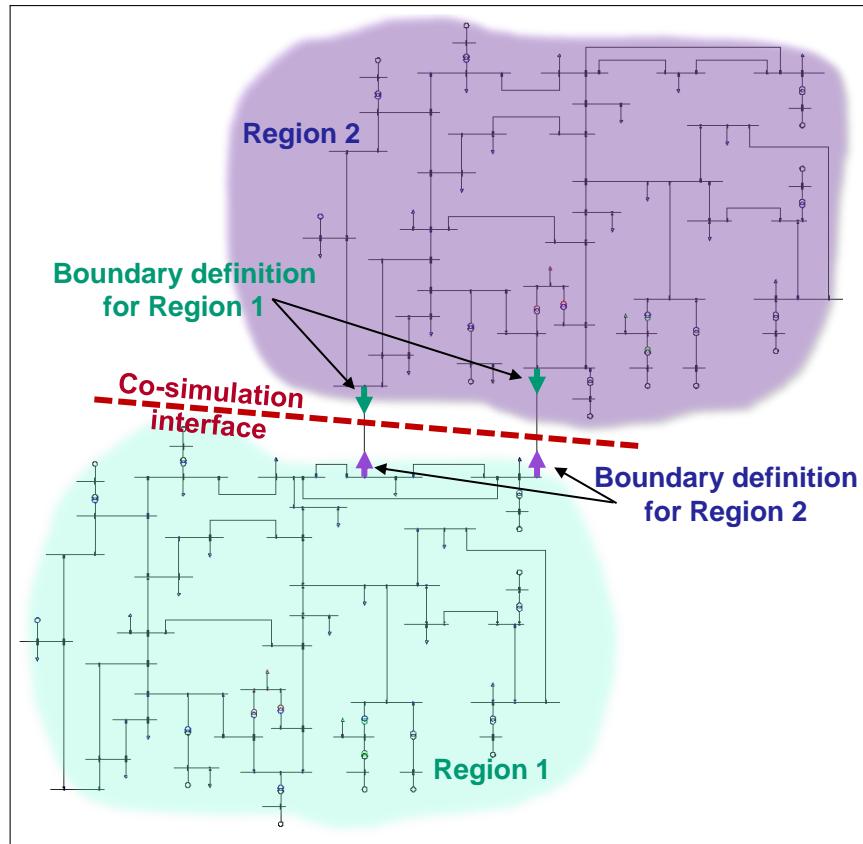


Figure 29.11.3: Boundary definition using *Branch-based splitting* of a two area power system

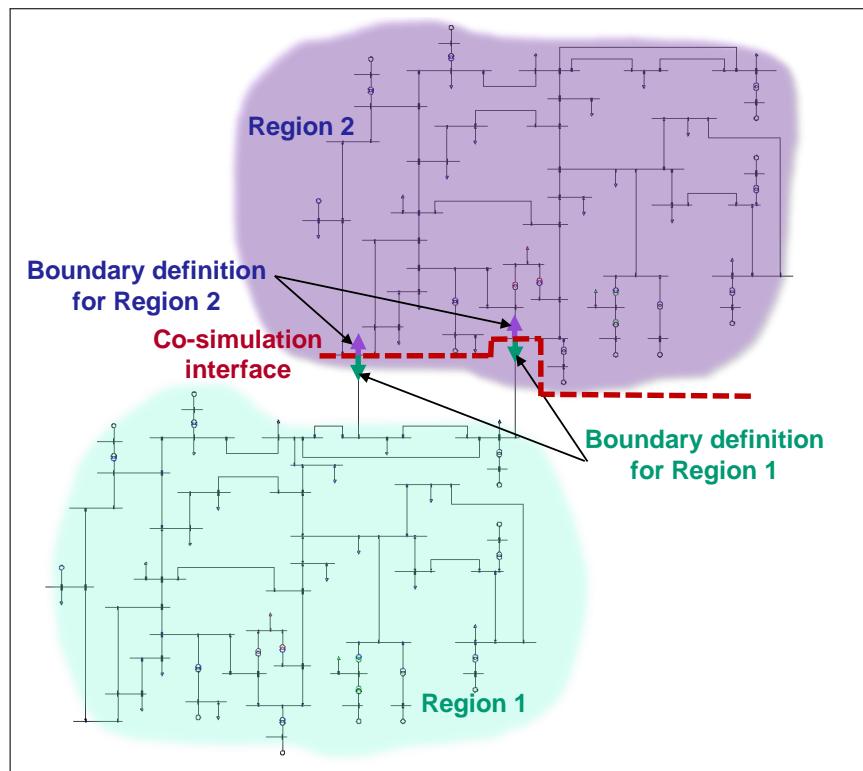


Figure 29.11.4: Boundary definition using *Node-based splitting* of a two area power system

The region splitting methods are supported as shown in Table 29.11.1.

<b>Splitting Method</b>	Explicit co-simulation		Implicit co-simulation
	Current/Voltage source equivalent	Multi-port Thevenin equivalent	
Node-based		X	
Branch-based	X	X	X

Table 29.11.1: Supported splitting method for Co-Simulation

### 29.11.3 Performing a Co-simulation and Retrieving Results

To perform a co-simulation, execute the following steps:

- Make sure that the co-simulation has been successfully configured (refer to Section 29.11.2).
- Open the *Initial conditions for co-simulation (ComCosim)* command dialog:
  - Go to the *Basic Options* page and select either the *Single domain* or the *Multiple domain Co-simulation* type.
  - Configure the *Synchronisation period* appropriately. The synchronisation period will influence highly the simulation results and the overall performance. The co-simulation *domain* choice and the dynamic properties of the power system have a direct influence on the optimum choice of the *Synchronisation period*.
  - Configure other settings according to specific requirements (for a detailed description of options refer to Section 29.11.4)
  - Execute the *Initial conditions for co-simulation (ComCosim)* command. Prior to running the co-simulation, the *Initial conditions for co-simulation* is to be executed. *PowerFactory* automatically starts the required parallel processes and initialises the simulation in each *region*.
- Execute the simulation using the *Run Simulation (ComSim)* command. After the simulation has been executed, results can be plotted, exported or post-processed as for any other dynamic simulation.
- At the end of a co-simulation, the *Results (ElmRes)* object linked in the *Basic Options* page of the *Initial conditions for co-simulation (ComCosim)* has the following structure:
  - *Region 1 - Results* object(*ElmRes*): contains the results of all elements within *Region 1*.
  - *Region 2 - Results* object(*ElmRes*): contains the results of all elements within *Region 2*.
  - ⋮
  - *Region n - Results* object(*ElmRes*): contains the results of all elements within *Region n*.
- Visualise and analyse results. The simulation results of elements within each *region* can be shown in a plot, as follows:
  - Click the *Insert Plot* icon , choose a *Curve plot* and then *OK*.
  - Under column *Element* select the element of interest. Click *OK*.
  - Under column *Variable* select the variable of interest. Click *OK*.

**Note:** *PowerFactory* automatically identifies the region in which the element is placed and automatically retrieves the variable of interest from the corresponding *Results* object. The name of the variable contains a suffix denoting the region from which the values are taken e.g. “@Region1”

### 29.11.4 The “*Initial conditions for co-simulation*” (*ComCosim*) Command

The *single/multiple domain* co-simulation makes use of the *Initial conditions for co-simulation* command . The command options are described below.

**Basic Options page - General tab**

*Co-simulation type:*

- *Single domain (RMS-RMS,EMT-EMT)*: choice of a single *domain* co-simulation (with internal solver)
- *Multiple domain (RMS-EMT)*: choice of a multiple *domain* co-simulation (with internal solver)
- *Co-simulation with external application*: this option configures *PowerFactory* to execute a co-simulation based on an external solver (more information on this type of co-simulation can be found in Section 29.12).

*Method:*

- *Implicit method (exact approach)*: choice to apply an implicit co-simulation method;
- *Explicit method (approximate approach)*: choice to apply an explicit co-simulation method;

*Number of selected regions for Co-Simulation*: this field provides overview information on the number of *regions* that have been already configured for use in a co-simulation; the *regions* are defined in the *Single/Multiple Domain* page.

*Synchronisation period*: via this field, a user-defined fixed synchronisation period between the co-simulated *regions* can be provided;

*Start time*: field available for *Multiple domain (RMS-EMT)* option only; the simulation in each *region* is initialised at the *Start time*;

*Initial settings*: field available for *Single domain (RMS-RMS,EMT-EMT)* option only; a reference link to a single *Calculation of initial conditions (ComInc)* command which is to be used for all *regions* of the co-simulation;

*Results*: A selection link to the results file (*ElmRes*) which aggregates all co-simulation results of elements is provided.

**Basic Options page - Advanced tab**

*Use Load Flow of complete network for initialisation of regions*: Set this flag in order to perform an initial Load Flow calculation of the complete network for initialisation of regions. If set, then a reference to the Load Flow calculation command is shown below the flag.

*Options for calculations on grid before reduction*

*Do not adapt Load Flow Calculation Method to Simulation method of regions*: Set this flag in order to not adapt Load Flow Calculation Method to Simulation method of regions when computing calculations before regions are separated.

---

**Note:** When this option is not set, *PowerFactory* may apply modifications to the *Load Flow Calculation* command executed before splitting the network into regions. Namely, the balanced/unbalanced load flow method may be changed depending on the choice of the dynamic simulation method of regions. Without this option, some cases may lead to a failed load flow calculation. Use this option to avoid such situations.

---

*Skip Calculation of Initial Conditions*: Set this flag in order to avoid the execution of the Calculation of Initial Conditions of the whole network (i.e. before regions are separated).

---

**Note:** When this option is not set, *PowerFactory* will always perform a *Calculation of Initial Conditions* command before regions are separated. In some cases, a combined simulation of the entire network is nevertheless not possible (e.g. case of an RMS/EMT co-simulation with local models designed only for either the RMS or the EMT simulation domain, but not for

---

both), and therefore, a combined dynamic simulation start will lead to errors when executing the co-simulation. To avoid such situations, make sure to set this option.

---

*Update plots during simulation:* Depending on this flag, the time domain plots which show co-simulation results can be updated either during or at the end of the co-simulation run. For improved performance, the flag should be un-checked.

---

**Note:** Within co-simulation not all load flow options are supported. Within the load flow command dialog, page “Active Power Control”, the option *Active Power Control → as Dispatched* is supported. Furthermore, the following active power *Balancing* options are supported for the explicit method:

- *by reference machine*,
- *by load at reference bus*,
- *by static generator at reference bus*.

All *Balancing* options are supported by the implicit co-simulation method. In case of a *Distributed slack* option, then the *Reference Busbar* location (if selected) is required to be within the co-simulated *region*.

---

*Display messages of all regions:* During the simulation run, if this flag is checked, then the output window will include all generated messages of the simulated *regions*;

*Reuse parallel processes:* Check this flag if already-created parallel processes are to be re-used;

*Parallel computation settings:* A link to the parallel computation settings is provided. Further information on the *Parallel computation settings* and the *Parallel computing manager* is available in Section 6.6.1.

*Indication of used/available cores for co-simulation:* Information is provided on the number of used and available processor cores for a co-simulation run.

#### **Single/Multiple Domain page - Regions tab**

*Definition of regions:* the *regions* to be used in the co-simulation can be defined within this tabular list. In the case of a *Multiple domain co-simulation*, a specific *Calculation of initial conditions (ComInc)* command can be assigned for each *region* in the *Settings* column. The domain used for each *region* is shown in the *Domain* column, and can be changed within the corresponding *Calculation of initial conditions (ComInc)* command (along with all other settings).

---

**Note:** If no *Boundary* element is existing in the *Boundaries* folder (*Network Model → Network Data → Boundaries*), then the boundary selection field is inactive. *Boundary* objects must be defined first.

---

*Create remaining region:* If the regions in the table do not completely cover the grid, this is indicated by a message and a button is provided to create a region covering the remainder of the grid. If the co-simulation method is set to *Explicit* and the *Remote network representation* is set to *Multi-port Thevenin equivalent*, then this region is created to make a node-based split, otherwise it is created to make a branch-based split.

*Network partitioning:* If the *Regions* table is empty or the network is completely covered, then the automatic partitioning function is available via the *Network partitioning* pane. The **Partition Network** button can be used to automatically split the currently active power system into a customisable number of regions. All power system changes applied by the network partitioning tool are recorded in a dedicated Network Variation.

The following customisation options can be set prior to partitioning the network:

- Number of regions: Set a positive integer number greater than or equal to 2. The set value defines the number of regions that shall be created by the partitioning algorithm.
- Only distributed parameter lines admissible as boundary branches: Check this option in order for *PowerFactory* to identify as eligible boundary branch components only branches configured as a *Distributed Parameter* line model. Otherwise, all branches are considered in the partitioning algorithm.
- Automatic adjustment of boundary branches to “Distributed Parameter”: Check this option in order for *PowerFactory* to automatically adjust branches which are initially configured as *Lumped Parameter* line model, to *Distributed Parameter* line model. The option creates, if necessary, new types sharing the same data as the original types. Whenever the existing data is incompatible with a *Distributed Parameter* line model, the parameters are appropriately adjusted using default values (e.g. zero values for line capacitance are changed to non-zero default values). Whenever the case, boundary branches are referenced to the newly created branch types. Changes to the boundary branch are recorded within the dedicated Network Variation.

*Elements excluded from being boundary branches:* A user defined set can be selected. Select here those elements which are not intended to be used as boundary branches for co-simulation.

*Event locations:* Check this option in order to automatically add to the fore-mentioned set those elements involved in pre-defined simulation events (which are appearing in the Simulation Events queue).

#### **Single/Multiple Domain page - Transformation tab**

*Transformation settings:* In the case of a *Multiple domain* co-simulation, this tab allows the conversion of electrical quantities between the RMS and EMT domains to be appropriately configured. The following options are available:

- *Same transformation for all:* Same transformation settings are used for all *boundary branches* involved in the co-simulation; at the bottom of the dialog, the relevant *RMS-EMT transformation settings* (*SetTransf*) object is linked.
- *Different transformation per region:* For each set of two *neighbouring regions* one transformation is assigned. All possible combinations are defined in the associated list.

---

**Note:** Whenever available, the tabular list can be automatically populated by clicking on the button Reset and load all. For each possible combination in the list, the *Settings* column can be configured with *RMS-EMT transformation settings* (*SetTransf*) objects. The current list is completely removed by clicking on the button Clean up. Additionally, the Clean up command will also delete the unused transformation settings (*SetTransf*) objects, usually located inside the *Initial conditions for co-simulation* command object.

---

#### **Single/Multiple Domain page - Remote Network Representation tab**

The options in this tab are active only for *Explicit method (approximate approach)* method of co-simulation.

*Equivalent for remote networks* pane has the following options:

- *Current/Voltage sources at boundary buses:* The applied network reduction method uses simple current/voltage sources at the boundary buses for the representation of remote networks.
- *Multi-port Thevenin equivalent:* The network reduction computes a multi-port equivalent of the entire remote network of each *region*. The method has the following options:
  - *Conversion to stable EMT equivalent where required:* The reduced network equivalent may sometimes be numerically unstable due to the existence of negative resistances. If this flag is active, then any negative resistance is set to zero.

- *Boundaries without slack pane:*

- \* *Alternative slack locations:* select the bus/set of buses where a reference machine (slack) element is to be inserted in case one or a set of regions have missing reference machine elements. A *region*(boundary) without reference machine is a *region* that contains only passive elements, loads and no elements with grid-forming capability (e.g. synchronous machines, static generators with reference machine capability). If no bus or set of buses is assigned, then the program automatically assigns reference machine elements (a voltage source element), in regions with missing reference machines at the *Boundary/Edge nodes*.

**Simulation Master page:** this page is only relevant for the co-simulation with external application. Refer to Section [29.12](#) for further information.

**FMU Agent page:** this page is only relevant for the co-simulation with external application. Refer to Section [29.12](#) for further information.

### 29.11.5 Terms and Definitions for Co-simulation

The most important terms needed for co-simulation are detailed below.

- **Co-Simulation:** Simulation in parallel of different sub-systems which form a coupled problem.
- **Co-Simulation domain:** The analysis domain of the coupled problem. Typically, for power system analysis tools the **co-simulation domains** will be one of the three available time-domain simulation methods, as described in Section [29.2](#).
- **Simulation unit:** A **Simulation unit** consists of one time domain simulator and its associated power system model in a combined *Co-Simulation*.
- **Region:** Within the *Single/Multiple domain co-simulation*, multiple *simulation units* can be included, while the associated power systems model of each unit is referred to as a **Region**. This is due to the fact that in the case of power system analysis applications the linking between the *simulation units* is typically done at the border between physically separated regions of the power system, as shown in Figure [29.11.1](#). Co-simulation domains are assigned individually for each *region*.
- **Neighbouring regions:** A *region i* is a **neighbouring region** with respect to *region j* if at least one electrical connection path exists between the two *regions* that does not cross through another *region*.
- **Interface elements:** The intersection between the elements of a *region* and one of its *neighbouring regions* defines the *interface elements* of the two *regions*. Interface elements can be AC lines (branch-based splitting) or terminals (node-based splitting)

**Note:** For a valid co-simulation, the following is required:

- the *interface elements* must consist only of AC line elements.
- it is not allowed to include dynamic models whose controls span across co-simulation *regions*.

- **Boundary:** Boundary objects (*ElmBoundary*) are *PowerFactory* grouping objects which specify a topological cut through the power network (refer to Section [15.4](#) for more information). Boundary objects are used within co-simulation to define the *interface elements* of *neighbouring regions*. The co-simulation **Boundary** is defined by a boundary object which cuts the power network via all AC line elements which separate one *region* from all other *neighbouring regions*. As a consequence, the number of co-simulation *regions* equals the number of co-simulation *boundaries*. The elements of a co-simulation *region* are the interior region elements of the associated co-simulation boundary object.
- **Boundary branch:** An *interface element* which is an ac line (*ElmLne*). Practically, **boundary branches** are those ac line elements that are common between a *region* and one *neighbouring region*.

## 29.12 Co-simulation with External Application

This section provides guidance on the configuration and use of the *Co-simulation with External Application*. This section is organised as follows:

- An overview of the *Co-simulation with External Application* is provided in Section 29.12.1.
- The configuration of the co-simulation tool is provided in Section 29.12.2.
- The execution and plotting of co-simulation results are detailed in subSection 29.12.3.
- The *Preparation as FMU Agent (ComCosimsetup)* command is described in subSection 29.12.4
- The FMU Agent I/O (*ElmAgentio*) and FMU Agent (*ElmAgent*) objects are described in subSection 29.12.5
- The *Initial conditions for co-simulation (ComCosim)* command is detailed in subSection 29.12.6
- The terms and definitions are given in subSection 29.12.7.

### 29.12.1 Overview of the Co-simulation with External Application

*PowerFactory* supports the [Functional Mock-up Interface \(FMI\)](#) as a standard framework for the *Co-simulation* of power systems. As such, *PowerFactory* can be used to coordinate several [Functional Mock-up Units \(FMUs\)](#) in a combined power systems *Co-simulation*, as shown in Figure 29.12.1. The components supported by *PowerFactory*  and, potentially, by a generic third-party software  are correspondingly marked.

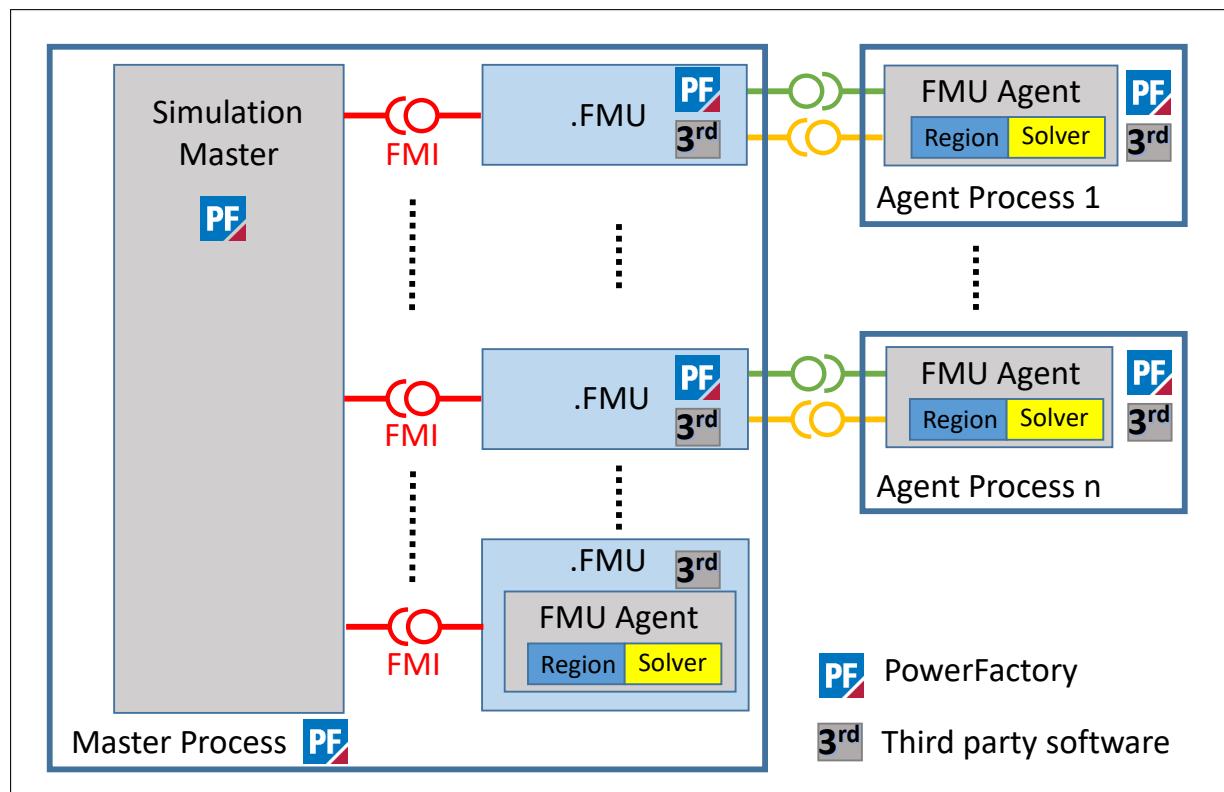


Figure 29.12.1: Architecture of the co-simulation with external application

The *Simulation Master* interacts with other applications (i.e. tool-coupling) or with other standalone *FMUs* via the *.FMU* files. Furthermore, *PowerFactory* can generate *FMUs* utilised by an *FMI*-compliant

simulation tool for the purpose of *Co-simulation*. As such, *PowerFactory* creates multiple *FMUs*, one for each defined [region](#) of an existing power system model. A *PowerFactory*-created *.FMU* can only be linked with a *PowerFactory*-hosted *FMU Agent*, and viceversa.

The *FMI* link is a compiled library access connection (the *.FMU* files must be located on the *Simulation Master's* local machine), compliant with the *FMI* standard and having a specific set of input/outputs signals being exchanged (refer to Section [29.12.2.6](#)). The methods by which the required set of exchanged signals are computed in an *FMU* is not specified, each *FMU Agent* being responsible for proper signal evaluation.

The green and orange links are not specified by the *FMI* standard. In the case of *PowerFactory FMU Agents*, the link is based on the TCP/IP communication protocol. Other *FMU Agents* can implement customised interfaces, independently of the *Co-simulation for external application* requirements (in other words, the way in which the data are transferred across these interfaces is not relevant to the *Co-simulation for external application* function). There are no other particular requirements between the [Master Process](#) and the [Agent Processes](#), except that the required interface links (, and ) must be functional and available.

The [Co-simulation with External Application](#) is designed for (but not restricted to) two main use cases, as follows:

- *Co-simulation* between *PowerFactory* and a third party software: power system models developed in *PowerFactory* and others in a third party tool may be included in a combined *Co-simulation*.
- Distributed *Co-simulation* between multiple *PowerFactory* applications: multiple *PowerFactory* instances located in different locations can carry out a *Co-simulation* via a TCP/IP based network.

The [FMI](#) standard specifies two main components:

- *FMI for Model Exchange* (*FMU* model import/export)
- *FMI for Co-simulation* (*FMU* model import/export)

The *FMI for Co-simulation* component is relevant for the *Co-simulation with external application*.

Each *FMU agent* is responsible for supporting the designated [Co-simulation domains](#) and with exchanging the relevant signals. The framework for *Co-simulation* supports the following domains:

- RMS balanced
- RMS unbalanced
- EMT

Two *PowerFactory* commands are used for *co-simulation with an external application*:

- *Preparation as FMU agent* (*ComCosimsetup* ) - detailed in Section [29.12.4](#)
- *Initial conditions for co-simulation command* (*ComCosim* ) - detailed in Section [29.12.6](#)

## 29.12.2 Configuring the *Co-simulation with external application*

The *Co-simulation* is configured according to the following procedure:

- **Definition of regions** - Section [29.12.2.1](#)
- **Checking requirements for power system elements relevant to FMI signal exchange** - Section [29.12.2.2](#)
- **Generation of *.FMU* files** - Section [29.12.2.3](#)
- **Configuring the *PowerFactory FMU Agents*** - Section [29.12.2.4](#)

- **Configuring the *Simulation Master*** - Section 29.12.2.5
- **Execution of co-simulation** A co-simulation can be executed whenever all *FMUs* (and their corresponding *.FMU* files) are generated and the co-simulation pre-requisites are fulfilled. Refer to Section 29.12.3 for information on how to execute a co-simulation using external application.

### 29.12.2.1 Definition of *regions*

In *PowerFactory* this is done by creating a *Boundary* object (*ElmBoundary*  ) for each *region*. Refer to Section 15.4 for more information on boundaries, their definition and usage.

---

**Note:** Boundaries can be defined using the *Boundary Definition Tool* or directly from the single line diagram by selecting one (or multiple) branch elements, right-clicking and selecting *Network Groupings* → *Boundary* → *New...*. For co-simulation, the definition from the single line diagram is typically used.

---

The *boundary*'s interior region defines each *Co-simulation region*. The [internal region elements](#) and [external region elements](#) can be easily identified. The way the boundary must be defined (i.e. the *network splitting method*) is dependent on the choice of the remote network equivalent. The available *network splitting methods* are shown in Table 29.12.1 as a function of the remote network equivalent. The two splitting methods are graphically demonstrated using a two area example in Figure 29.12.2 (Branch-based) and Figure 29.12.3 (Node-based). In the case of *Branch-based* splitting it is apparent that two boundaries must be defined at the connecting cubicles of a branch element (an AC line element) and have the *Orientation* towards the branch. In the case of *Node-based* splitting it is apparent that two boundaries must be defined at the same connecting cubicle of a branch element and have opposite orientations (one pointing towards the branch, another one pointing towards the busbar). The same procedure can be applied to any other more complex network configurations.

Splitting Method	Remote Network Equivalent	
	Current/Voltage source equivalent	Multi-port Thevenin equivalent
Node-based		X
Branch-based	X	X

Table 29.12.1: Supported splitting method for Co-Simulation

---

**Note:** It is important to underline that in the case of *Branch-based* splitting the *interface elements* must be a set of AC lines. The types of supported lines are marked with a "Y" in Tables 29.12.2 and 29.12.3 for each co-simulation variant.

---

<b>Co-simulation method</b>	<b>Remote Network Representation</b>	<b>ElmLne</b>	<b>ElmTow</b>	<b>ElmCabsys</b>	
		AC lumped parameters			
implicit co-simulation (internal / external)		N	N	N	
explicit co-simulation (internal / external)	Current/Voltage sources at boundary buses	Y	Y	N	
	Multi-port Thevenin equivalent	Y	Y	Y	
		AC distributed parameters: - constant parameters - R/2 at both ends			
implicit co-simulation (internal / external)		Y	Y	N	
explicit co-simulation (internal / external)	Current/Voltage sources at boundary buses	Y	Y	N	
	Multi-port Thevenin equivalent	Y	Y	Y	
		AC distributed parameters: - constant parameters - R/4 at both ends; R/2 in the middle			
implicit co-simulation (internal / external)		N	N	N	
explicit co-simulation (internal / external)	Current/Voltage sources at boundary buses	Y	Y	N	
	Multi-port Thevenin equivalent	Y	Y	Y	

Table 29.12.2: Supported AC line models for branch-based splitting co-simulation (1)

<b>Co-simulation method</b>	<b>Remote Network Representation</b>	<b>ElmLne</b>	<b>ElmTow</b>	<b>ElmCabsys</b>	
		AC system, FDP (modal domain)			
implicit co-simulation (internal / external)		N	N	N	
explicit co-simulation (internal / external)	Current/Voltage sources at boundary buses	Y	Y	N	
	Multi-port Thevenin equivalent	Y	Y	Y	
		AC system, FDP (phase domain with ULM or FDM)			
implicit co-simulation (internal / external)		N	N	N	
explicit co-simulation (internal / external)	Current/Voltage sources at boundary buses	Y	Y	N	
	Multi-port Thevenin equivalent	Y	Y	Y	
		DC system, all variants			
implicit co-simulation (internal / external)		N	N	N	
explicit co-simulation (internal / external)	Current/Voltage sources at boundary buses	N	N	N	
	Multi-port Thevenin equivalent	Y	Y	Y	

Table 29.12.3: Supported AC line models for branch-based splitting co-simulation (2)

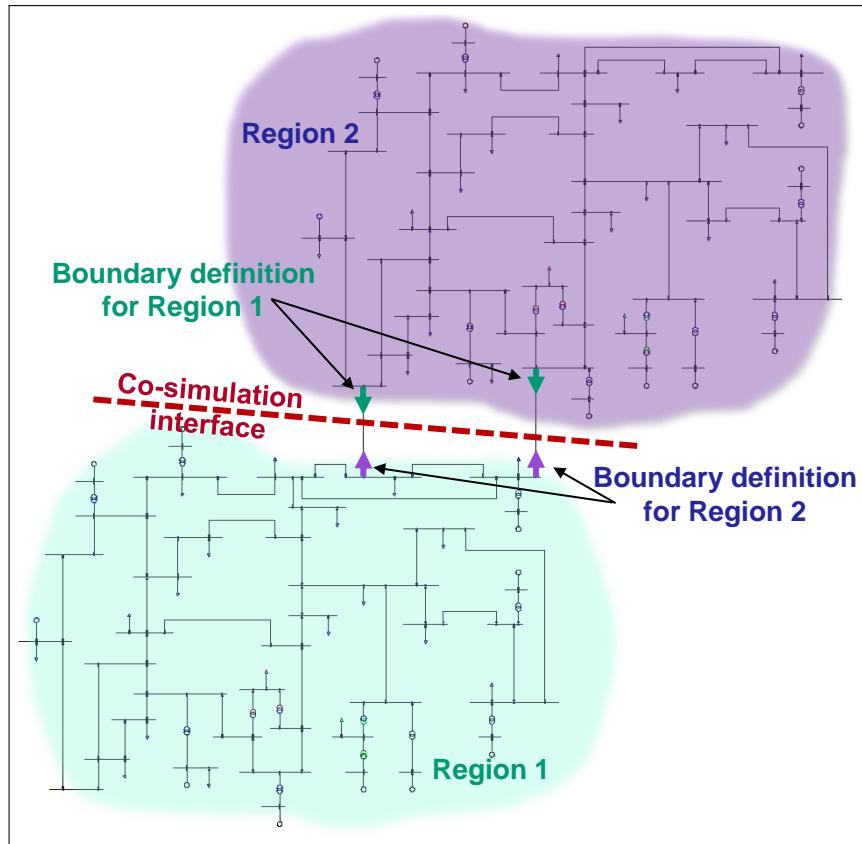


Figure 29.12.2: Boundary definition for *Branch-based* network splitting method of a two area power system

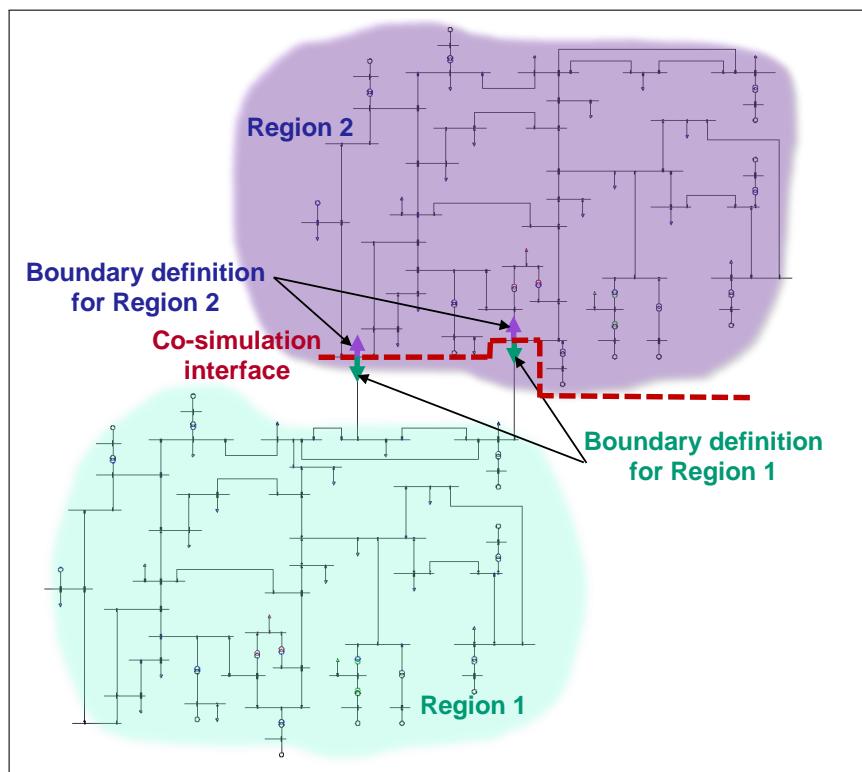


Figure 29.12.3: Boundary definition for *Node-based* network splitting method of a two area power system

### 29.12.2.2 Checking requirements for power system elements relevant to FMI signal exchange

The relevant elements for FMI signal exchange are those elements directly connected to [edge nodes](#). Each of these elements must have a power system wide unique [foreign key](#). If the *foreign key* is empty for a relevant element, then an automatically generated *foreign key* will be assigned.

---

**Note:** Each *PowerFactory* object has a dedicated [foreign key](#) attribute (“*for\_name*”), found in the *Description* page of the element’s Edit dialog. Third party software must also define a compliant *foreign\_key* for each relevant element.

---

### 29.12.2.3 Generation of .FMU files

Based on the defined *regions* and *foreign keys* corresponding *.FMUs* are generated. An *.FMU* describes all input/output signal connections of the contained *region* to any other *.FMUs* (describing the *neighbouring regions*). In *PowerFactory* this process involves the configuration and execution of the *Preparation as FMU agent* (*ComCosimsetup* ):

---

**Note:** Refer to Section [29.12.4](#) for more information on configuration settings of the command.

- The previously defined boundary objects are selected.
- The remote network equivalent options (*Current-/Voltage-source/Multi-port Thevenin*) are selected depending on the previously made choice for the remote network representation when boundaries were defined.
- The command allows the (a) network reduction of the remote network for each region, (b) the creation of relevant *FMU* objects (internal *PowerFactory* objects) and (c) the creation of relevant *.FMUs* (external file).
- If the defined *regions* do not completely cover the power system, an additional *.FMU* is created for the *rest of the system*. For correct operation, make sure that *foreign keys* are defined for the elements connected to the *edge nodes* of the *rest of the system* as well.
- The set of signals being exchanged must be identified at this stage. Comprehensive information regarding the selection of signals, their purpose and the used format is provided in Section [29.12.2.6](#).

### 29.12.2.4 Configuring the *PowerFactory* FMU Agents

For each *PowerFactory*-generated *.FMU* one *PowerFactory* instance must be started and a *FMU Agent* must be configured using the *Initial conditions for co-simulation* command (*ComCosim* ). The settings used in this case are:

- *Basic Options page* → *General tab* → *Co-simulation type*: Co-simulation with external application
- *Basic Options page* → *General tab* → *Task*: Run as FMU Agent
- *FMU Agent page* configuration. Refer to Section [29.12.6](#) for more information on configuration settings.

#### Preparation of the FMU Agents

On the *PowerFactory* side, a project can be prepared for co-simulation with an external application by using the command *Preparation as FMU agent* (*ComCosimsetup* ). An example of a configuration setup based on a two area power system (refer to Figure [29.11.4](#)) is shown in Figure [29.12.4](#).

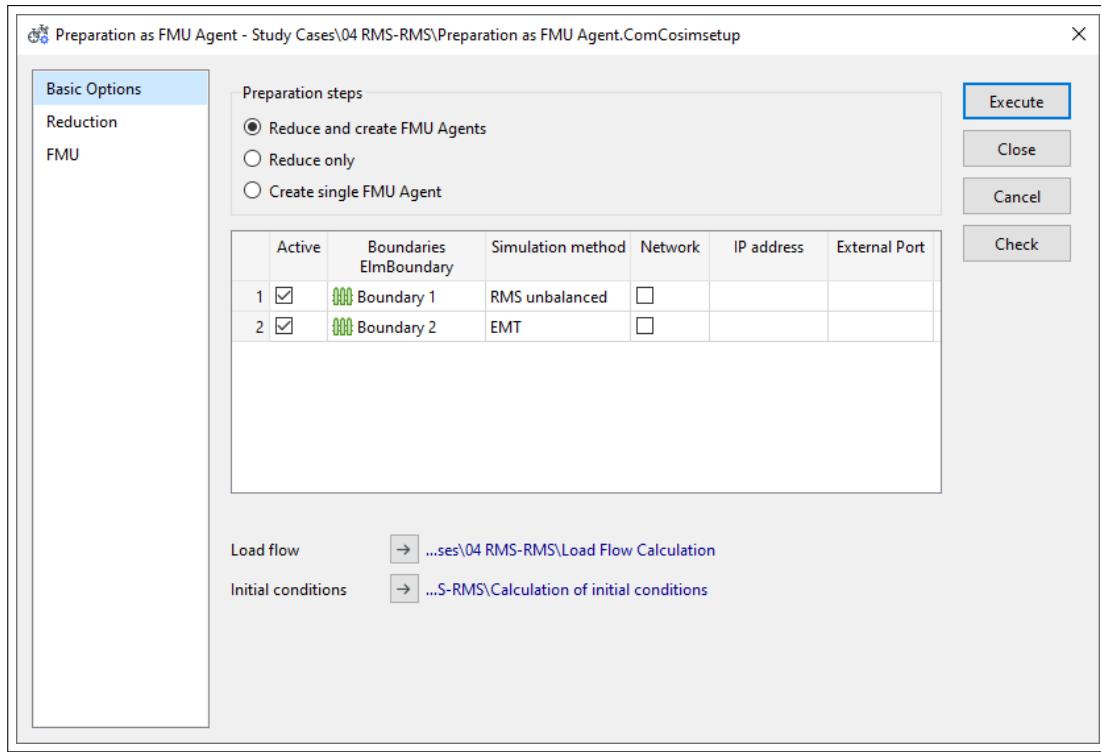


Figure 29.12.4: Co-simulation with external application - configuration options (example)

The command is able to perform the following tasks:

- Reduce and create FMU Agents: performs network reduction of the power system *regions* which are to be simulated externally and creates corresponding *FMU Agent I/O* objects for each active list entry.

**Note:** The *FMU Agent I/O* object (*ElmAgentio*) is described in detail in Section 29.12.5

- Reduce only: performs network reduction of the power system *regions* which are to be simulated externally. It also creates new *FMU Agent I/O* objects but no *.FMU* files. This option is useful when the FMUs are already created but the network variations or the network reduction must be re-created/re-executed e.g. some subsequent topological modifications have been made to the system.
- Create single FMU Agent: creates one single *.FMU* file based on an already existing *FMU Agent I/O* object (*ElmAgentio*).

**Note:** The FMU Agents can be created based on an explicit (default) or an implicit co-simulation algorithm. The implicit co-simulation method is intended only for a co-simulation in which all FMI Agents are *PowerFactory* instances.

### 29.12.2.5 Configuring the *Simulation Master*

Based on one *PowerFactory* instance, a *Simulation Master* is configured using the *Initial conditions for co-simulation* command (*ComCosim* ). The settings used in this case are:

- *Basic Options page* → *General tab* → *Co-simulation type*: Co-simulation with external application
- *Basic Options page* → *General tab* → *Task*: Run as simulation master

- *Simulation Master* page configuration. Refer to Section 29.12.6 for more information on configuration settings.

### Preparation of the Simulation Master

A separate *PowerFactory* instance is required to be available. The *PowerFactory* project which has been used to generate the FMU Agents must be activated on this new *PowerFactory* instance. None of the FMU Agent specific study cases are to be used (i.e. the automatically generated cases upon execution of the *ComCosimsetup* command). Instead, the original study case should be used. To configure the Simulation Master, do the following:

- With the original study case active, open the *Initial conditions for co-simulation* command (*ComCosim*).
- Set the command according to the previously described settings, namely: *Co-simulation type*: Co-simulation with external application; *Task*: Run as simulation master.
- Go to the Simulation Master page. The previously generated FMU Agents are automatically loaded. Each row in the tabular list contains a reference to an *FMU Agent* object (*ElmAgent*). The *FMU Agent* object defines the file path to the previously generated *.FMU* file existing on the Simulation Master machine. Also, the *FMU Agent* can be set to *out of service*.
- The *Add Agent* command can be used to add another FMU Agent. This is useful whenever a third party or externally generated *.FMU* file is available, yet it was not previously generated by the *Preparation of FMU Agent* command.

#### 29.12.2.6 Exchanged *.FMU* signals with the *Simulation Master*

Depending on the boundary definition and of the configured *Co-simulation domain*, the *FMUs* must exchange a specific set of signals at each synchronisation time with any neighbouring *FMU*. When mixing *Co-simulation domains* (RMS balanced, RMS unbalanced, EMT), *PowerFactory* exchanges the “simplest” of the two signal types, i.e. balanced preferred over unbalanced and RMS preferred over EMT type signals, as shown in Figure 29.12.5.

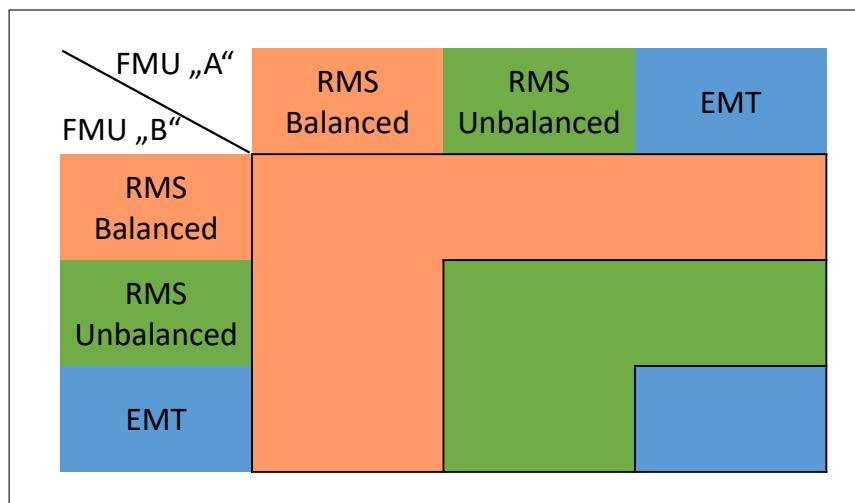


Figure 29.12.5: Selection of *Signal domain* depending on *Co-Simulation domain* of two participating *FMUs*

When using *Branch-based splitting* and the remote network is represented using a *Current-/Voltage-source equivalent*, the interchanged signals for each FMU are listed in Table 29.12.4.

Signal Domain	Input/ Output	Signal names	Unit	Description
RMS Balanced	Input	{foreign key}.bus{0/1}.{U/I}.re	kV/kA	Real part of positive sequence voltage/current
	Input	{foreign key}.bus{0/1}.{U/I}.im	kV/kA	Imaginary part of positive sequence voltage/current
	Output	{foreign key}.bus{1/0}.{I/U}.re	kA/kV	Real part of positive sequence current/voltage
	Output	{foreign key}.bus{1/0}.{I/U}.im	kA/kV	Imaginary part of positive sequence current/voltage
	Input	{foreign key}.bus{0/1}.{U/I}.A.re	kV/kA	Real part of phase A to ground voltage/phase A current
	Input	{foreign key}.bus{0/1}.{U/I}.A.im	kV/kA	Imaginary part of phase A to ground voltage/phase A current
	Input	{foreign key}.bus{0/1}.{U/I}.B.re	kV/kA	Real part of phase B to ground voltage/phase B current
	Input	{foreign key}.bus{0/1}.{U/I}.B.im	kV/kA	Imaginary part of phase B to ground voltage/phase B current
RMS Unbalanced	Input	{foreign key}.bus{0/1}.{U/I}.C.re	kV/kA	Real part of phase C to ground voltage/phase C current
	Input	{foreign key}.bus{0/1}.{U/I}.C.im	kV/kA	Imaginary part of phase C to ground voltage/current
	Output	{foreign key}.bus{1/0}.{I/U}.A.re	kA/kV	Real part of phase A current/phase A to ground voltage
	Output	{foreign key}.bus{1/0}.{I/U}.A.im	kA/kV	Imaginary part of phase A current/phase A to ground voltage
	Output	{foreign key}.bus{1/0}.{I/U}.B.re	kA/kV	Real part of phase B current/phase B to ground voltage
	Output	{foreign key}.bus{1/0}.{I/U}.B.im	kA/kV	Imaginary part of phase B current/phase B to ground voltage
	Output	{foreign key}.bus{1/0}.{I/U}.C.re	kA/kV	Real part of phase C current/phase C to ground voltage
	Output	{foreign key}.bus{1/0}.{I/U}.C.im	kA/kV	Imaginary part of phase C current/phase C to ground voltage
EMT	Input	{foreign key}.bus{0/1}.{U/I}.A	kV/kA	Phase A to ground instantaneous voltage/phase A instantaneous current
	Input	{foreign key}.bus{0/1}.{U/I}.B	kV/kA	Phase B to ground instantaneous voltage/phase B instantaneous current
	Input	{foreign key}.bus{0/1}.{U/I}.C	kV/kA	Phase C to ground instantaneous voltage/phase C instantaneous current
	Output	{foreign key}.bus{1/0}.{I/U}.A	kA/kV	Phase A instantaneous current/phase A to ground instantaneous voltage
	Output	{foreign key}.bus{1/0}.{I/U}.B	kA/kV	Phase B instantaneous current/phase B to ground instantaneous voltage
	Output	{foreign key}.bus{1/0}.{I/U}.C	kA/kV	Phase C instantaneous current/phase C to ground instantaneous voltage

Table 29.12.4: Signal interface specification for one common branch of two *FMUs* using *Branch-based splitting* and a *Current-/Voltage-source* remote network equivalent

where:

- {foreign key} is the foreign key (object attribute *for\_name*) of one *interface relevant element*;
  - bus{0/1} identifies the side of the branch that the signal is taken from;
  - U indicates that a voltage signal is being sent/received on this side;
  - I indicates that a current signal is being sent/received on this side;
  - A/B/C indicates the relevant phase being used;
  - re/im indicates that either the real or the imaginary part of the positive sequence vector is used.
- 

**Note:** • *PowerFactory* will always send voltage on one side of the branch and current on the other.

---

When using *Node-based splitting* and a *Multi-port Thevenin* equivalent representation of the remote network, the interchanged signals for each FMU are listed in Table 29.12.5. Elements within the local boundary generate output signals; elements not within the local boundary (which are thus removed in the network reduction procedure) generate input signals.

Signal Domain	Input/ Output	Signal names	Unit	Description
RMS Balanced	Input	{foreign key}.bus[0/1].U.re	kV	Real part of positive sequence voltage
	Input	{foreign key}.bus[0/1].U.im	kV	Imaginary part of positive sequence voltage
	Output	{foreign key}.bus[0/1].I.re	kA	Real part of positive sequence current
	Output	{foreign key}.bus[0/1].I.im	kA	Imaginary part of positive sequence current
	Input	{foreign key}.bus[0/1].U.A.re	kV	Real part of phase A to ground voltage
	Input	{foreign key}.bus[0/1].U.A.im	kV	Imaginary part of phase A to ground voltage
	Input	{foreign key}.bus[0/1].U.B.re	kV	Real part of phase B to ground voltage
	Input	{foreign key}.bus[0/1].U.B.im	kV	Imaginary part of phase B to ground voltage
RMS Unbalanced	Input	{foreign key}.bus[0/1].U.C.re	kV	Real part of phase C to ground voltage
	Input	{foreign key}.bus[0/1].U.C.im	kV	Imaginary part of phase C to ground voltage
	Output	{foreign key}.bus[0/1].I.A.re	kA	Real part of phase A current
	Output	{foreign key}.bus[0/1].I.A.im	kA	Imaginary part of phase A current
	Output	{foreign key}.bus[0/1].I.B.re	kA	Real part of phase B current
	Output	{foreign key}.bus[0/1].I.B.im	kA	Imaginary part of phase B current
	Output	{foreign key}.bus[0/1].I.C.re	kA	Real part of phase C current
	Output	{foreign key}.bus[0/1].I.C.im	kA	Imaginary part of phase C current
EMT	Input	{foreign key}.bus[0/1].U.A	kV	Phase A to ground instantaneous voltage
	Input	{foreign key}.bus[0/1].U.B	kV	Phase B to ground instantaneous voltage
	Input	{foreign key}.bus[0/1].U.C	kV	Phase C to ground instantaneous voltage
	Output	{foreign key}.bus[0/1].I.A	kA	Phase A instantaneous current
	Output	{foreign key}.bus[0/1].I.B	kA	Phase B instantaneous current
	Output	{foreign key}.bus[0/1].I.C	kA	Phase C instantaneous current

Table 29.12.5: Signal interface specification for one common node of two *FMUs* using *Node-based splitting* and a *Multi-port Thevenin* remote network equivalent

Where:

- {foreign key} is the *foreign key* (object attribute *for\_name*) of one *interface relevant element*;
- bus{0/1} identifies the side of the branch that the signal is taken from;
- U indicates that a voltage signal is being sent/received on this side;
- I indicates that a current signal is being sent/received on this side;
- A/B/C indicates the relevant phase being used;
- re/im indicates that either the real or the imaginary part of the positive sequence vector is used.

- 
- Note:**
- For *FMUs* using *Node-based splitting* and a *Multi-port Thevenin* remote network equivalent the *interface relevant elements* are those elements connected to the common terminal.
  - Both voltage and current are transferred in this approach.
  - The generated signals (without the signal type dependent decorations) can be viewed in the *ElmAgentio* that is generated during preparation.
- 

Further important notes:

- The reference system of each FMU is assumed to be *Nominal frequency*.
- When a *PowerFactory* FMU is used, *PowerFactory* applies the following *Calculation of Initial Conditions* settings: Reference (*iopt\_coiref*) set to “Nominal Frequency”.
- Frequency is implicitly exchanged between FMUs via the voltage and current signals. For example, in an RMS simulation, when the phasor voltage quantities are stationary, then the frequency is nominal  $f = f_n$ . When the phasor voltage quantities are oscillatory with a constant frequency  $df$ , then the actual frequency is  $f = f_n + df$ .
- if the FMUs are connected via multiple common nodes/branches, then for each common node/branch one set of signals (as per Tables 29.12.4 and 29.12.5) will exist in the interface.
- The *foreign key* is used to identify each common node/branch of two FMUs.

### 29.12.3 Performing out a Co-simulation and Retrieving Results

**Prerequisites:** With reference to Figure 29.12.1, the following requirements are enforced for a successful co-simulation:

- The co-simulation has already been configured; refer to Section 29.12.2.
- Availability of *PowerFactory* instances:
  - One *PowerFactory* instance is required for operating the *Simulation Master*
  - For each *PowerFactory* generated *.FMU*, one *PowerFactory* instance is required.
- For each “third party” *.FMU* it is assumed that the requirements of the *third party software* are fulfilled.
- Each communication link schematically represented in the figure is assumed to be operational.

To carry out a co-simulation, execute the following steps:

- Make sure that the co-simulation case has been successfully configured (refer to Section 29.12.2).
- For each *PowerFactory*-generated *.FMU* one *PowerFactory* instance must be started and configured as follows:
  - Each *PowerFactory* instance must contain the source project with the *FMU* specific study case active (an *FMU* dedicated network variation is also active). Network variations corresponding to other *FMUs* should not be activated in this *PowerFactory* instance.
  - *FMU Agent* must be configured using the *Initial conditions for co-simulation* command (*Com-Cosim* ). The applied settings in this case are:
    - \* *Basic Options page* → *General tab* → *Co-simulation type*: Co-simulation with external application
    - \* *Basic Options page* → *General tab* → *Task*: Run as *FMU Agent*
    - \* *FMU Agent* page configuration: The corresponding *FMU Agent I/O* object is automatically listed. It is possible to open this object and review the defined electrical input and output signals which will be linked via the FMI. The connection TCP/IP port of the agent is shown (and editable).

- If the FMU Agent is configured, then click **Execute**. In the output window the FMU Agent will provide relevant messages on the operation status.
- Re-do the steps above for each participating FMU Agent, one for each *PowerFactory* process.
- Based on one *PowerFactory* instance, a *Simulation Master* is configured using the *Initial conditions for co-simulation* command (*ComCosim* ). The settings used in this case are as previously detailed: *Co-simulation type*: Co-simulation with external application; *Task*: Run as simulation master.
- Go to the *Simulation Master* page and review the already configured information. Now set the *Synchronisation period* and the *Start time*.
- Click **Execute** in order to initialise the co-simulation. All participating FMU Agents should now make contact with the *Simulation Master* and compute the initial conditions in the individual regions.
- From the *Simulation Master* open the *Start Simulation (ComSim)* command dialog and set the simulation time.

After the simulation has been executed, results can be plotted, exported or post-processed as for any other dynamic simulation.

- At the end of a co-simulation, the *Results (ElmRes)* object linked in the *Basic Options* page of the *Initial conditions for co-simulation (ComCosim)* contains the results of elements corresponding to each *FMU Agent*. Results of elements belonging to external *FMU Agents* are not available.
- View and analyse results. The simulation results of elements within each *region* can be shown in a plot, as follows:
  - Click the *Insert Plot* icon , choose a *Curve plot* and then **OK**.
  - Under the “y-Axis” page, double-click on an empty field of the column *Results file* of the *Curves* tabular list.
  - Select the co-simulation *Results* object
  - Under column *Element* select the relevant element.
  - Under column *Variable* select the variable of interest. Click **OK**.

#### 29.12.4 The “*Preparation as FMU Agent*” (*ComCosimsetup*) Command

The co-simulation with external application makes use of the *Preparation as FMU Agent* command . The command executes a network reduction for each of the defined boundaries/regions and correspondingly generates *FMUs/.FMUs*. The changes to the power system model applied for each boundary are stored in a network variation. A set of  $n$  network variations are generated, one for each region. For each *FMU Agent* a dedicated study case is created.

##### **Basic Options** page

###### *Preparation steps:*

- Reduce and create FMU Agents: performs network reduction of the power system *regions* which are to be simulated externally and creates corresponding *FMU Agent I/O* objects for each active list entry. It creates one study case and one network variation for each *FMU Agent I/O* object. The network variation stores the network reduction modifications applied to the whole system.
- Reduce only: performs network reduction of the power system *regions* which are to be simulated externally. It also creates new *FMU Agent I/O* objects but no *.FMU* files. This option is useful when the *FMUs* are already created but the network variations or the network reduction must be re-created/re-executed e.g. some subsequent topological modifications have been made to the system.

- Create single FMU Agent: creates one single *.FMU* file based on an already existing *FMU Agent I/O* object (*ElmAgentio*). Note, the *FMU Agent* object (*ElmAgentio*) is described in detail in Section 29.12.5.

Tabular input options for *Reduce and create FMU Agents*:

- Active: For each row entry of this table (i.e. FMU Agent), a checkbox is available to include or exclude the row from the preparation step.
- Boundaries: For each FMU Agent, one *boundary* element (*ElmBoundary*) can be set in the corresponding field. The *boundary* defines the relevant region for this *FMU Agent*.
- Simulation method: For each FMU Agent, the simulation method can be selected in this field.
- Communication: Choose between Local, TCP/IP LAN and TCP/IP WAN;
- IP address: For the FMU Agent whose communication is not local, this field defines the IP address of the *FMU Agent*. If the *FMU Agent* is placed behind a router, this IP address is the router's address.
- WAN port: (optional) For each FMU Agent whose communication is TCP/IP WAN, this field defines the WAN's external port which will be accessed by the *PowerFactory .FMU*.

---

**Note:** **Reduce and create FMU Agents** is the default option used. The original, fully defined power system, study case should be used. Based on the power system topology of this study case, a number of FMU Agents can be defined. Upon execution of this command, for each FMU Agent, a new dedicated study case is generated along with required configuration.

---

Tabular input options for *Reduce only*:

- Active: For each row entry of this table (i.e. FMU Agent), a checkbox is available to include or exclude the row from the preparation step.
- Regions/ElmBoundary: For each FMU Agent, one *boundary* element (*ElmBoundary*) can be set in the corresponding field. The *boundary* defines the relevant region for this *FMU Agent*.
- Simulation method: For each FMU Agent, the simulation method can be selected in this field.

---

**Note:** **Reduce only** is typically used when only topological changes are applied on a given *FMU Agent* which do not affect other FMUs nor do they result in different boundary settings/configuration. In this situation, the dedicated study case for the targeted *FMU Agent* should be activated, and this option then used.

---

Input options for *Create single FMU Agent*:

- FMU Agent I/O definitions / *ElmAgentio*: in this table, the relevant boundaries can be selected, one for each FMU Agent. One *.FMU* file is generated for each entry row.
- Network: For **all** *PowerFactory* FMU Agents, the network type can be selected for the interface links (loopback and WAN) depicted in Figure 29.12.1. If checkbox is not set, the communication network is local (i.e. using loopback interface "127.0.0.1" in the machine hosting the *Simulation Master* process). If checkbox is set, the communication network is defined by subsequent fields *IP address* and *WAN port*.
- Communication: Choose between Local, TCP/IP LAN and TCP/IP WAN;
- IP address: For the FMU Agent whose communication is not local, this field defines the IP address of the *FMU Agent*. If the *FMU Agent* is placed behind a router, this IP address is the router's address.

- WAN port: (optional) For each FMU Agent whose communication is not local and the machine hosting the FMU Agent is behind a router, this field defines the router's external port which will be accessed by the *PowerFactory .FMU*.

**Note:** **Create single FMU Agent** is typically used when an already created *FMU Agent I/O* object is available (obtained via the Preparation as FMU Agent command) but the *.FMU* file still needs to be created. In such a situation, the dedicated study case for the targeted *FMU Agent* should be activated, and this option further used.

*Method:* Choose between *Implicit method(exact approach)* and *Explicit method(approximate approach)*. Note that the *Implicit method* is only available when all co-simulation units are *PowerFactory* FMU Agents, and each one is set to support the *Implicit method*.

*Load flow:* A reference link to the currently used *Load Flow Calculation* command is provided.

*Initial conditions:* A reference link to the currently used *Initial Conditions* command is provided.

#### **Reduction page - Transformation Settings:**

*Transformation settings* - this pane can be configured in order to appropriately represent the conversion of electrical quantities between the RMS and EMT domains. The following options are available:

- *Same transformation for all:* A single transformation method is used for all *boundary branches* involved in the co-simulation; at the bottom of the dialog, the relevant *RMS-EMT transformation settings (SetTransf)* object is linked.
- *Different transformation per region:* For each set of two *neighbouring regions* one transformation is assigned. All possible combinations are defined in the associated list.

**Note:** Whenever available, the tabular list can be automatically populated by clicking on the button *Reset and load all*. For each possible combination in the list, the *Settings* column can be configured with *RMS-EMT transformation settings (SetTransf)* objects. The current list is completely removed by clicking on the button *Clean up*. Additionally, the *Clean up* command will also delete the unused transformation settings (*SetTransf*) objects, usually located inside the *Initial conditions for co-simulation* command object.

#### **Reduction page - Remote Network Representation:**

*Equivalent for remote networks* pane has the following options:

- *Current/Voltage sources at boundary buses:* The applied network reduction method uses simple current/voltage sources at the boundary buses for the representation of remote networks.
- *Multi-port Thevenin equivalent:* The network reduction computes a multi-port equivalent of the entire remote network of each *region*. The method has the following options:
  - *Conversion to stable EMT equivalent where required:* The reduced network equivalent may sometimes be numerically unstable due to the existence of negative resistances. If this flag is active, then any negative resistance is set to zero.
  - *Boundaries without reference machine* pane:
    - \* *Alternative locations for reference machines:* select the bus/set of buses where a reference machine (slack) element is to be inserted if one or a set of regions have missing reference machine elements. A *region* (boundary) without reference machine is a *region* that contains only passive elements, loads and no elements with grid-forming capability (e.g. synchronous machines, static generators with reference machine capability). If no

bus or set of buses is assigned, then the program automatically assigns reference machine elements (a voltage source element), in regions with missing reference machines at the *Boundary/Edge nodes*.

*Options for calculations on grid before reduction*

*Do not adapt Load Flow Calculation Method to Simulation method of regions:* Set this flag in order to not adapt Load Flow Calculation Method to Simulation method of regions when computing calculations before regions are separated.

---

**Note:** When this option is not set, *PowerFactory* may apply modifications to the *Load Flow Calculation* command executed before splitting the network into regions. Namely, the balanced/unbalanced load flow method may be changed depending on the choice of the dynamic simulation method of regions. Without this option, some cases may lead to a failed load flow calculation. Use this option to avoid such situations.

---

*Skip Calculation of Initial Conditions:* Set this flag in order to avoid the execution of the Calculation of Initial Conditions of the whole network (i.e. before regions are separated).

---

**Note:** When this option is not set, *PowerFactory* will always perform a *Calculation of Initial Conditions* command before regions are separated. In some cases, a combined simulation of the entire network is nevertheless not possible (e.g. case of an RMS/EMT co-simulation with local models designed only for either the RMS or the EMT simulation domain, but not for both), and therefore, a combined dynamic simulation start will lead to errors when executing the co-simulation. To avoid such situations, make sure to set this option.

---

**FMU page - General:**

*FMU base name:* The FMU's name

*Output folder:* The folder where the *.FMU* files are saved.

*Local port:* The IP port of the *FMU Agent* for inbound connections from the *Simulation Master* can be configured here. If multiple *FMU Agents* are generated, then this field defines the local port of the first *FMU Agent*, any subsequent *FMU Agents* being assigned local ports which are indexed starting with the first one. The end of range local port (the local port of the last *FMU Agent*) is provided as an information message.

*Encrypt communication:* For all *PowerFactory* FMU Agents, TCP/IP communication can be encrypted using this flag.

*Configured Co-Simulation:* A reference to the *Initial Conditions for Co-Simulation* is automatically provided here for easy access. The settings of this object can be easily accessed using this reference.

**FMU page - Additional Info:**

*Author:* A user-defined text field for storing author information that is saved in the *.FMU*.

*Version:* A user-defined text field for storing version information that is saved in the *.FMU*.

*Copyright:* A user-defined text field for storing copyright information that is saved in the *.FMU*.

*Description:* A user-defined text field for storing description information that is saved in the *.FMU*.

*Licence:* A user-defined text field for storing licence information that is saved in the *.FMU*.

**Output page:** Choose between short and detailed output messages.

### 29.12.5 FMU Agent I/O and FMU Agent objects

The *FMU Agent I/O* object (*ElmAgentio*) has the following configuration options:

#### Basic Data page

- Out of service: flag to disable the *FMU Agent I/O*
- Simulation methods of remote boundaries: The simulation domain can be selected for each boundary
- *Transformation settings* - this pane can be configured in order to appropriately represent the conversion of electrical quantities between the RMS and EMT domains. The following options are available:
  - *Same transformation for all*: A single transformation method is used for all *boundary branches* involved in the co-simulation; at the bottom of the dialog, the relevant *RMS-EMT transformation settings* (*SetTransf*) object is linked.
  - *Different transformation per boundary*: For each boundary one transformation is assigned. All possible combinations are defined in the additional table column.

#### Grid Connections page

- Electrical Inputs: the FMI compliant input signals are listed in this tab
- Electrical Outputs: the FMI compliant output signals are listed in this tab
- Additional Information: this tab provide information regarding the branches used for the network equivalent (these are common impedance (*ElmZpu*) elements).

The *FMU Agent* object (*ElmAgent*) has the following configuration options:

#### Basic Options page:

- Name: the name of the object
- Path: the file path to the *.FMU* file
- Out of service: flag to disable the *FMU Agent*
- Further information detailing the properties of the *.FMU* file.

#### Signals page:

- List of signals: provides a tabular list of all signals exchanged by the FMU Agent, along with individual signal details.

#### Description page:

- Description: a text field which contains an FMU text description (if any);
- License: a text field which contains an FMU licence description (if any);

### 29.12.6 The “Initial conditions for co-simulation” (*ComCosim*) Command

The co-simulation with external application makes use of the *Initial conditions for co-simulation* command . As the command is described in detail in Section 29.11.4, only the relevant information regarding the external application based co-simulation is given here.

**Basic Options page - General tab**

*Co-simulation type:* select the option “Co-simulation with external application”.

*Task pane:*

- Run as simulation master
- Run as FMU Agent

Options for *Run as simulation master*:

- *Information message:* The number of selected regions for co-simulation
- *Synchronisation period:* via this field, a user-defined fixed synchronisation period between the co-simulated *regions* can be provided;
- *Start time* - the co-simulation start time (identical for all FMU Agents)

Options for *Run as FMU Agent*:

- *Information message:* The number of neighbouring regions for co-simulation (for the specified FMU Agent)
- *Initial conditions:* a reference to the *Calculation of initial conditions (ComInc)* command which is to be used for the *FMU Agent*;
- *Results:* A selection link to the results file (*ElmRes*) which stores the co-simulation results for the FMU Agent.

**Simulation Master page**

- The previously generated FMU Agents are automatically loaded. Each row in the tabular list contains a reference to an *FMU Agent* object (*ElmAgent*). The *FMU Agent* object defines the file path to the previously generated *.FMU* file existing on the Simulation Master machine. Also, the *FMU Agent* can be set to *out of service*.
- The *Add Agent* command can be used to add another FMU agent. This is useful whenever a third party or externally generated *.FMU* file is available, yet it was not previously generated by the *Preparation of FMU Agent* command.

**FMU Agent page** (these settings are applicable only for *PowerFactory .FMUs*)

- *Port:* the local TCP/IP port of the FMU Agent for inbound requests from the *.FMU*. Note, changing the port number here without updating the *.FMU* file will lead to a broken connection.
- *Initial conditions:* (informative) a reference to the *Calculation of initial conditions (ComInc)* command which is to be used for the *FMU Agent*;
- *Simulation command:* (informative) a reference to the *Run simulation (ComSim)* command which is to be used for the *FMU Agent*;
- (informative) The corresponding *FMU Agent I/O* object is automatically listed. It is possible to open this object and review the defined electrical input and output signals which will be linked via the FMI.

### 29.12.7 Terms and Definitions for Co-simulation with External Application

The terms and definitions used for the *Co-simulation with External Application* are detailed below.

**Co-Simulation:** Simulation in parallel of different sub-systems which form a coupled problem. Each sub-system represents a part of a power system dynamic simulation model.

**Simulation unit:** A *Simulation unit* consists of one time domain simulator (referred to as a *solver*) and its associated power system model (referred to as a *region*).

**Co-Simulation with external application:** A *co-simulation* involving *PowerFactory simulation units* and (optionally) third-party developed *simulation units*. The *Co-Simulation with external application* follows a distributed computing architecture, allowing *simulation units* to be executed on different machines.

**Co-Simulation domain:** The analysis domain of the coupled problem. Within the *PowerFactory's co-simulation with external application* the *co-simulation domains* are: RMS balanced, RMS unbalanced and EMT (refer to Section 29.2).

**Simulation master:** A computer program that coordinates all *simulation units* in a combined *co-simulation with external application*.

**Region:** Within the *co-simulation with external application*, multiple *simulation units* can be included, while the associated power systems model of each unit is referred to as a *Region*. This is due to the fact that in the case of power system analysis, the linking between the *simulation units* is typically done at the border between physically separated regions of the power system, as shown in Figure 29.11.1. The *co-simulation domains* are individually assigned for each *region*. A *region* is defined in *PowerFactory* by an associated *Boundary* object (*ElmBoundary*). *Boundary* objects are saved in *Network Model* → *Network Data* → *Boundaries* folder of the *PowerFactory* project.

**Internal region elements:** Those power system elements which are located inside the *region*. In *PowerFactory* the *internal region elements* are identical with the internal elements of the *Boundary* object defining the *region*.

**External region elements:** Those power system elements which are located outside the *region*. In *PowerFactory* the *external region elements* are the elements external to the *Boundary* object defining the *region*.

**Neighbouring region:** A *region i* is a *neighbouring region* with respect to *region j* if at least one electrical connection path exists between the two *regions* that does not cross through any other *region*.

**Interface elements:** The intersection between the elements of a *region* and one of its *neighbouring regions* defines the *interface elements* of the two *regions*.

**Edge node** (of a *region*): Any node (terminal) in the power system model which is directly connected with elements both **within and outside** the respective *region*. In *PowerFactory* a node is defined by the terminal object, of class *ElmTerm*. The *Edge nodes* must not necessarily be *internal region elements*.

**FMI:** The *Functional Mock-up Interface* (FMI) is a free standard which facilitates the exchange of simulation models among a large number of simulation programs. The FMI is a multi-purpose standard, i.e. the co-simulation of power systems is only one application. Currently, more than **150 tools** actively support FMI. *PowerFactory* supports FMI version 2.0.

**FMU:** A *Functional Mock-up Unit* (FMU) component, according to *FMI* standard.

**.FMU:** An FMU file (i.e. the realisation of an *FMU*), directly accessible by the *Simulation Master*. The *.FMU* is compliant with the requirements of exchanged signals with the *Simulation Master*, as specified in Section 29.12.2.6.

**FMU agent:** A *Simulation Unit* whose corresponding *.FMU* complies with the *FMI for co-simulation* part of the *FMI* standard version 2.0.

**Master process:** A process of a computer program which contains the *Simulation Master*. The *Master process* belongs to *PowerFactory*.

**Agent process:** A process of a computer program which contains the *FMU agent*. The *Agent process* belongs either to *PowerFactory* or a third party program.

**Foreign key:** The elements connected to an [edge node](#) (of a [region](#)) are used in the signal definition of the [FMI](#) compliant interface between the [simulation master](#) and the [.FMU](#). These elements need to be uniquely identified throughout the co-simulation interface using identifiers. For the purpose of [co-simulation with external application](#) this unique identifier is called a *foreign\_key*. Within *PowerFactory* every object has a dedicated *foreign\_key* (i.e. attribute “*for\_name*”), found in the *Description* page of the element’s Edit dialog. By default, the attribute *for\_name* is empty, hence for all elements connected to an [edge node](#) of any [region](#) it must be manually defined. This will ensure a correctly defined signal list of each [FMU](#). It is recommended to define a *foreign\_key* using only letters and numbers and avoiding special characters and blank spaces.

## 29.13 Frequency Response Analysis

*PowerFactory* supports the computation of the frequency response of dynamic models implemented in DSL. The calculation uses time-domain simulation and it is implemented within the Frequency Response Analysis command (*ComFreqresp*).

*Frequency Response Analysis* is intended for use within dynamic systems which are linear and time-invariant around their initial operating point. Non-linear and discontinuous behaviour of dynamic models may affect the accuracy of results. Special care needs to be taken in order to make sure that:

- a flat start simulation is existing as a prerequisite;
- the operating point does not change throughout the simulation;
- any dynamic models which, around their operating point, are non-linear, contain variations over time or generate discontinuities are excluded from the simulation.

### 29.13.1 Basic Usage

To perform a Frequency Response Analysis calculation two approaches are possible:

- Via the *Simulation RMS/EMT* toolbox in the main toolbar:
  - Click on the *Calculation of Frequency Response* icon 
  - The Frequency Response Analysis command dialog is shown.
- Via the Composite Model Graphic:
  - Show the graphic of the Composite Model which contains the DSL model of interest.
  - Right-click one input signal of a slot containing the DSL model of interest and choose *Calculation → Calculation of Frequency Response...*
  - The Frequency Response Analysis command dialog is shown.

The Frequency Response Analysis calculation settings are configured using the command dialog via the three available pages as below:

- Basic Data
- Output
- Advanced

These pages are described in detail in the following subsections.

Once the settings have been configured, the calculation can be started using the **Execute** button.

### 29.13.2 Basic Data Page

In the *Basic Data* page - *General*, the following settings are available:

**Plots** pane: There are two options for generating plots:

- **Bode**: Check this flag in order to generate a Bode Diagram.
- **Nyquist**: Check this flag in order to generate a Nyquist Plot.

**Frequency analysis data** pane: here the frequency range of interest, the frequency step and the individual frequency amplitude by which the input signal is being tested can be configured.

- **Step**: the frequency step in Hz used by the frequency response analysis. This parameter will impact both the frequency step by which the input signal is tested and the step in the output plots. Note that this parameter is inversely proportional to the duration of one simulation window. The minimum value that can be set in this field is 0.005 Hz.
- **Start at step**: the frequency step at which the frequency response analysis starts. The input field accepts a non-zero integer number. This parameter will impact both the minimum frequency at which the input signal is tested and the minimum frequency of the output plots.
- **Minimum frequency**: the actual minimum frequency in Hz being used in the calculation is displayed next to the **Start at step** input field. This parameter can only be changed by modifying parameters **Start at step** and **Step**.
- **End at step**: the frequency step at which the frequency response analysis ends. The input field accepts a non-zero integer number. This parameter will impact both the maximum frequency at which the input signal is tested and the maximum frequency of the output plots.
- **Maximum frequency**: the actual maximum frequency in Hz being used in the calculation is displayed next to the **End at step** input field. This parameter can only be changed by modifying parameters **End at step** and **Step**.
- **Amplitude**: the amplitude of the input signal at minimum frequency defined as an absolute value. Note that this parameter can be adjusted upwards/downwards such that the particular model being tested does not go out of the initial linear operating range.

**Simulation data** pane: this is used for configuring the dynamic simulation used by the Frequency Response Analysis command.

- **Initial conditions**: the *Initial conditions* object being used is referenced within this field. Edit the settings of the initial conditions according to the analysis requirements.

---

**Note:** Adjustments to initial conditions are possible and recommended in order to fine tune the results. The calculation of initial conditions can be executed independently of the *Frequency Response Analysis* in an initial stage in order to make sure that a flat start simulation is set up. Note also that only fixed time step simulation is supported.

---

- **Integration step size**: the simulation integration step size in seconds is displayed. This value can be changed via the *Initial conditions* object.
- **Recommended value**: the recommended value of the simulation integration step size. This parameter is shown if (and only if) the simulation integration step size is larger than this value. Simulation integration step sizes smaller than the recommended value are accepted but may lead to larger execution times. Simulation integration step sizes larger than the recommended value are accepted up to the Nyquist criterion limit but may lead to inaccurate results.
- **No. of windows**: the number of windows used for the simulation. For simulations in which a flat start simulation is not possible, a larger number of windows is required, and proportional to the decay time of the initial transients. The calculation of the *Frequency Response Analysis* is

executed using the simulation results of the last window, while the results of all other time windows are not considered.

**Note:** Adjusting the number of windows is needed only if the simulation does not have a flat start.

If this is the case, then one simulation using the aforementioned *Initial conditions* should be performed independently of the *Frequency Response Analysis* calculation. The simulation time until steady state is obtained should be measured and divided by the duration of one window of the *Frequency Response Analysis*. The number of windows must then be greater than this value.

- **Simulation duration:** the total simulation time in seconds is displayed.

**Transfer function input/output definition** pane: this pane is used for the selection of input and output signals of the transfer function(s).

- **Composite Model:** the composite model which contains the DSL model of interest (optional setting). This field limits/filters the selection of the *Input DSL* model being used to the ones which are contained within this composite model. This setting can be useful in case of a project with a large number of DSL models.
- **Input DSL:** the DSL model whose input signal is defined within the transfer function(s). The *Input DSL* selection window contains either all calculation relevant common models within the active project (if no *Composite Model* is selected) or all calculation relevant common models located within the *Composite Model* (if such a *Composite Model* has been defined in the above field).
- **Input signal:** the input signal of the transfer function(s). A drop down list containing all available input signals of the *Input DSL* model is made available. From the given signal list, define the input signal of the transfer function(s).
- **Output signal(s):** the number of output signals (if any) of the transfer function(s) is displayed. These output signals are defined using the *Add* button, shown using the *Show* button and removed using the *Remove All* button. The number of generated transfer functions equals the number of selected output signals. Each transfer function has the same input signal while the output signal may differ.
- **Show button:** the output signal(s) being used for the computation of the transfer function(s) along with the model(s) to which they belong can be shown as follows:
  - Click on the *Show* button;
  - Open the Variable Selection dialog corresponding to one of the elements shown in the list (e.g. right-click the corresponding row and select *Edit*);
  - The output signal(s) are listed within the *Selected Variables* field.
- **Add button:** one or more output signals can be defined using the *Add* button as follows:
  - Click on the *Add* button;
  - Select the element which contains the variable of interest. If not shown directly in the selection dialog, other calculation relevant elements can be chosen by using the item *Signal(s) from other elements*.
  - Open the *Variable Selection* window corresponding to this element (e.g. if available directly in the selection dialog then double-click element's icon)
  - Identify the variable of interest (i.e. using either the *Simulation RMS* or the *Simulation EMT* filters) and add it to the *Selected Variables* field.
  - Click the *OK* button to add the variable

**Note:** The output signal(s) of the transfer function can be any calculation relevant variable, not only output signals of DSL models. Nevertheless, do make sure that the output signal has practical meaning within the context of the specific study.

- **Remove All button:** Use this button to remove all already selected output signals.

In the *Basic Data* page - *Input signal diagram*, the waveform of the injected fourier signal and the amplitude spectrum (see Ch. 29.13.7) over one time window are shown.

### 29.13.3 Output Page

The following general settings are subject to user configuration on the *Output* page:

**Results:** The object in which the *Frequency Response Analysis* results are stored.

**Output stability margin indices:** If selected, the gain and phase margin indices are calculated and displayed in the output window. If the Nyquist plot option is checked in the *Basic Data* page then the Module and Delay margins are additionally printed to the output window.

---

- Note:**
- The Gain margin, in dB, is the reciprocal of the magnitude of the open-loop transfer function calculated at the frequency at which the phase angle is  $-180^\circ$  (i.e. the phase crossover frequency).
  - The Phase margin, expressed in degrees, is calculated as  $180^\circ$  plus the phase angle  $\phi$  of the open loop transfer function at the frequency at which the magnitude of the open-loop transfer function is unity (i.e. the gain crossover frequency).
  - The Module margin is defined as the minimum distance between the Nyquist trajectory and the critical point  $(-1,0)$ .
  - The Delay margin, in seconds, equals the phase margin divided by frequency  $\omega_{\phi_m}$ . The frequency  $\omega_{\phi_m}$  represents the frequency on the Nyquist curve at which the phase margin is calculated. The Delay margin represents the time it takes for a line containing the complex plane origin and the phase margin crossing point to rotate clockwise with frequency  $\omega_{\phi_m}$  until it overlaps the line containing the complex plane origin and the critical point.
- 

**Normalise output signal:** The transfer function magnitude is calculated including the magnitude of both the input and output signals. If unchecked, the input signal magnitude is disregarded in the calculation of the transfer function. For almost all analysis types, this flag is checked.

**Bode plot** pane: A number of options are available for customisation of the Bode Diagram.

- **Show original signal:** The time domain simulated waveform of the transfer function output signal is shown within the Bode Diagram along with the magnitude and phase of the transfer function. The time domain waveform of the output signal can be useful when adjusting the *Frequency Response Analysis* parameters available in the *Basic Data* page (e.g. ensuring that the output signal waveform is not saturated, etc.)
- **Linear x-axis:** The Bode Diagram is presented using a linear axis for the frequency.

**Nyquist plot** pane: A number of options are available for customisation of the Nyquist plot.

- **Show original signal:** The time domain simulated waveform of the transfer function output signal is shown within the Nyquist plot along with the complex plane diagram. The time domain waveform of the output signal can be useful when adjusting the *Frequency Response Analysis* parameters available in the *Basic Data* page (e.g. ensuring that the output signal waveform is not saturated, etc.)
- **Show positive frequencies only:** The Nyquist plot is drawn using only the results corresponding to non-zero positive frequencies.
- **Complementary sensitivity margin:** A guiding circle is shown on the Nyquist plot. Let  $M$  be the inverse of the complementary sensitivity margin ( $M$  is known in literature as the magnitude of the closed-loop frequency response of unity feedback systems) then on the Nyquist plot a circle of radius  $|M/(M^2 - 1)|$  and centered on  $(-M^2/(M^2 - 1), 0)$  is added. In general, all points on this circle represent constant values of magnitude of the complementary sensitivity transfer function or, in the case of unity feedback systems, constant values of magnitude of the closed loop transfer function (i.e. M circle).

### 29.13.4 Advanced Page

The following general settings are subject to user configuration in the *Advanced* page:

**Consider simulation events:** If ticked, then any pre-defined RMS- or EMT-simulation events/faults will be executed during the *Frequency Response Analysis*. If unticked (default), any pre-defined RMS- or EMT-simulation events/faults will not be executed during the *Frequency Response Analysis*.

### 29.13.5 Output Plots

Upon executing the *Frequency Analysis* command, new plots are generated automatically and shown on a new page. Depending on the command settings, the following plots are presented:

- Bode Diagram:
  - Transfer function magnitude plot (magnitude over frequency)
  - Transfer function phase plot (phase over frequency)
  - (optional) Output signal waveform (time-series of the output signal in the last time window)
- Nyquist Plot:
  - Nyquist curve (complex plane representation)
  - (optional) Output signal waveform (time-series of the output signal in the last time window)

### 29.13.6 Output Results Files

Calculation results of the time domain simulation are stored in one *Results Object* (*ElmRes*) located by default in the currently active study case (a subfolder of the *Study cases* project folder). The default name of the results object is “Frequency Response”. Calculation results of the transfer function are stored in the *Results Object* stored within the Plot itself for both magnitude and phase angle.

### 29.13.7 Application notes and guidelines

The procedure followed by the Frequency Response Analysis tool is as follows:

- Identifies the initial value of the input test signal by executing the *Calculation of Initial conditions* without modifying the original system (step 1 in Figure 29.13.1). The initial value of the input test signal will be stored and further used.
- Sets up a Fourier Source element and configures it. The configuration is done such that the following injected signal components are programmed in the Fourier Source
  - the initial value of the test signal (when the upstream model was connected); this initial value is programmed as being the DC component of the Fourier source
  - all other sinusoidal signal injections of the Fourier Source.
- Replaces the original link between the tested input signal in the tested model and any signal coming from an upstream model (if any) - refer to step 2 in Figure 29.13.1. The Fourier source output is then connected to the input test signal of the tested model thus opening any loop in the original model. If the variable in which the input test signal is to be injected is part of a closed loop, then the user must create an additional input signal in the model that allows for the connection of the Fourier source without opening the original closed loop.
- Runs a dynamic simulation

- Computes the FFT of the “input signal”  $u(t)$ . Computes the FFT of a second signal (regarded as “output signal”  $y(t)$ ) from the simulation results (the user selects both input and output signals from the command dialog; multiple output signals can be selected for the same calculation, which results in multiple transfer function plots)
- Computes the quotient between the magnitude of the output/input FFT results at each calculated frequency in the FFT.
- Computes the difference between the phase angles of the output and input FFT results at each calculated frequency in the FFT.
- Displays a “Bode Plot” which contains the quotient of the magnitudes on one plot and the phase angle difference in another plot as functions of frequency. This is regarded as the Bode plot of the “transfer function” of the output/input pair. Alternatively, a Nyquist Plot can be generated, in which the complex value of the transfer function for varying frequencies is displayed in the complex plane.

### Fourier (input test) signal definition in the Frequency Response Analysis

In the above, a two steps procedure is adopted for configuring and linking the Fourier Source signal with the input signal under test, as depicted in Figure 29.13.1.

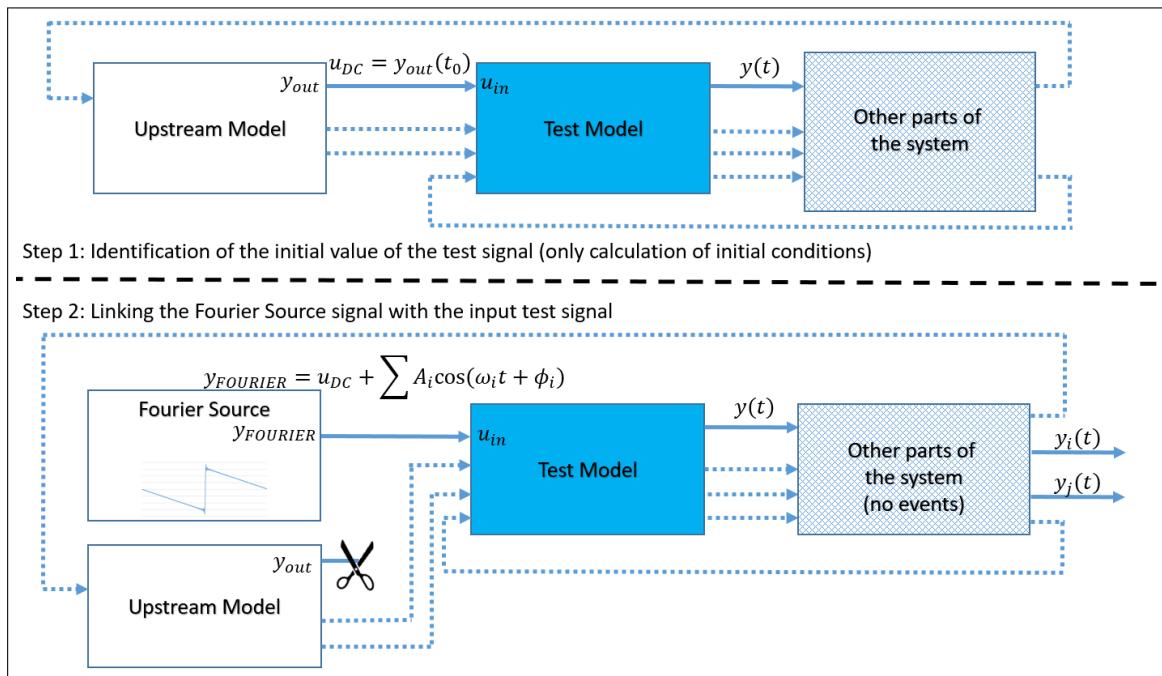


Figure 29.13.1: Setup procedure of the frequency response analysis tool

As such, the implementation has no switch that makes the transition between the Fourier source and any pre-existing upstream models (either DSL or of other type) that might be connected to that particular signal. The initial value of the signal is identified from the calculation of initial conditions using the original system, in which the Fourier source is not connected. This initial value is regarded as a bias and programmed in the Fourier Source as a DC component ( $u_{DC}$ ). The test signal  $u(t)$  contains a number of sinusoidal signal components, as further described:

$$u(t) = y_{FOURIER}(t) = u_{DC} + \sum_{i=StartStep}^{EndStep} A_i \cos(\omega_i \cdot t \pm \phi_i),$$

where:

- *StartStep* is the parameter **Start at step** and *EndStep* is the parameter **End at step** from the calculation dialog described in Section 29.13.2
- $i$  is a given integer step between *StartStep* and *EndStep*
- $A_i$  is the amplitude of the sinusoidal signal component at step  $i$
- $\omega_i$  is the angular frequency of the sinusoidal signal component corresponding to step  $i$  and equal to  $\omega_i = 2\pi(\text{MinFrequency} + (i - 1)\text{Step})$ . *Step* is the parameter **Step** from the calculation dialog described in Section 29.13.2. Variable *MinFrequency* is equal to *StartStep* · *Step*
- $\phi_i$  is the angular phase displacement of the sinusoidal signal injection at step  $i$ . The phase angle is set constant to  $\phi_i = \frac{\pi}{2}$ .
- in the equation above, a positive phase angle is chosen for every step  $i = k * \text{StartStep}$ ; a negative phase angle is chosen for every step  $i = k * \text{StartStep} + 1$ , with  $k$  an integer number,  $k = 1.. \frac{\text{EndStep}}{2}$ .

The amplitude parameter of each frequency component  $A_i$  is defined as such:

$$A_i = \frac{A_0}{\omega_i/(2\pi)} = \frac{A_0}{f_i}$$

where  $f_i$  is the frequency in Hz of the sinusoidal signal component at step  $i$

The signal  $u(t)$  has the following properties:

- the value of the signal at start time  $t_0 = 0$  is equal to  $u_{DC}$
- the value of the signal's time derivative at start time  $t_0 = 0$  is equal to 0.

### Practical application notes

Caution should be taken when parameterising the Frequency response analysis tool.

The following requirements need to be fulfilled:

- the analysed model is in a quasi steady-state operation point and that the excitation signal does not move the operation point away to another one.
- the model is behaving linearly around its current operation point

The default value of the **Amplitude** parameter is set to 0.01 which sometimes might be too large to maintain a linear operating point for the tested model. The maximum deviation of the test signal from its DC bias is calculated as the sum of the magnitudes of all sinusoidal signal injections programmed in the Fourier Source. As such, the norm of the test signal deviation fulfils:

$$\|u(t) - u_{DC}\| < \sum_{i=\text{StartStep}}^{\text{EndStep}} \frac{A_0}{f_i}$$

---

**Note:** The optimal amplitude of the Fourier Source signal will depend on the typical range of the input test signal and the degree of non linearity of the system (typically when signals are using a p.u. system, then the **Amplitude** parameter should be relatively small e.g. 0.001 p.u.).

---

From the above, it is seen that for frequency step increments, the amplitude decays with a double linear logarithmic characteristic, in which the characteristic will always intersect the frequency 1 Hz at  $A_0$ . An example is provided in Figure 29.13.2, in which  $A_0 = 0.01$ ,  $\text{Step} = 0.1\text{Hz}$ ,  $\text{StartStep} = 1$  and  $\text{EndStep} = 100$ .

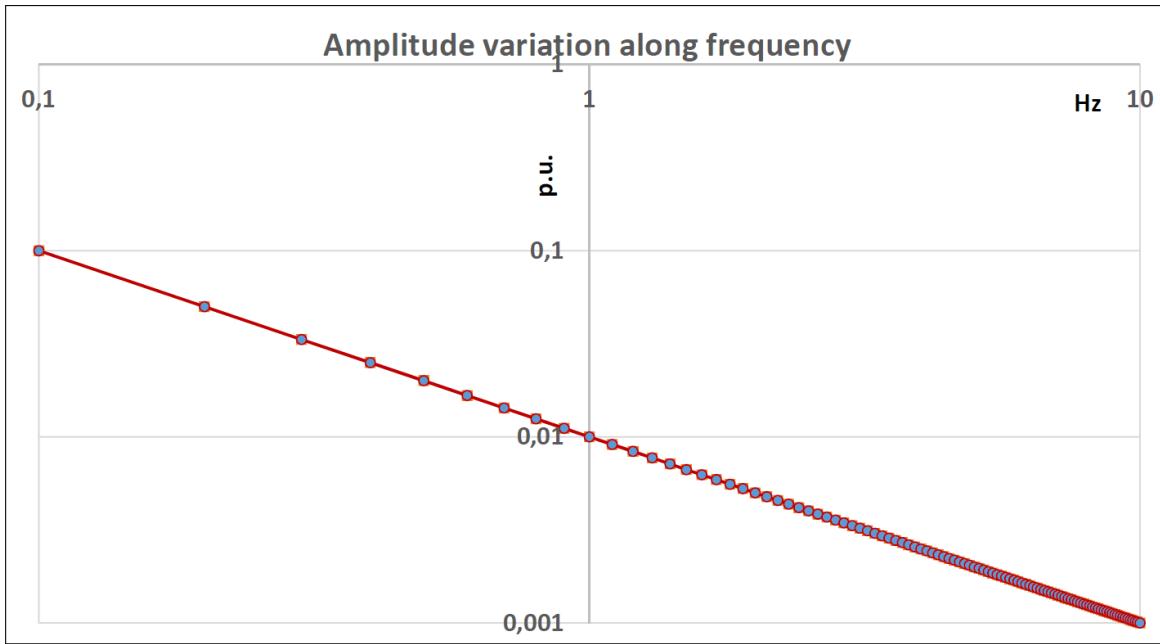


Figure 29.13.2: Amplitude variation along frequency

For the above parameters, the resulting input test signal  $u(t)$  looks as in Figure 29.13.3 (where the DC component is excluded):

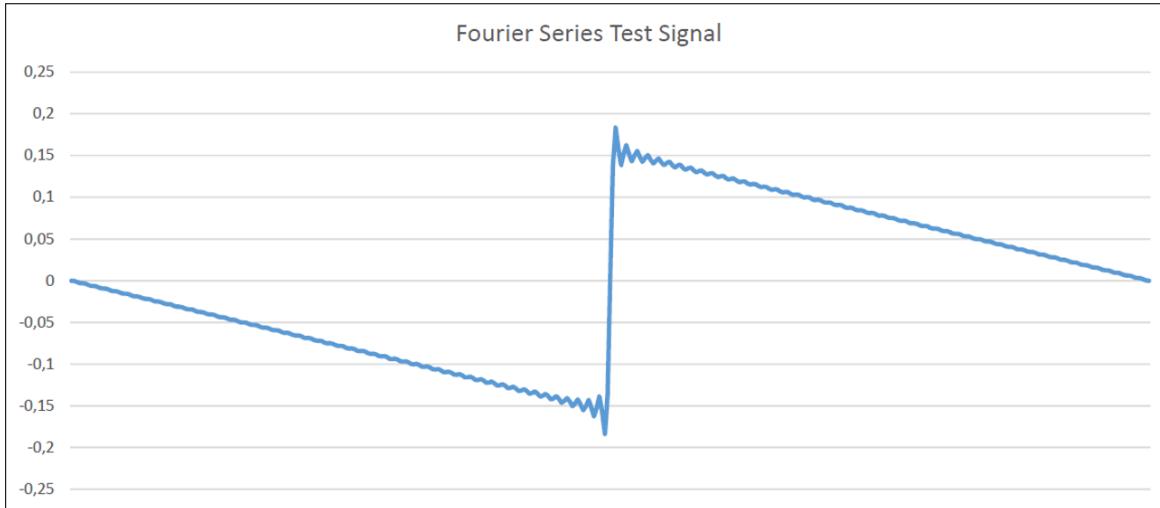


Figure 29.13.3: Exemplary input test signal

The **No. of windows** parameter should also be carefully selected. The Frequency Response analysis simulates the system for a time duration of  $t = N_{windows}/Step$  [s]. The last  $1/Step$  seconds in the simulation will be used by the command to compute the transfer function. In this period it is important that any external perturbation of the simulation quantities are avoided. For example, slowly decaying transients should have already been extinguished, various simulation events should not be triggered, etc. Sometimes, whenever frequency response results seem implausible, it is recommended to increase the **No. of windows** parameter to a larger value and test whether the generated results are the same. If different results are obtained this is an indication that the simulation signal has not yet reached a steady state for that particular time window.

Finally, it is recommended that an in-depth analysis of the equations of the tested model is performed to ensure that the simulation model does not introduce self-triggered events and to assess the effect of

discrete non-linearities on the frequency response.

## 29.14 Frequency Analysis

*PowerFactory* supports two important options for performing signal frequency analysis:

- Fourier Analysis, using Fast Fourier Transform (FFT), and
- Prony Analysis.

Both these are offered in the Frequency Analysis command (*ComFrequency*).

### 29.14.1 Prony Analysis Overview

While the FFT Analysis is commonly known, Prony Analysis is a calculation method which allows to calculate a frequency decomposition of a given signal. Prony Analysis belongs to the general topic of exponential data fitting. This field of research aims to find a sum of exponentials which fits best to a given series. Prony Analysis and FFT differ in several respects:

- Prony Analysis is not limited to a discrete grid of frequencies, whilst FFT is restricted to discrete frequencies, i.e. Prony Analysis detects the exact frequency of a certain harmonic.
- Unlike FFT, Prony Analysis is able to detect and quantify harmonic wave trends in time, i.e. increasing or decreasing waves (waves with damping) in the signal.

Prony Analysis decomposes a given signal into damped sinusoidal oscillations, the so-called modes of a signal. In contrast to the Fast Fourier Transform (FFT), where the algorithm considers predefined frequencies, Prony Analysis determines the exact values for the modes (frequency, phase, etc.).

The Frequency Analysis Tool offers the possibility to apply Prony Analysis for a single *Time Point* or over a predefined *Time Range* (using a smaller sized sliding window sweeping between the start and end time references with a given step size).

---

**Note:** A **Time Point** is defined here and throughout Section 29.14 as a single set of time contiguous data points of a waveform. The set is delimited by a start and an end time. Each *Time Point* is called, in this context, a *Window*.

A **Time Range** is defined here and throughout Section 29.14 as a set of time contiguous data points of a waveform. The set is delimited by a start and an end time and may include a finite number of *Time Points/Windows*.

---

The result for a calculation at a single *Time Point* is presented in bar diagrams showing the amplitude, damping and energy of each mode, see Figure 29.14.1. Calculation of Prony Analysis over a given *Time Range* is useful to detect changes in harmonic, interharmonic or subsynchronous oscillations, see Figure 29.14.2. Both figures below show an example of a generator current in a 60 Hz power system in which a subsynchronous resonance is excited at  $t = 0.0\text{ s}$ . The subsynchronous resonance causes a current component with a frequency of 39.7 Hz (ca. 20 Hz below fundamental frequency). In addition there are current components with a frequency of 81.4 Hz (ca. 20 Hz above fundamental frequency) and higher frequencies, which are well damped and therefore do not impact the system.

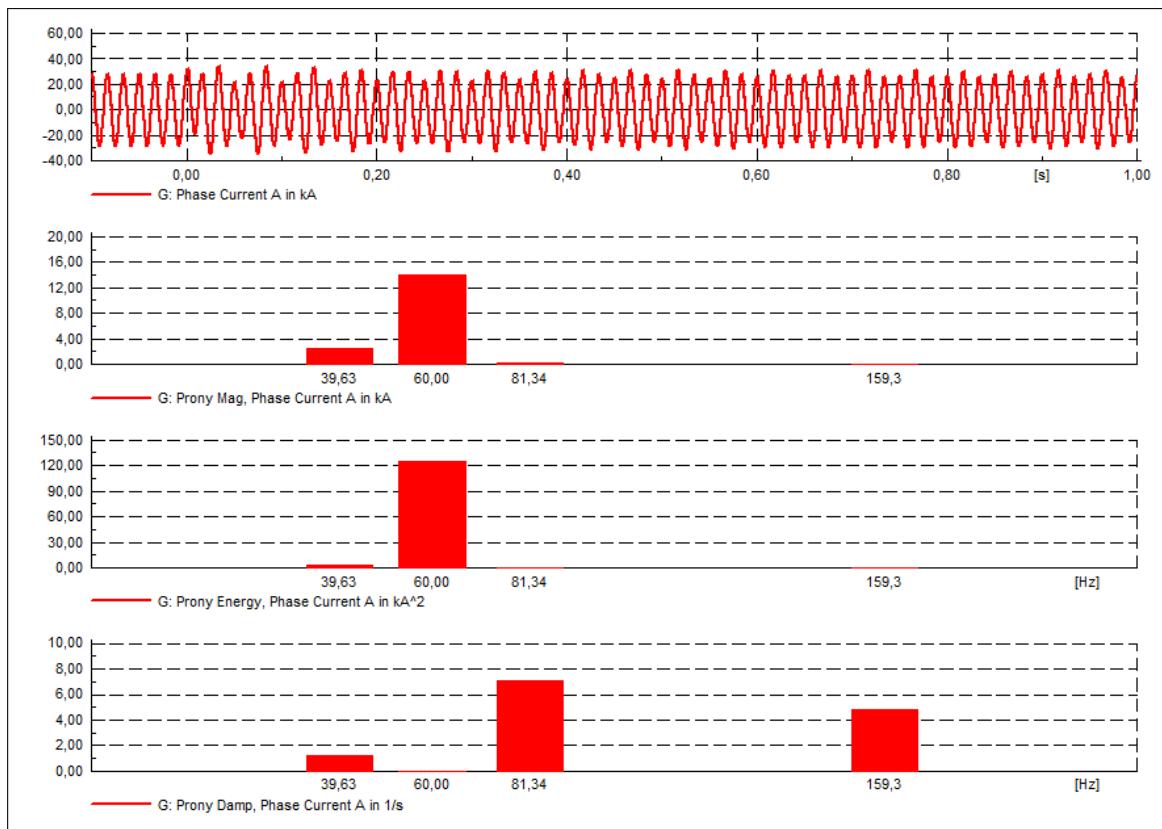


Figure 29.14.1: Prony Analysis results, three plots showing the magnitude, energy and damping of the modes of a given signal for a single *Time Point*

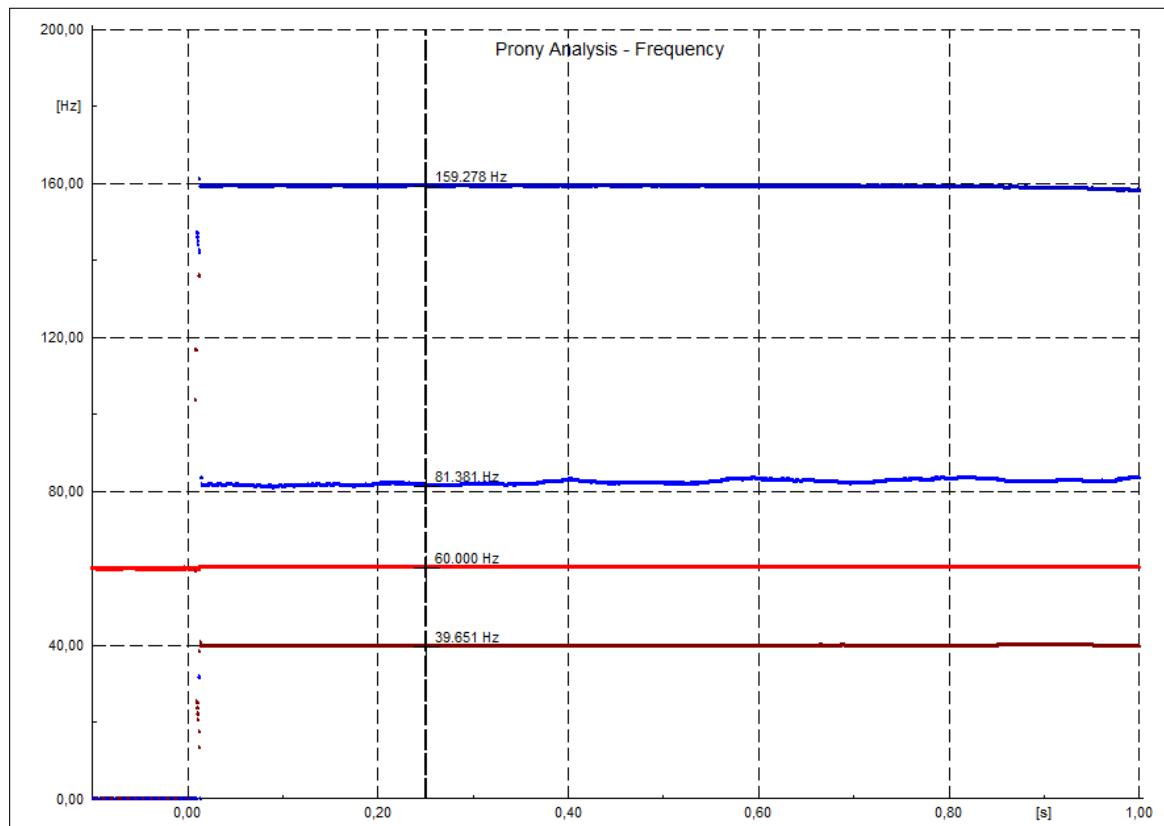


Figure 29.14.2: Prony Analysis calculated over a *Time Range*

### 29.14.2 Basic Usage

To perform a Frequency Analysis calculation several approaches are possible:

- Via the context menu:
  - Make sure that at least one signal is shown in a plot (*VisPlot*)
  - Right-click on any area of the subplot and from the context menu choose *Frequency Analysis*. . . . The mouse pointer will change appearance to  .
  - Move the mouse pointer to the time which corresponds to the starting point of the used data. Left click and hold while moving the mouse pointer to the time corresponding to the end point of the used data. Release the left click mouse button.
  - The Frequency Analysis command Dialog is shown.
- Via the *Plot* toolbar
  - Make sure that at least one signal is shown in a subplot (*VisPlot*) of a Plot Page (*SetVipage*)
  - Click on the Frequency Analysis icon  located on the Plots toolbar. The mouse pointer will change appearance to  .
  - Move the mouse pointer to the time which corresponds to the starting point of the used data. Left click and hold while moving the mouse pointer to the time corresponding to the end point of the used data. Release the left click mouse button.
  - The Frequency Analysis command dialog is shown.

The Frequency Analysis calculation settings are configured using the command dialog via three pages:

- Basic Options
- FFT
- Prony Analysis

A more detailed description of each page is provided in the following subsections.

After configuring the *Frequency Analysis* settings, click on **Execute** to carry out the calculation.

### 29.14.3 Basic Options Page

In the *Basic Options* page, the following settings are subject to user configuration:

**Calculation method** pane: The two calculation functions are made available via this pane:

- **Fast Fourier Transform (FFT)**
- **Prony Analysis**

**Range of calculation** pane: The Frequency Analysis command supports either the *Single time point* or the *Calculation over time* options.

- **Single time point:** This range of calculation is supported for FFT and Prony Analysis. As input data, a single *Time Point* is used.
- **Calculation over time:** This option is supported only by the *Prony Analysis* function. As input data, a *Time Range* is used.

---

**Note:** As a prerequisite, the time range chosen for calculation over time (difference between *End Time* and *Start Time*) must be larger than the corresponding time of the sliding window (the data corresponding for reading a number of *n* samples where *n* is defined by parameter **Size** available in the *Prony Analysis* page, Section 29.14.5).

**Definition of range:** If *Calculation over time* has been selected for *Prony Analysis* then the *Definition of range* pane is shown with the following settings:

- **Start time:** Data range being used for calculation is defined by a start and an end time. The *Start time* defines the point in time of the beginning of the *Time Range* with respect to the input variable (as defined in the “Input Curves” field).
  - **End time:** The *End time* defines the point in time corresponding to the last data point in the *Time Range*.
  - **Estimated number of calculations:** The number indicates an estimation of the total number of Prony Analysis calculations. This number depends mainly on the *Start time*, *End time* (Basic Options Page), the resampling and the *Step size (of sliding window)* (Prony Analysis Page) parameters.
- 

**Note:** In Prony Analysis , while moving (sliding) the *Window* over the defined *Time Range* with a certain step size, the *Calculation over time* will run a *Prony Analysis* for each *Window* inside the *Time Range*, as long as the *Window* fits inside the *Time Range*. For a trivial example, a signal is given with evenly distributed data points with a sampling rate of 100 Hz. For the *Prony Analysis* based on a *Calculation over time* the following options are set:

- Start Time = 0 s
- End Time = 1 s
- Size (of Window) = 40 samples
- No resampling
- Step size (of sampling window) = 20 samples

Based on this input data a number of 4 calculations will be executed on these *Windows*: (1) 0s-0.4s, (2) 0.2s-0.6s, (3) 0.4s-0.8s, (4) 0.6s-1.0s.

---

**Input curves:** Input signals can be defined in the table “Input curves”. Prony Analysis or FFT will be calculated for all curves with the same settings.

**Delete unused frequency calculation results...:** The *Frequency Analysis* creates a results file (*.ElmRes*) for each input curve and stores it in the subfolder “Signal Processing” of the currently active *Study Case* folder. Removing the plots generated by *Frequency Analysis* from the *plot page* does not automatically remove the previously generated results files. As a consequence, the “Signal Processing” subfolder might get cluttered with a lot of unused objects (e.g. not shown in a plot, not used by a DPL script, etc). *PowerFactory* keeps track of the results files which are currently in use by various subplots and provides the possibility to clean up the “Signal Processing” subfolder. To do so, click on the Delete unused frequency calculation results... button. A selection window containing all unused results files is shown. To remove all unused files, select all objects and click **OK**.

## 29.14.4 FFT Page

The **FFT** page is organised in two tabs: *General* and *Advanced*. The following settings of the *General* tab are subject to user configuration:

**Window:** The window pane configures the settings of the data used as input for the *FFT* calculation with reference to the original signal.

- **Size:** This parameter sets the number of data samples being used by the *FFT* calculation. This number fulfils the condition  $Size = 2^k$  where  $k$  is a positive integer. If a different number is typed in, then *PowerFactory* replaces it by the *power of 2* number closest to it.
- **Begin:** This parameter sets the starting time of the data set used for the *FFT* calculation.

- **End:** This parameter cannot be directly set. Being based on the window *Size* and *Begin* parameters, *PowerFactory* automatically displays the time of the last used data point.

**Fundamental Frequency:** Let  $n$  be the size of the time window, let  $f$  be the fundamental frequency, and let  $r$  be the sampling rate. Then, the following relation applies:

$$f = \frac{r}{n}$$

---

**Note:** In particular, if no resampling is selected, the algorithm uses the raw input signal and therefore the sampling rate is given. In this case, the size of window  $n$  and the fundamental frequency  $f$  depend on each other. In contrast, if the signal is resampled,  $n$  and  $f$  can be chosen freely, while the sampling rate  $r$  is adjusted according to equation above (i.e.  $f = r/n$ ).

---

**Resample signal (with linear interpolation):** Check this box to enable resampling of input data using linear interpolation.

**Sampling rate:** Provided that the **Resample signal (with linear interpolation)** checkbox is ticked, the sampling rate in Hz can be defined using this parameter.

---

**Note:** Resampling for FFT Analysis employs an equidistant resampling algorithm with linear interpolation of the given data. This might be particularly useful if the input data is not equidistant. In the case of subsampling, the Frequency Analysis results may be wrong due to aliasing effects.

---

The following settings of the *Advanced* tab are subject to user configuration:

**Representation (magnitude) pane:** This pane provides two options for computing the FFT magnitude results as described below.

- **Amplitude spectrum:** The magnitude will remain unchanged if the *Amplitude spectrum* is chosen.
- **RMS spectrum:** For currents and voltages, the setting *RMS spectrum* results in magnitude multiplication by  $1/\sqrt{2}$ .

**THD-Calculation** pane: A separate variable is stored in the results file representing the harmonic distortion (HD). This variable is calculated based on the options set in this pane, as described below.

The square-root of the sum of the squared HD values will give the total harmonic distortion (THD).

- **Based on RMS-values:** The variable contains magnitudes relative to the RMS value of the magnitude of the first 50 modes, i.e. relative to:

$$\sqrt{\sum_{i=0}^{50} Mag(mode_i)^2}$$

This result corresponds to the harmonic factor (HF) of the harmonic load flow calculation.

- **Based on fundamental frequency values:** The variable contains magnitudes relative to the magnitude of the mode corresponding to the fundamental frequency.

This result corresponds to the harmonic distortion (HD) of the harmonic load flow calculation.

## 29.14.5 Prony Analysis Page

The **Prony Analysis** page is organised in two tabs: *General* and *Advanced*. The following settings of the *General* tab are subject to user configuration:

**Window pane:** configures the sample *Size*, *Begin* and *End* times for a single *Time Point* used by the Prony Analysis calculation with reference to the original signal.

- **Size:** this parameter sets the number of data samples being used by the *Prony Analysis* in a single *Time Point*.
- **Begin:** if the *Range of calculation* is set to *Single time point* (refer to Section 29.14.3) then the **Begin** parameter is made available for editing. This parameter sets the starting time of the *Time Point*.
- **End:** if the *Range of calculation* is set to *Single time point* (refer to Section 29.14.3) then the **End** time is shown, calculated based on the window *Size*, *Begin* parameters and the sample rate.

**Number of modes:** The total number of computed modes. This parameter is specified on a case-by-case basis, considering the harmonic contents of the analysed signal. As a rule of thumb, being given a signal containing a number of  $k$  harmonics (including in  $k$  the fundamental frequency as well), the *Number of modes* should be chosen higher than  $2k + 2$ .

**Resample signal (with linear interpolation):** Check this box to enable resampling of input data using linear interpolation.

**Sampling rate:** Provided that the *Resample signal (with linear interpolation)* checkbox is ticked, the sampling rate in Hz can be defined using this parameter.

---

**Note:** Resampling for Prony Analysis employs an equidistant resampling algorithm with linear interpolation of the given data. This might be particularly useful if the input data is not equidistant. In the case of subsampling, the Frequency Analysis results may be wrong due to aliasing effects.

---

**Sort according to:** For calculation over time, sorting of the modes becomes necessary. The sorting can be done according to different criteria, such as energy or error impact. The modes are named mode 0, mode 1, ... according to the sorting algorithm. Modes with negative frequency are arranged at the end (usually, half of the modes have negative frequency, as the modes should come in complex conjugate pairs).

---

**Note:** Sorting affects the colouring of the plots as each colour in the plot corresponds to a mode.

---

When using the *Calculation over time* range of calculation, the resulting modes can be sorted according to the following calculation variables:

- Energy
- Error impact
- Frequency
- Amplitude
- Damping
- Phase

**Step size (of sliding window):** If calculation over time is performed, the *Step size* can be adapted, while the *Begin* parameter for the window is automatically set.

The following settings of the *Advanced* tab are subject to user configuration:

**Set frequency to zero if absolute value of frequency is small:** If the graphical representation is still not satisfactory, this might be due to the fact that the mode being the DC part, will have a frequency

which is near zero, but not exactly zero - possibly sometimes positive, sometimes negative. Therefore, as modes with negative frequency are arranged at the end, at some time points, the DC part is arranged at the end, at other points it is arranged as expected. This leads to undesired colouring effects in the plots. The settings provided on the advanced subpage of the Prony Analysis *Set frequency to zero...* will set the frequency to zero of all modes which have a frequency smaller than a given threshold. One must be aware that this option will change the results, thus might increase the fit error. On the other hand, the sorting of the modes should be more predictable with this setting enforced.

- **Threshold:** This parameter sets the threshold, in Hz, of the *Set frequency to zero...* option.

#### 29.14.6 Recalculation

The user may re-run an already performed calculation with slightly modified settings. For example, in order to find good calculation parameters, the user may modify just a few settings, and leave the rest as it is. In this case:

- Double click the plot where the frequency analysis results are displayed.
- A button **Recalculate** allows to open the *Frequency Analysis* command with all the previously used settings for the calculation of the specific *Frequency Analysis* results. These settings are stored in the previously generated results file (*ElmRes*).
- After adjusting the settings as required, the *Frequency Analysis* can be executed again.

---

**Note:** By default the old results file is overwritten. The user however can choose to generate a new results file on the *Basic Options* page.

---

#### 29.14.7 Output Plots

Upon executing the *Frequency Analysis* command, new plots are generated automatically and shown on a new page together with the original signals. Depending on the command settings, along with the original signals, the following plots are presented:

- FFT Analysis:
  - FFT magnitude plot (magnitude over frequency)
  - FFT phase plot (phase over frequency)
- Prony Analysis (*Single time point*):
  - Mode Magnitude plot
  - Mode Damping plot
  - Mode Energy plot
- Prony Analysis (*Calculation over time*) - For each input curve, two pages with five plots are generated:
  - Page 1: Signal to noise ratio (SNR) plot
  - Page 2: Mode Frequency plot
  - Page 2: Mode Damping plot
  - Page 2: Mode Amplitude plot
  - Page 2: Mode Energy plot

## 29.14.8 Output Results Files

Calculation results are stored in *Results Objects (ElmRes)* under the “Signal processing” subfolder of the active *Study case* folder. Each *Frequency Analysis* of a curve leads to a new results object.

There are three types of results objects:

- Prony Analysis (time range) (stores results of Prony Analysis using *Calculation over time*)
- Prony Analysis (stores results of Prony Analysis using *Single time point*)
- FFT Analysis

For all above *Frequency Analysis* functions, the calculation settings are saved and stored in the corresponding results file. This is particularly useful in case of performing a **Recalculate** action or in order to document the applied settings for a particular result.

### Calculated variables: Prony Analysis using *Calculation over time*

The x-Axis variable is:

- *time*: Time

The calculation variables per mode are listed below:

- *freq*: Frequency
- *damp*: Damping
- *mag*: Amplitude
- *phi*: Phase
- *complConj*: Complex conjugate mode (-1, if no complex conjugate exists)
- *errImp*: Error impact calculated as in (29.1).
- *energy*: Energy of the mode (refer to (29.2) and (29.3)).

The calculation results for the whole analysis are:

- *diffSup*: Maximal error to input series (refer to (29.4)).
- *errRms*: RMS error to input signal.
- *snr*: Signal to noise ratio. (refer to (29.5); for voltages or currents, the prefactor 10 is replaced by 20 in (29.5).)

### Calculated variables: Prony Analysis using *Single time point*

The variable on the x-Axis is:

- *freq*: Frequency

The calculation results per mode are:

- *damp*: Damping
- *mag*: Amplitude
- *phi*: Phase
- *complConj*: Complex conjugate mode (-1, if no complex conjugate exists)
- *errImp*: Error impact calculated as in (29.1).
- *energy*: Energy of the mode as in (29.2) and (29.3)

Calculation results for whole analysis: (same in each row)

- *diffSup*: Maximal error to input series as in (29.4).
- *errRms*. RMS error to input signal.
- *snr*. Signal to noise ratio as in (29.5).

### Calculated variables: *FFT Analysis*

The variable on the x-Axis is:

- *fnow*: Frequency

The calculation results per harmonic order are:

- *FFT\_mag*: Amplitude
- *FFT\_phi*: Phase
- *FFT\_re*: Real part
- *FFT\_im*: Imaginary part
- *FFT\_HD*: Harmonic Distortion
- *FFT\_PSD*: Power Spectrum Density

**Variable naming conventions:** *Prony Analysis* generates “user defined variables” in the results files as described above. For example, *Prony Analysis* using *Calculation over time* creates the following result variables:

- *mode0:freq(inputsignal)*. Refers to frequency of mode 0 of Prony Analysis of series “inputsignal”.
- *snr(inputsignal)*. Refers to *snr* of Prony Analysis result of series “inputsignal”.

Variables generated from ComTrade files data are treated in an analogous way.

### 29.14.9 General Recommendations on the Use of *Prony Analysis*

*Prony Analysis* is, as many other frequency analysis tools are, sensitive to the applied input signal and configuration settings. Validating the obtained results may prove to be an iterative process where the calculation performance indices are evaluated. Further on, corrections to the used settings are applied and the calculation is redone in order to maximise the quality of obtained results. The following indices should be observed after performing a *Prony Analysis*:

- *snr* - Signal to noise ratio. This performance index supplies information on the quality of the calculated signal  $\hat{y}(t)$  (based on the decomposition results) as compared with the original signal. A high value of the *snr* indicates a good correlation between the calculated signal  $\hat{y}(t)$  and the original signal  $y(t)$ .
- *diffSup* - Maximal error to input series. Small values indicate a good fit.
- *errRms* - RMS error to input signal. Small values indicate a good fit.

Determining which modes are relevant to the calculated signal is done by evaluating the following results:

- *energy* - Energy of the mode. This variable provides information on the energy contents of the specific mode. A relative comparison between the energy of each calculated mode can suggest whether a specific mode is relevant for the calculated signal or not. Modes with less energy may be regarded as fictitious modes. Typically these modes have a low *amplitude* as well.
- *errImp* - Error impact. A small error indicates important modes.

Consider the example of analysing the phase currents of a thyristor based rectifier (within a HVDC station) before, during and after a network voltage transient (short circuit on an AC bus nearby). In this example, only one performance index is optimised (signal to noise ratio *snr*). The resulting current waveform (for one phase), i.e. the original signal is shown in Figure 29.14.3.

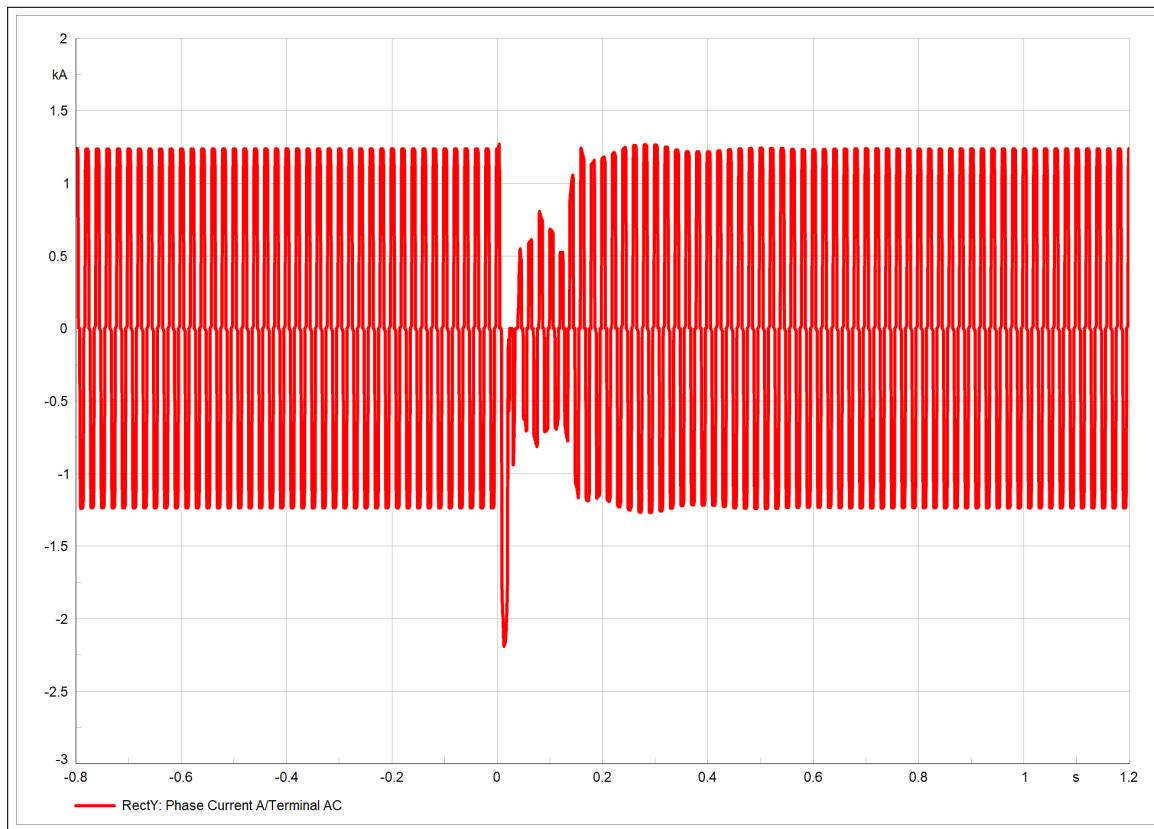


Figure 29.14.3: Original signal (AC phase current) to be processed by Prony Analysis

The input curve has a sample rate of approximately 20 kHz, unevenly spaced. For the *Prony Analysis* based on a *Calculation over time* the following options are set:

- Start Time = -0.8 s
- End Time = 1.2 s
- Size (of Window) = 50 samples
- Resampling at 5 kHz (generating evenly spaced data)
- Step size (of sampling window) = 20 samples
- Number of modes = 12

Based on this input data the *Prony Analysis* is executed, with the *snr* represented by the red curve in Figure 29.14.4. The signal to noise ratio *snr* is low (between 5 and 13 dB). The *snr* deteriorates further during the transient (it consistently drops in the 0dB range), which is to be expected, as the parameterisation algorithm does not fit well the calculated signal to the original one especially when transients appear (a large band frequency spectrum is present). Improving the calculated curve implies tuning the calculation settings as much as possible. As one solution, the following changes are applied:

- Start Time = -0.8 s
- End Time = 1.2 s
- **Size (of Window) = 100 samples** - changed to fit into one fundamental frequency (50Hz)

- Resampling at 5 kHz
- **Step size (of sampling window) = 50 samples** - changed to lower the total number of calculations (it has no practical effect on improving the results themselves)
- **Number of modes = 24** - enables the calculation to estimate the calculated signal based on a higher number of modes, hence increasing the accuracy by reproducing a larger number of frequency components of the original signal.

Based on this input data the *Prony Analysis* is executed once more, with the *snr* represented by the green curve in Figure 29.14.4. A visible improvement is observed in both the steady state and the transient ranges. Now, all other results of the individual modes can be confidently used further in the analysis. Note that there are no standardised limits on the level of minimum *snr* for validating results, moreover, the user being able to specify own requirements of pass/fail.

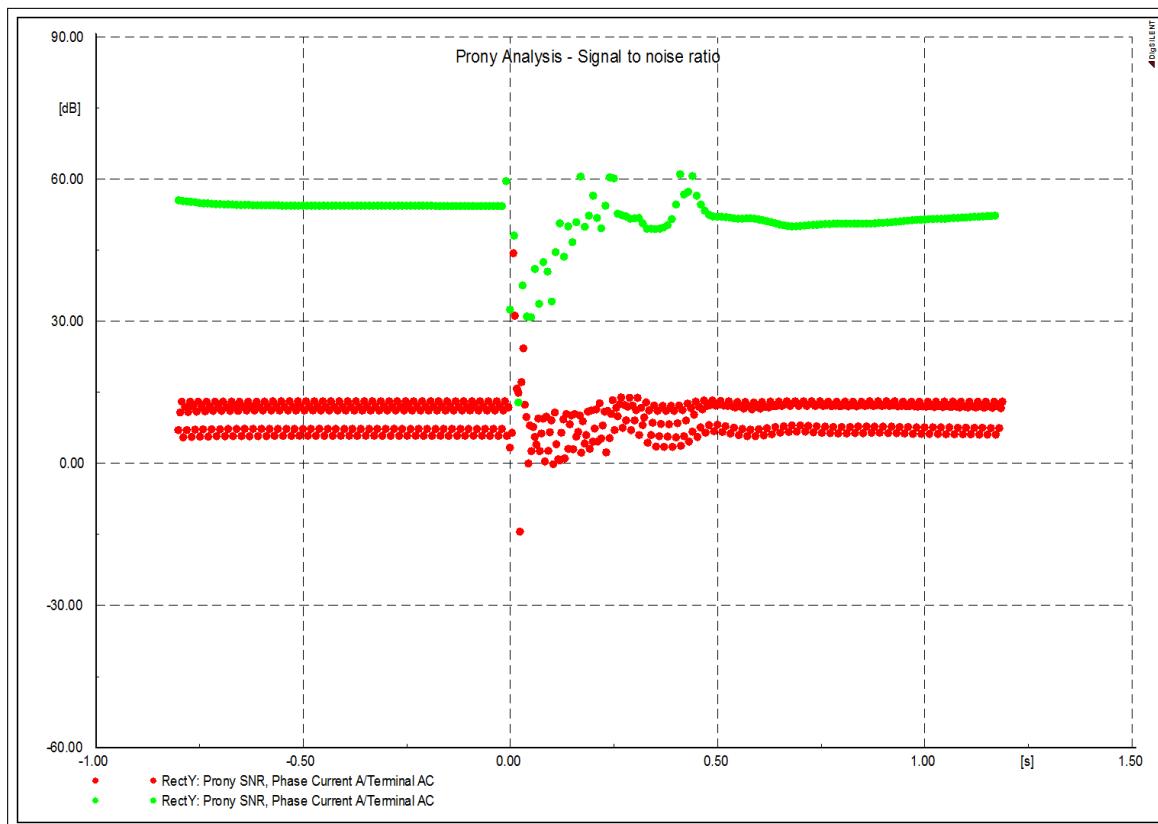


Figure 29.14.4: Evaluation of signal-to-noise ratio for different Prony Analysis settings

The final optimised results are shown in Figure 29.14.5. The first 6 modes sorted according to their energy are shown (the option *Sort according to* has been set to *Energy*). The typical low frequency harmonic current spectrum of the rectifier is observed. The 17th and the 19th harmonics are not displayed as the energy contained is too low. Furthermore, the damping, amplitude and energy of each mode of oscillation are quantified providing valuable insight into the actual behaviour of the rectifier unit (along with its controllers).

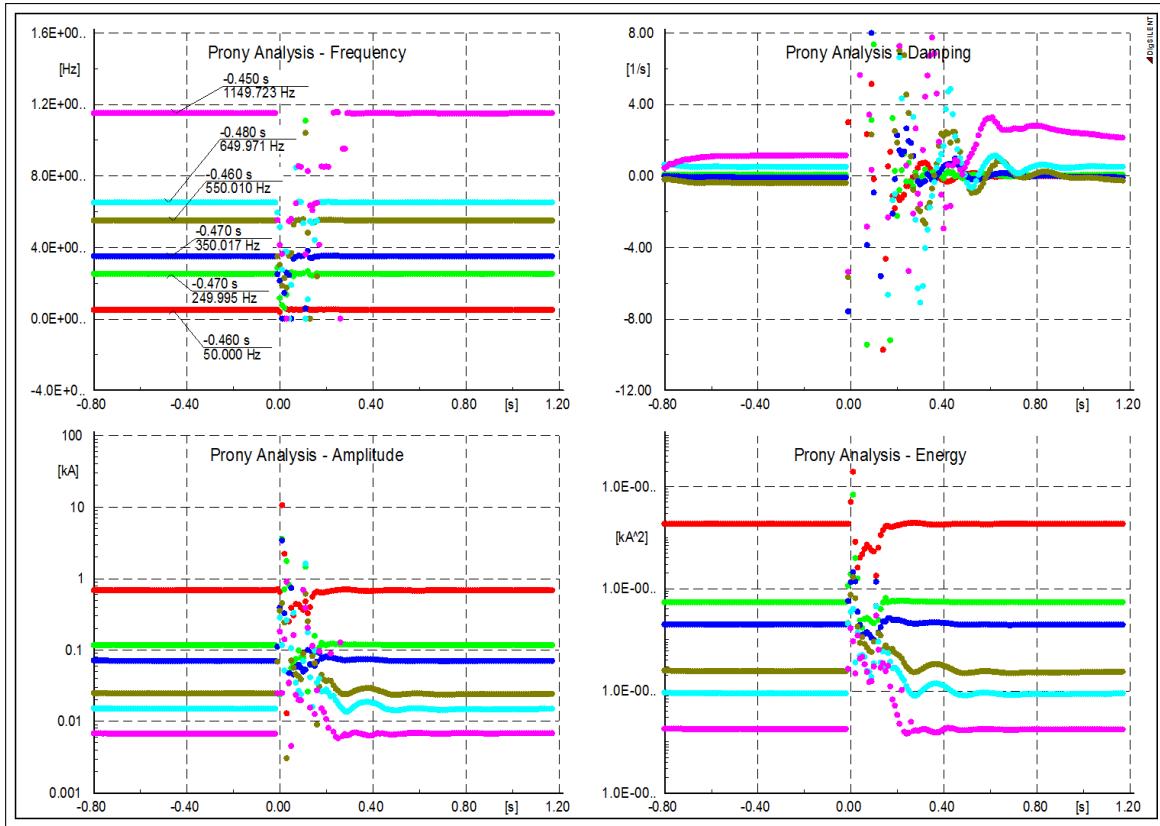


Figure 29.14.5: Prony Analysis on AC phase current of a thyristor rectifier before/during/after a three phase short circuit

### 29.14.10 Quick Overview of Used Formulas

Throughout the *Frequency Analysis* command, a series of formulas are relevant for understanding the various result variables. The goal of *Frequency Analysis* is to represent an input signal as a sum of modes, where  $y(t)$  is the original signal:

$$y(t) \approx \sum_{k=0}^{n-1} \text{mode}_k(t).$$

In the case of FFT Analysis:

$$\text{mode}_k = \text{mag} \cdot e^{i\phi_k} \cdot e^{it2\pi f_k}.$$

In the case of Prony Analysis:

$$\text{mode}_k = \text{mag} \cdot e^{i\phi_k} \cdot e^{it \cdot (\text{damp}_k + 2\pi f_k)}.$$

Letting  $\hat{y}(t)$  denote the calculated signal, and  $\hat{y}_{-l}(t)$  the calculated signal where  $\text{mode}_l$  is removed:

$$\hat{y}_{-l}(t) = \sum_{k \neq l} \text{mode}_k(t).$$

The error impact of a mode  $\text{mode}_l$  is defined as:

$$\text{errImp}_l := \sqrt{\int (y(t) - \Re(\hat{y}_{-l}(t)))^2 dt}. \quad (29.1)$$

The energy of a mode which has no complex conjugate in the representation of  $y(t)$  is defined to be:

$$\text{energy}_k := \int |\Re(\text{mode}_k(t))|^2 dt \quad (29.2)$$

The energy of a mode which has a complex conjugate in the representation of  $y(t)$  is defined to be:

$$\text{energy}_k := \int |2\Re(\text{mode}_k(t))|^2 dt \quad (29.3)$$

In general, the input signal will differ from the calculated representation:

$$\text{diffSup} := \sup_{t \in \text{Window}} |\hat{y}(t) - y(t)| > 0 \quad (29.4)$$

Another possibility to evaluate correctness is to determine the signal to noise ratio as calculated below:

$$a := \frac{\sqrt{\int \hat{y}(t)^2 dt}}{\sqrt{\int (y(t) - \hat{y}(t))^2 dt}}$$

In many cases, the signal is relatively big with respect to noise. Therefore, it might be useful to consider the signal to noise ratio on a logarithmic scale as below. This quantity is without dimension, however it is defined to be in Decibel.

$$\text{snr} := 10 \log(a) [dB] \quad (29.5)$$

As the input data of the input signal comes in discrete measurements, almost all integrals above are evaluated as sums.

## Chapter 30

# Models for Dynamic Simulations

This Chapter is organised in the following sections:

- Section 30.1 - System Modelling Approach - An overview of dynamic modelling with *PowerFactory*
- Section 30.2 - High-level Control System Representation - A unified approach to creating interconnected dynamic models
- Section 30.3 - DSL: Integrating *DSL Models* into a Simulation
- Section 30.4 - DSL: The *DlgSILENT* Simulation Language - An easy to use dynamic systems modelling language tailored for power systems simulation
- Section 30.5 - DSL: Overview of the *DSL Model Type*
- Section 30.6 - DSL: Creating *DSL Model Types* using Block Diagrams
- Section 30.7 - DSL: Coded *DSL Model Types* (non-graphically defined)
- Section 30.8 - Modelica: Integrating Modelica Models into a Simulation
- Section 30.9 - Modelica: a non-proprietary, object-oriented, equation based language
- Section 30.10 - Modelica: Creating Modelica Models using Block Diagrams
- Section 30.11 - Modelica: Creating Modelica Models using Code
- Section 30.12 - Modelica: Overview of the *Modelica Model Type*
- Section 30.13 - Modelica: Overview of the *Modelica Model*
- Section 30.14 - Interfaces for Dynamic Models
- Section 30.15 - Developing User-defined Power Electronics Models for EMT Simulation
- Section 30.16 - DSL: Technical Reference

## 30.1 System Modelling Approach

### 30.1.1 Overview of dynamic models in *PowerFactory*

In the field of power system analysis, there exists a number of approaches for the representation of dynamic models which roughly specify the way a simulation model is realised, packaged and used within one or multiple simulation environments. These approaches depend largely on various criteria, such as:

- the nature of the dynamic system (e.g. electrical, mechanical, control; time continuous or digital);
- adequacy of represented internal model functionality (ranging from basic to complex functions) w.r.t. analyses in which the model is used;
- accuracy of simulation results w.r.t. reference models/equipment/data;
- the simulation domain (e.g. electromechanical transient i.e. RMS, electro-magnetic transients i.e. EMT);
- the model specification source (manufacturer-based, standardised or other source);
- the level of model obfuscation required (e.g. open-source, compiled code, encrypted information);
- the intended range of operation scenarios (e.g. small signal or large signal disturbances).

Given the above criteria, dynamic models in *PowerFactory* can be categorised as follows (see Figure 30.1.1):

- Availability within *PowerFactory*: *Built-in* / *User-defined*;
- Model's relation with *PowerFactory*: externally interfaced / *PowerFactory* native;
- Model's native modelling language: *DIGSILENT* Simulation Language (DSL) / Modelica Language;
- Model's external interface type: Functional Mock-up Interface (refer to Section 30.14.1) / IEC 61400-27 (refer to Section 30.14.2).

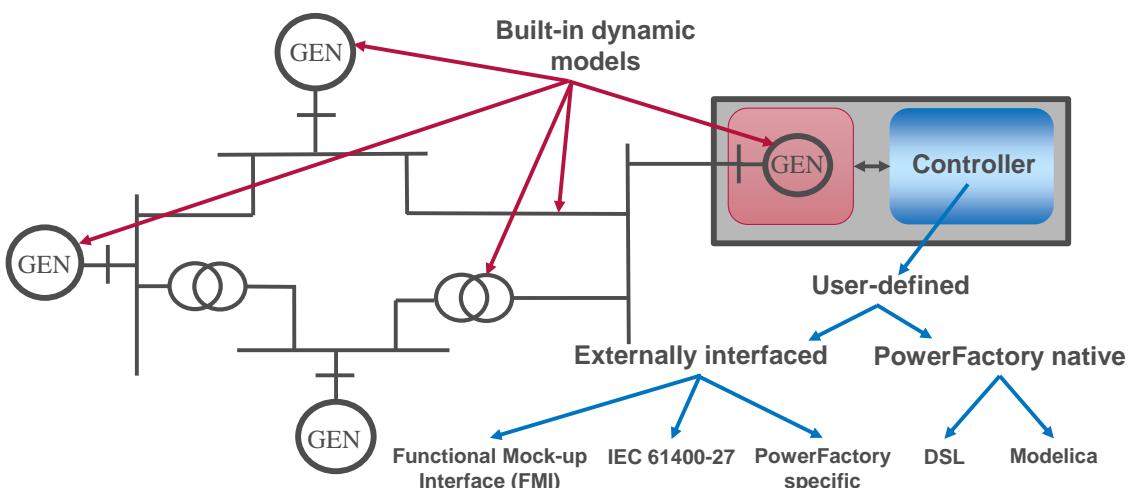


Figure 30.1.1: Types of dynamic models in *PowerFactory*

### 30.1.2 Model availability within *PowerFactory*: *Built-in* or *User-defined* models

For many commonly used dynamic components (e.g. a synchronous machine), *PowerFactory* supplies *Built-in* representations for RMS- and EMT-domain simulations. *Built-in* dynamic models are generally available for any of the *PowerFactory* network elements, and are thoroughly described within the *Technical Reference Documentation* (from *PowerFactory* click *Help* → *Technical References* → *Models*). A *Built-in* dynamic model denotes typically one power system component which is readily provided by *PowerFactory* and has the following properties:

- shipped together with the *PowerFactory* simulation environment;
- self contained - does not require other external components in order to have it functional;
- packaged within a built-in *PowerFactory* object(s) e.g. for a synchronous machine component the relevant objects are the *Synchronous machine element* (*ElmSym*) and the *Synchronous machine type* (*TypSym*);
- parameterisable based on the built-in functionality;
- non-editable - not possible to make modifications to the internal model structure ;
- Data containment - the dynamic model is packaged internally within the *PowerFactory* data model and is included within a *PowerFactory* project export/import process.

*Built-in* dynamic models are available for almost all network components, for example:

- synchronous and asynchronous machines;
- PWM converters, line commutated-converters;
- passive network components such as lines, transformers and cable systems;
- general purpose loads, etc.

On the other hand, there exists a large class of dynamic models whose structure does change depending on any of the aforementioned criteria. Therefore, the term *User-defined Models* is introduced to characterise those dynamic models whose internal realisation, not known *a priori* by the simulation environment, is required to be implemented with a high degree of customisation capability. A large part of this Chapter is dedicated to building or integrating *User-defined Models* using *PowerFactory*. The terms *Built-in* and *User-defined* are therefore used to refer to the *PowerFactory* implementation nature of a dynamic model. Examples of *User-defined Models* are: power electronics converter controllers, voltage regulators of synchronous machines, fault-ride through blocks of renewable generators.

A *User-defined* dynamic model denotes one power system component which is typically developed externally to the *PowerFactory* package and may have the following properties:

- Custom-made model structure - the internal model structure is created, modified and delivered by a *PowerFactory* user (known as the *Model developer*).
- Open model access - the model structure and implementation can be freely accessed by any third party user (known as the *Model user*). The *Model developer* has certain means of restricting/obfuscating parts of the developed model structure, depending on the choice of native modelling language. If the model is *Externally interfaced* then the model access may be restricted e.g. by using only compiled code, when sharing model data.
- Graphical model representations - well-known signal block diagrams may be supported by *User defined models*, depending on the choice of native modelling language and/or implementation.
- Data containment - the dynamic model is packaged within the *PowerFactory* data model and is included within a *PowerFactory* project export/import process. *User-defined Models* based on native modelling language can be completely contained within the *PowerFactory* data model. *User-defined Models* based on externally interfaced components (e.g. external DLL files) are not completely contained within the *PowerFactory* data model.

*PowerFactory* supplies a flexible framework for developing *User-defined Models*, mainly based on the *DlgSILENT* Simulation Language (DSL), and alternatively on Modelica Language. *Model developers* are able to design customised dynamic models for various systems, typically in the area of control engineering e.g. controllers for electrical generators. Full flexibility is possible in the model design/development phase, process described in Sections 30.6 (for DSL) and 30.11 (for Modelica).

*PowerFactory* contains a broad range of *User-defined* dynamic models within the “*DlgSILENT Library*”. Refer to the Technical Reference: [Synchronous Generator Power Plants Reference](#) for a description of the standard models available in the *DlgSILENT* library. The available library contains a large set of commonly used dynamic models, such as:

- standard IEEE models for excitation voltage controllers, prime movers and other associated control units commonly used for conventional generation;
- standard WECC models for stability analyses (e.g. renewables, SVS, load, HVDC);
- standard IEC models for stability analyses (e.g. wind turbine models acc. to IEC 61400-27);
- standard ENTSO-E models for stability analyses;
- *DlgSILENT* generic models for HVDC, PV, FACTS, Grid-forming Converters, Battery Systems, etc.

For each of the above dynamic models located in the *DlgSILENT Library*, the benefits of using the *User-defined* modelling environment are multiple:

- *Model users* have access to the internal structure of these dynamic models;
- *Model users* can copy locally and further develop the available models for their own needs;
- Implementation is transparent - unlike built-in models, the model equations are readily available and the model behaviour can be understood in detail.

### 30.1.3 Externally interfaced versus *PowerFactory* native models

While depending on many criteria (e.g. the type of analysis, the availability of model information, the *Model developer* modelling preferences, end-user access requirements), *User-defined* dynamic models may exist as *externally interfaced* or as *PowerFactory native* dynamic models.

Externally interfaced models are typically generated using a third-party simulation modelling tool and are coupled with *PowerFactory* via a set of input/output signals. These models typically follow one of the existing standardised interfaces for exchanging models (e.g. FMI Standard, IEC 61400-27). Alternatively, a number of *PowerFactory* specific external interfaces are supported (e.g. *DSL C Interface*) Model parameterisation structures may be transferred across the model interface for the purpose of parameterising the dynamic model from within *PowerFactory*. Externally interfaced models require at the interface boundary a specific link with the *PowerFactory* environment. The use of these models typically covers the representation of detailed control equipment, which has already been designed in a third-party simulation tool. The externally interfaced models may contain any internal model functionality, the model's realisation being independent of *PowerFactory*, up to the interface boundary, where the interface specifications must strictly be complied with.

*PowerFactory* native dynamic models are models which are completely defined within *PowerFactory* either using *DlgSILENT* Simulation Language (DSL) or the Modelica Language. *PowerFactory* native dynamic models are based on an *Interpreted* model, which defines the model's equations in an open format (either using model code, block diagrams or a combination of these two). For various reasons (e.g. performance improvement, further obfuscation of model's internal structure), *Interpreted models* can be compiled using specific interfaces (e.g. from a *DSL Model* to a *DSL C Interface* based model - refer to Section 30.14.3 for more information). For simplicity, whenever a *PowerFactory*-specific or a standardised interface is employed, then such models are regarded as *Externally interfaced* models.

For example, all *User-defined* models found in the DLgSILENT Library are available either as *PowerFactory* native dynamic models or as *Externally interfaced* models using the *DSL C Interface*. Files corresponding to the *DLgSILENT* Library compiled models are supplied with *PowerFactory*.

### DSL versus Modelica dynamic models

*PowerFactory* native dynamic models can be implemented using either DSL or Modelica. Choosing one or the other depends on the *Model developer*, a choice which largely decides the way *PowerFactory* processes these models.

### Functional Mock-up Interface versus IEC 61400-27 interfaced models

An externally interfaced model can be integrated with a *PowerFactory* simulation if the interface is compliant with one of the following interfaces:

- Functional Mock-up Interface (FMI) (see Section [30.14.1](#))
- IEC 61400-27 Interface (see Section [30.14.2](#))

Each of the available interfaces is discussed in the corresponding sections.

### 30.1.4 Complete *Power Equipment* simulation models

A complete *Power Equipment* simulation model is a relatively complex data set that needs to provide specific *PowerFactory* functionality for the purpose of correct model operation under prescribed conditions. The functionality may range from load flow calculation to RMS- and/or EMT-Simulation and may be integrated within either *Built-in* or *User-defined* models. Such a complete *Power Equipment* simulation model is composed of a large set of *PowerFactory* objects, each of these objects being coordinated with the others.

To simplify the grouping, packaging and data containment of complete *Power Equipment* simulation models, the concept of *General Templates* is introduced in *PowerFactory*. A *General Template* is able to pack all objects belonging to a larger simulation model (including the *Built-in* and the *User-defined* components) in order to facilitate easy data management, fast model deployment and worry-free sharing of such complex models. Examples of *Template* models are provided in the *Global Library* (under *DLgSILENT Library* → *Templates*), where a large set of various models can be easily deployed and simulated in a *PowerFactory* project.

The use of *General Templates* as a means to create, use and share complete *Power Equipment* simulation models is common in *PowerFactory* especially in the case of manufacturer specific simulation models. Further information on the creation and use of *General Templates* is provided in Section [14.9.1](#).

### 30.1.5 Compilation of *PowerFactory* native dynamic models

#### 30.1.5.1 C\C++ Compiler - availability and prerequisites

*PowerFactory* requires that a C\C++ Compiler is available in order to compile native dynamic models.

The *PowerFactory* installation provides an option (set by default) for including a freely available MinGW C-Compiler. This compiler allows users to compile *PowerFactory* native dynamic models directly from *PowerFactory* without the need to instal any additional third party software. If another compiler is to be used, it can be configured separately. For more information on how to choose and configure the C-Compiler within *PowerFactory*, refer to Section [5.2.4](#).

The availability of a C-compiler in *PowerFactory* (either MinGW or any other supported compiler), impacts the capability of RMS- and EMT- simulations to support *PowerFactory* native dynamic models, as follows:

- **DSL models:** In order to run power system simulations including DSL models, a C-Compiler is not mandatory. Nevertheless, it is recommended to include a C-Compiler in order to benefit from the increased simulation performance provided by the *Automatic compilation* function (refer to Section 30.1.5.2 for more information);
- **Modelica models:** In order to run simulations including *PowerFactory* Modelica models, a C-Compiler is required. It is therefore mandatory that the *PowerFactory* installation includes the integrated C-Compiler or a third party C-Compiler is configured (refer to paragraph “C/C++ Compiler” within Section 5.2.4).

### 30.1.5.2 Compilation of DSL models

DSL models can be compiled either manually or automatically:

- Manual compilation is described in Section 30.14.3;
- Automatic compilation is described here.

*PowerFactory* is capable of automating the process of model compilation, enabling users to benefit from improved simulation performance without the need to manually compile individual DSL models.

The “Automatic compilation” option for DSL models is included in the “Calculation of initial conditions” command (refer to Section 29.3.3 for more details). With this function, the initialisation command can automatically check for compilation and compile all currently non-compiled DSL models of level 5 or higher. The functionality is effective only at run-time, so users can still apply modifications to the existing DSL models, even if *PowerFactory* has already generated compiled code for them. *PowerFactory* automatically monitors the state of the DSL models as well, meaning that any subsequent simulation runs on an unchanged DSL model will not trigger a re-compilation. Conversely, if any modification has been applied to a DSL model then a re-compilation is triggered for that specific model.

### 30.1.5.3 Compilation of Modelica models

*PowerFactory* requires that Modelica models are pre-compiled before executing a dynamic simulation. The process can be achieved as follows:

- manually, before run-time (i.e. before the execution of the *Calculation of Initial Conditions* command) - by setting the Modelica Model Type (*TypMdl*) to *Compiled* and assigning a corresponding file path pointing to the compiled Modelica model file (.pfmu)
- automatically, at run-time (i.e. during the execution of the *Calculation of Initial Conditions* command) - for each calculation-relevant Modelica Model Type (*TypMdl*) which has not been manually compiled. Each calculation-relevant Modelica model type (*TypMdl*) must successfully compile in order for the initial conditions command to execute successfully.

---

**Note:** Calculation-relevant Modelica model types are those *TypMdl* objects referenced by an “in service” Modelica Model (*ElmMdl*), currently being employed in the simulation.

---

### Consistency of Modelica Models

During simulation, the behaviour of the Modelica models is largely determined by the compiled file. If the same model is recompiled in a later version of *PowerFactory*, the newly compiled version will incorporate improvements and optimisations introduced in that version. These changes can result in modified and improved dynamic behaviour. To reduce the risk of a model’s behaviour changing between versions of *PowerFactory* - without being affected by future compiler enhancements - it is recommended to manually compile the model after it is finalised and always use this compiled model. The handling and distribution of compiled models can be conveniently managed using the PFDX format (refer to Section 9.1.5.1).

**30.1.5.4 Management of compiled files of *PowerFactory* native dynamic models**

To perform various actions on relevant compiled files, please refer to Section [30.14.4](#).

## 30.2 High-level Control System Representation

### 30.2.1 Data Structures for Dynamic Models within *PowerFactory*

From a data management perspective, data of dynamic models follow the same type/element concept that exists throughout *PowerFactory*. That is, a clear separation is enforced between instance-independent and instance-specific information. As such, dynamic data are separated in a *PowerFactory* project as conceptually depicted in Figure 30.2.1.

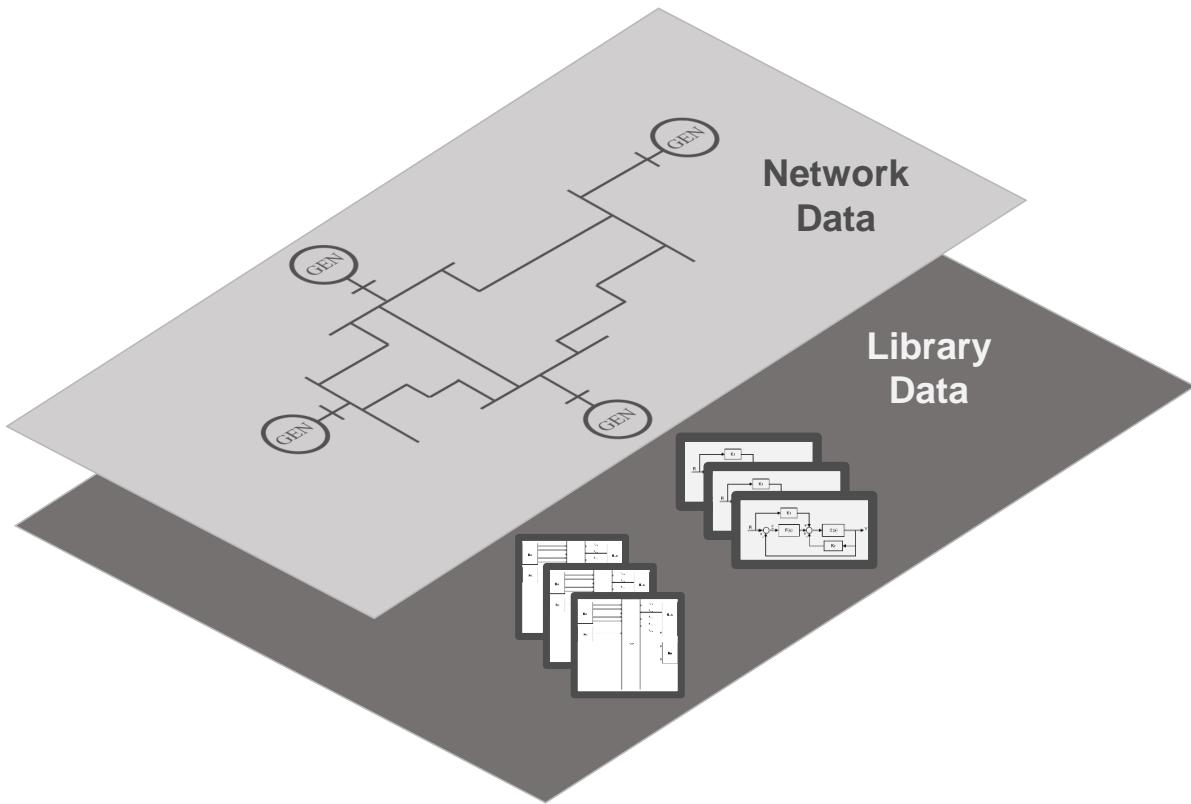


Figure 30.2.1: Dynamic models data separation between *Library Data* and *Network Data*

Given the complexity of a large scale power-system analysis problem, the *PowerFactory* modelling philosophy is targeted towards a strictly hierarchical system modelling approach, which combines both graphical and script-based modelling methods.

### 30.2.2 Composite Model Frames and Composite Models

A high-level model definition and signal interconnection layer is defined using two *PowerFactory* specific concepts: *Composite Model Frames* (*BlkDef* ) and *Composite Models* (*ElmComp* ). As conceptually shown in Figure 30.2.2, an arbitrary number of *Composite Model Frames* objects can be defined in the *Library*.

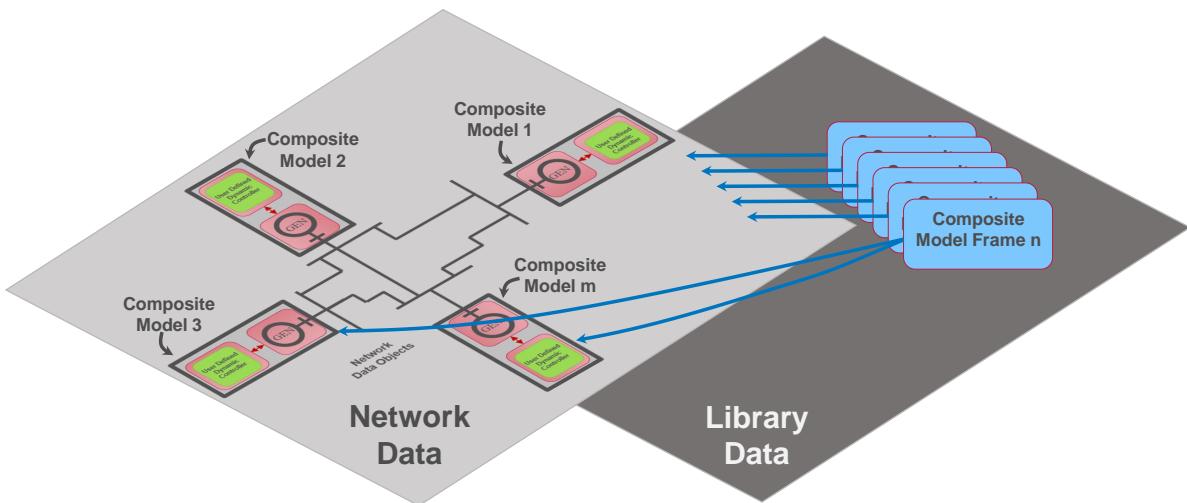


Figure 30.2.2: *PowerFactory* objects for implementation of high-level control systems

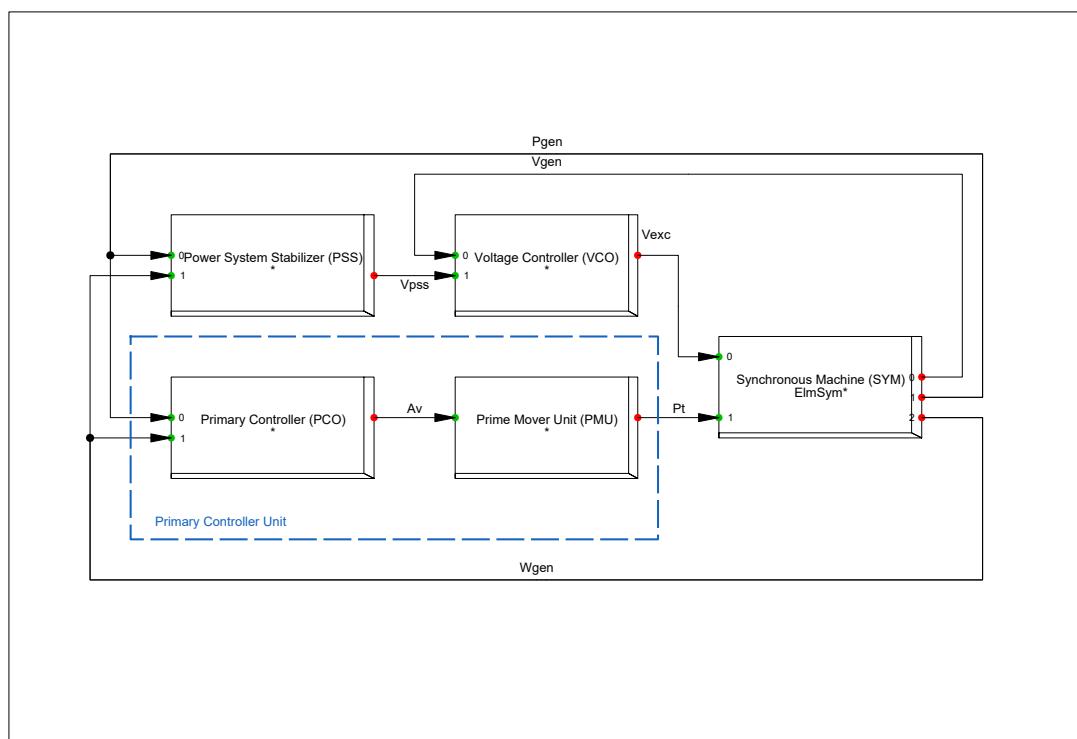
Each *Composite Model Frame* (*BlkDef* ) graphically defines a high-level control system signal flow diagram, by means of a combination of *Slots* and interconnecting signals. Each *Slot* represents a placeholder for corresponding network elements having a well defined functionality within this high-level control system. Typically, slots are placeholders for controlled network elements (e.g. synchronous machines, PWM converters, static generators), measurement devices (e.g. current or voltage measurements), user-defined equipment controller components (e.g. excitation voltage regulators, power system stabiliser units) or user-defined representations of physical components (e.g. turbine mechanical shaft systems).

A *Composite Model* (*ElmComp* ) represents “an instance” in the power system of one specific library *Composite Model Frame*. The graphical definition of the *Composite Model Frame* is read by the referencing *Composite Model* and, for each slot within the frame’s diagram, a network element can be assigned at this stage. Any number of *Composite Models* can be created on the basis of a single *Composite Model Frame*, thus allowing a large number of similarly controlled power system components to share the same high-level control structure. For example, to a large number of synchronous generating units (e.g. “m”), a controller may be attached to each one by creating “m” *Composite Models* that reference the same library located *Composite Model Frame*, frame which is designed for linking synchronous generators with corresponding synchronous machine controllers (e.g. voltage regulators) via pre-designated signal connections.

**Note:** The *Composite Model* and *Composite Model Frame* objects do not hold information about equipment controllers, they only link such controllers with network elements and other measurement devices based on the graphical diagram of the *Composite Model Frame*. The “controller” elements are discussed/introduced further below.

A simple example for the high-level control system diagram of a synchronous generator is shown in Figure 30.2.3. The diagram contains rectangular blocks (i.e. slots) interconnected between each other using routed signal lines. Signal lines are directional and thus define the signal flow during a dynamic simulation (as signal transfer from one slot to another). Individual slots are named based on the functional scope within the high-level control diagram.

CompositeModelFrame:

Figure 30.2.3: Example of a simple *Composite Model Frame* for conventional generation

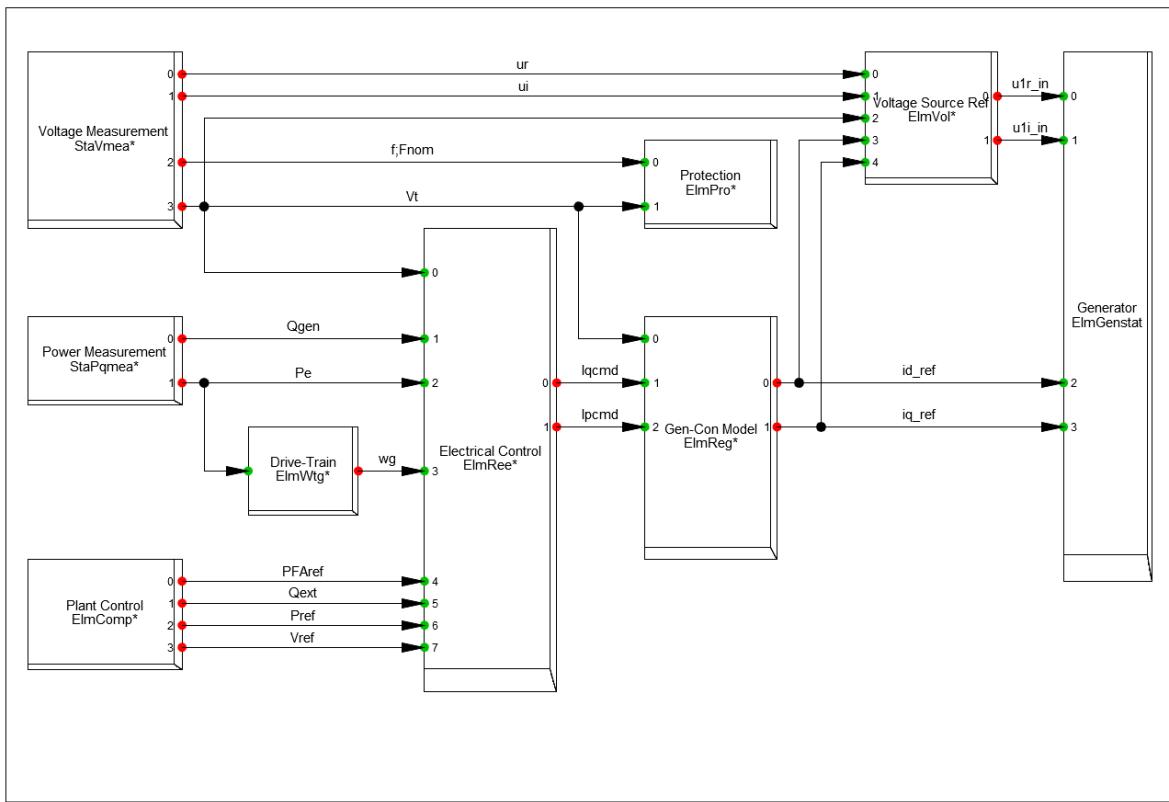
The implementation details of individual slots are not defined by the *Composite Model Frame* or the *Composite Model*. In fact, any component that complies with the given slot information is technically allowed to be assigned to a slot (via the *Composite Model*). This allows flexibility in the selection of the low-level control system components. For example, a large number of Power System Stabiliser types are available. By de-coupling the high-level diagram from the low-level implementation, the slot “Power System Stabiliser (PSS)” may accept any PSS type as long as the input/output signal names (i.e. the interface signals) are compliant. A large number of DlgSILENT Library models take advantage of this functionality; refer, for example, to the IEEE standard frames available at DlgSILENT Library → Dynamic Models→ IEEE→ Frames, where only two main *Composite Model Frames* are used for the entire IEEE conventional generation library.

One further example of a *Composite Model Frame* of a standardised WECC Wind Turbine Type 4 is shown in Figure 30.2.4. As opposed to the previous example, this implementation is a bit more complex. For many control systems, it is typical that the following functional blocks would be present:

- Measurement components
- Measurement pre-processing
- Control system functions
- Controlled/Actuated components

By visual inspection, it can be observed that measurement slots are placed on the left side of the diagram, providing various measurement quantities to a number of control slots. Furthermore, a slot corresponding to an actuated/controlled network element (e.g. a static generator) is placed on the right side of the diagram. The diagram of a *Composite Model Frame* does not state which network element it is to be assigned into any one slot, at most, it can add a filter for the class name of the targeted element. The actual assignment of a target element to a specific slot is done by the *Composite Model*.

Frame WECC WT Type 4A: Frame for WECC Wind Turbine Model Type 4A

Figure 30.2.4: Example of a *Composite Model Frame* for a standard based (WECC) type 4A wind turbine

### 30.2.3 Creating a new Composite Model Frame

A Composite Model Frame (*BlkDef* ) is a block diagram which defines two or more slots, their input and output signals, and the connections between them. A Composite Model Frame is always defined graphically.

To create a new *Composite Model Frame*:

- from the main menu:
  - Click *Insert* → *Dynamic Model* → *Composite Model Frame...*
  - In the newly shown dialog, give a name to the new object (of type *BlkDef*) and then click **OK**.
  - The diagram of the *Composite Model Frame* is now shown and it is ready to be customised
- from the Data Manager using the **New Object** button:
  - Focus the Data Manager on the library subfolder *Library* → *Dynamic Models*
  - Click on the *New Object* icon (
  - Select *Composite Model Frame*
  - In the newly shown dialog, give a name to the new object (of type *BlkDef*) and then click **OK**.
  - The diagram of the *Composite Model Frame* is now shown and it is ready to be customised
- from the *Dynamic Models* folder using the context menu (right-click):
  - Right-click on the *Dynamic Models* folder on the left side of the Data Manager and select *New* → *Dynamic Model* → *Composite Model Frame*
  - In the newly shown dialog, give a name to the new object (of type *BlkDef*) and then click **OK**.

- The diagram of the *Composite Model Frame* is now shown and it is ready to be customised

The newly created diagram will show a single rectangular block, which depicts the frame boundaries. Within the frame boundary, the interconnected control system can be drawn. The diagram name appears on the top left side.

---

**Note:** The boundary of a *Composite Model Frame* is not typically used to connect signals. There exist special cases in which such a scenario is plausible. In these cases, the boundary's left and right edges (of the outer rectangle) can be used to route input or output signal lines that connect outside the frame boundary. In such a situation, it is expected that this high-level control system will be connected to other high-level controllers in a more complex arrangement e.g. where communication between top-level controllers is required.

---

To show the Edit dialog of the *Composite Model Frame* do the following:

- Option 1: double click on any empty diagram area.
- Option 2: right-click on any empty diagram area and from the context-menu select *Edit*

### 30.2.4 Drawing a high-level control system in a *Composite Model Frame*

A new *Composite Model Frame* contains an empty diagram. The high-level controller is defined within this diagram by adding *Slots* and interconnecting them using *routed signal lines* and (optionally) *signal labels*.

Inside the diagram of a *Composite Model Frame* the following elements are allowed:

- **Slots** used to define placeholders of model components. Slots are customisable rectangular blocks containing one or more input/output nodes. An **Input/Output node** is used to define an input (green color) or an output (red color) of the *Slot*. The node types are color coded (refer to Figure 30.2.5):
  - **Red**: Output node (shown on the right side of a *Slot*)
  - **Green**: Input node (shown on the left side of a *Slot*)
  - **Magenta**: Input node, maximum limiting signal (shown on the top side of a *Slot*)
  - **Blue**: Input node, minimum limiting signal (shown on the bottom side of a *Slot*)
- **Routed signal lines** used to link an output signal node with an input signal node
- **Signal labels** used to link signals without creating a routed signal

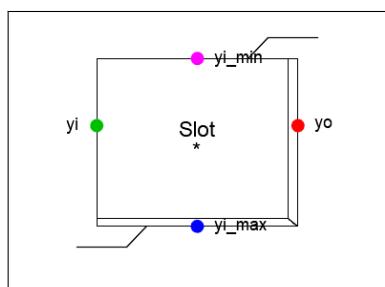
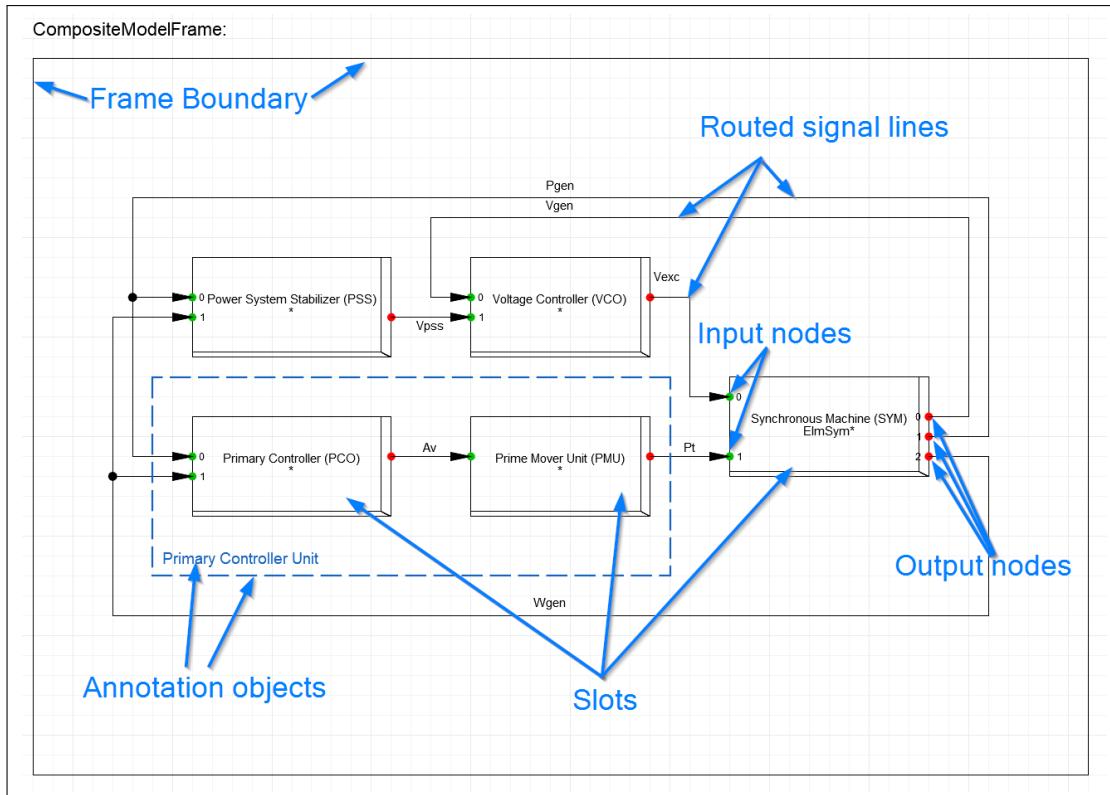


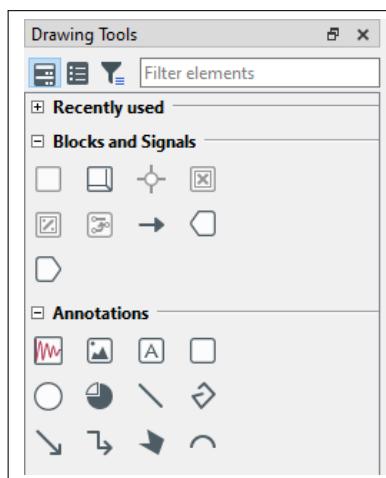
Figure 30.2.5: Example of various input/output node types of a *Slot* in a *Composite Model Frame*

With respect to the simple example shown in Figure 30.2.3, the basic components of a *Composite Model Frame* are highlighted in Figure 30.2.6.

Figure 30.2.6: Components of a *Composite Model Frame*

**Drawing these objects** can be enabled from the *Drawing Tools*. Additionally, the *Drawing Tools* contains graphical annotations as well (e.g. line, polygon, rectangle and text components) as shown in Figure 30.2.7.

**Note:** The diagram itself must be taken out of *freeze mode* ( ) in order to enable the use of *Drawing Tools*.

Figure 30.2.7: *Drawing Tools for Composite Model Frames*

To add a **Slot** to a diagram, do the following:

- Activate the *Slot* drawing mode by clicking the *Slot* ( ) icon in the *Drawing Tools*

- Position the mouse in a relevant empty area of the diagram where a *Slot* is to be added and then click to add it.
- Continue adding slots in the diagram or right-click an empty diagram area to deactivate the *Slot* drawing mode

A *Slot* can be customised as follows:

- Double click the *Slot* in order to open its Edit dialog
- Apply changes to the slot configuration, based on the information provided in Section 30.2.5. As a minimum, give it a relevant name.
- Enter the relevant input and output variables of the slot in the corresponding text fields. Each text field allows one long text string that completely defines the input/output nodes of the slot. The string can be customised as follows:
  - A new node is added to the slot for each comma separated text segment. For example:
    - \* typing in the *Input Signals* text field “ve,pt” will generate two input nodes, one node corresponding to *Input Signal* “ve”, another node corresponding to *Input Signal* “pt”.
    - \* typing in the *Output Signals* text field “i\_a,i\_b,i\_c,u\_a,u\_b,u\_c,” will generate six output nodes, one node corresponding to *Output Signal* “i\_a”, another node corresponding to *Output Signal* “i\_b”, and so on.
  - A node can aggregate two or more variables by separating these variables with a semicolon. For example:
    - \* typing in the *Input Signals* text field “ve;pt” will generate one input node, corresponding to *Input Signals* “ve” and “pt”.
    - \* typing in the *Output Signals* text field “i\_a;i\_b;i\_c,u\_a;u\_b;u\_c,” will generate two output nodes, one node corresponding to *Output Signals* “i\_a”, “i\_b” and “i\_c” and another node corresponding to *Output Signals* “u\_a”, “u\_b”, and “u\_c”.

Once a number of slots have been deployed, including their corresponding input/output signals, the slots can be interconnected using routed signal lines.

**To add a *Routed signal line*** (i.e. to interconnect *Slot* nodes of a diagram), do the following:

- Activate the *Signal* drawing mode by clicking the *Signal* (→) icon in the *Drawing Tools*
- Start by positioning the mouse on top of an *Output node* of a slot and then left-click it. Alternatively, a new branch-off signal can be created by clicking anywhere on an already existing *Routed signal line*.
- Move the mouse along a required signal route and make intermediate clicks to confine the *Routed signal line* within the desired path.
- Move the line path towards an *Input node* target destination and when on top of it, click once. The signal connection between the source *Output node* and the destination *Input node* is created.
- Continue adding signal connections to the diagram or right-click an empty diagram area to deactivate the *Signal* drawing mode

A *Routed signal line* can be customised as follows:

- Double click the *Signal* in order to open its Edit dialog
- Give a relevant *Name* to the *Signal* by editing the corresponding field. Note, the signal *Name* is independent of the names of the *Input* and *Output nodes*, as defined in the source and destination *Slots*.
- To learn more about *Routed signal lines* and better customise them, refer to the information provided in Section 30.2.5.

### Using *Signal labels*

*Slot* nodes can be interconnected using a combination of *Signal labels* and *Routed signal lines*. *Routed lines* can be “split” by *Signal labels*, thus allowing inter-connection of nodes without drawing a continuous line between them. *Signal labels* can be regarded as “shortcuts” and come in handy especially in cases of highly cluttered *Composite Model Frames*.

To add *Signal labels* to a *Composite Model Frame*, do the following:

- Activate the *Goto* drawing mode by clicking the *Goto* (□) icon in the *Drawing Tools*. *Goto* labels are used to be linked with an *Output node* of a slot (signal source). The signal of the *Goto* label will be later on linked with the *From* label and finally conveyed to the *Input node* of a slot (signal destination).
- Deploy the *Goto* label in an empty area next to the slot’s *Output node* which is of interest.
- Activate the *Signal* drawing mode by clicking the *Signal* (→) icon in the *Drawing Tools*. Connect the *Output node* of the slot with the *Input node* of the *Goto* label (□). Exit the *Signal* drawing mode by right-clicking in an empty area.
- Give a relevant name to the recently created *Routed signal line* (the signal name is further referred to “*SignalName*”).
- Activate the *From* drawing mode by clicking the *From* (□) icon in the *Drawing Tools*. *From* labels are used to link an externally named signal (e.g. of a *Goto* label for example) with an *Input node* of a slot (signal destination).
- Deploy the *From* label in an empty area next to the slot’s *Input node* (destination signal) of interest.
- Activate the *Signal* drawing mode by clicking the *Signal* (→) icon in the *Drawing Tools*. Connect the *Output node* of the *From* label (□) with the *Input node* of the destination *Slot*. Exit the *Signal* drawing mode by right-clicking in an empty area.
- Double click the newly created *From* label (□). In its Edit dialog, set the text field *Signal from* to be “*SignalName*”.
- A link is created between the *Output node* of signal source *Slot* and the *Input node* of a signal destination *Slot* (refer to Figure 30.2.8 for a simple example).

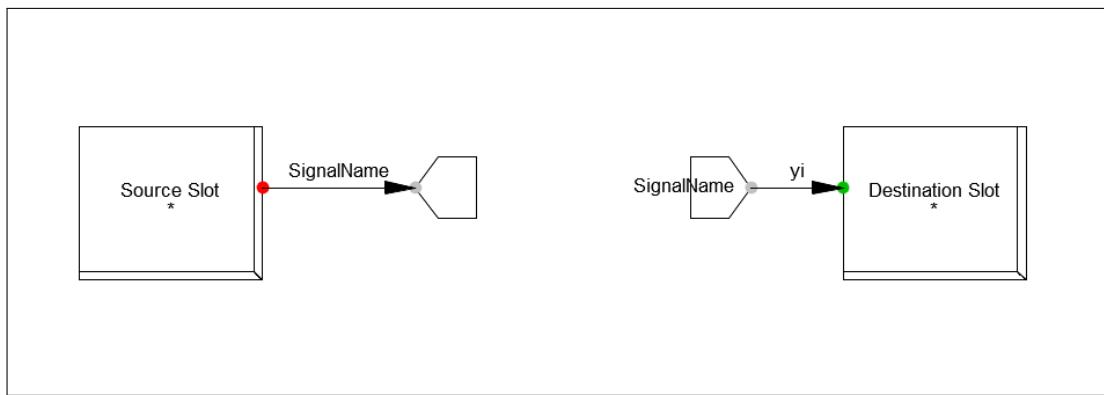


Figure 30.2.8: Using *Signal labels* to inter-connect *Output* and *Input* nodes

### 30.2.5 Configuration of *Composite Model Frame* components

#### 30.2.5.1 Customisation options for Slots

The **Slot options** are detailed as follows:

- **Name** - assign a relevant name to this slot. The *Slot* name is shown in the diagram and is used to identify the slot further when using a *Composite Model*.
- **Sequence** - This parameter accepts any integer number. It defines the order in which slots appear in the *Composite Model* dialog. The *Composite Model* dialog displays a listing of all slots within the referenced *Composite Model Frame* in a tabular format. The order in this tabular format, from top to down is decided by the *Sequence* parameter: the slot having the smallest *Sequence* value appears on top of the list (first row); the slot having the second smallest *Sequence* value appears on row 2 of the list, and so on. Note, inserting new *Slots* to the diagram will index upwards the *Sequence* value of these *Slots* (the highest value of any existing slot plus 1 is assigned to the last inserted *Slot*)
- **Type** - [optional]; typically, no type is assigned to a *Slot*. It can be used to assign a specific *DSL Model Type* (i.e. a DSL based controller). If assigned, the *Slot* automatically reads the inputs and outputs of the *DSL Model Type* and applies them graphically by adding input and output nodes to the *Slot*.
- *Filter for pane:*
  - **Class Name** - [optional], the class name(s) of the components which may be assigned to this slot can be stated here. During the *Component assignment* process of the *Composite Model*, *PowerFactory* will only allow components of the specified class(classes) to be assigned to this *Slot*. For example, if the *Class Name* of a slot is set to “ElmSym”, then only synchronous machine elements will be eligible for component assignment in the *Slot*.
  - **Model Name** - [optional], the model name(s) of the component which may be assigned to this slot can be stated here. During the *Component assignment* process of the *Composite Model*, *PowerFactory* will only allow components with the specified name(s) to be assigned to this *Slot*. For example, if the *Model Name* of a slot is set to “Generator 1”, then only an element named “Generator 1” will be eligible for component assignment in the *Slot*.

**Note:** The fields *Filter for Class Name* and *Model Name* accept standard *PowerFactory* filter text formats. The symbol “\*” is commonly used in both the fields *Class Name* and *Model Name* in order to denote “any”. For example, when a slot is intended to contain objects of any class starting with “Sta”, then the *Class Name* field shall contain “Sta\*”.

---

- *Classification pane:*
  - **Linear** - set this checkbox if the model to be contained in this slot is intended to be a linear-time invariant model.
  - **Automatic, model will be created** - rarely used; when this option is activated, the function *Slot Update* (refer to Section 30.2.6.1) automatically creates a DSL model and asks for a *DSL Model Type* from the library.
  - **Local, model must be stored inside** - This option is activated by default. If active, then when a **Slot Update** is executed in the *Composite Model*, *PowerFactory* will only search for elements which are stored inside the *ElmComp* for this slot. Any already existing reference to models which are stored outside (e.g. the synchronous generator in a plant model) will be removed from the *Component assignment* list.
  - **Main Slot** - this flag identifies this *Slot* as being critical to the correct operation of the entire high-level controller defined by this *Composite Model Frame*. Multiple slots of a *Composite Model Frame* can be set to be *Main Slot*. The operation of a corresponding *Composite Model* is affected by this flag as follows:
    - \* if the component assigned to a *Main Slot* is *out of service* then the entire *Composite Model* and its contents will not be considered during the current simulation.
    - \* if a *Slot* marked as being a *Main Slot* does not have any component assigned to it then the entire *Composite Model* and its contents will not be considered during the current simulation.
- *Upper Limitation pane:*
  - **Limiting Input Signals** - type in this text field the upper limitation input signals (if any), delimited by commas. The input signals will be shown on the top side of the *Slot* (using corresponding input nodes);

- *Lower Limitation* pane:
  - **Limiting Input Signals** - type in this text field the lower limitation input signals (if any), delimited by commas. The input signals will be shown on the bottom side of the *Slot* (using corresponding input nodes);
- *Variables* pane:
  - **Output Signals** - type in this text field the model's output signals (if any), delimited by commas. The input signals will be shown on the left side of the *Slot* (using corresponding input nodes). The numbers of output signals and shown output nodes need not be identical: signals can be aggregated into a single output node by separating them using semicolons (instead of commas).
  - **Input Signals** - type in this text field the model's input signals (if any), delimited by commas. The input signals will be shown on the left side of the *Slot* (using corresponding input nodes). The numbers of input signals and shown input nodes need not be identical: signals can be aggregated into a single input node by separating them using semicolons (instead of commas).

---

**Note: How to identify signals (and their names) of a model in order to set the inputs/outputs of a corresponding slot?**

- If the slot is meant for a built-in model:
  - \* Check the Technical Reference documentation of the specific built-in model (e.g. generator, measurement device)
  - \* Alternatively, using the Data Manager or the Network Model Manager, navigate to an element of the same class as the targeted built-in model of the slot.
  - \* Use the *Flexible Data* tab and open the *Variable Selection* window (e.g. right-click on the *Flexible Data* tab and choose *Variable selection*).
  - \* From the *Variable Selection* window, use the filter *Group* to get all the *Signals and States*.
  - \* A listing of all available signals is provided. Available signals may be inputs (type *IN*), outputs (type *OUT*), model states (type *STATE*) or model state derivative (type *d/dt*).
  - \* From the provided list, outputs (type *OUT*) can be set in the *Output Signals* field of a *Slot*. These signals can therefore be taken from the built-in model and further provided to other components placed in *Slots*. This is a typical use case in which a network element (e.g. a measurement device) provides signals to a controller slot via a specific signal.
  - \* From the provided list, inputs (type *IN*) can be set in the *Input Signals* field of a *Slot*. These signals can therefore be linked with other signals sourced from other components placed in other *Slots*. This is a typical use case in which a network element is controlled via a specific signal.
  - \* Note: all signals in *PowerFactory* (and in *DSL/Modelica*) are case sensitive.
- If the slot is meant for a User-defined model (e.g. *DSL* or *Modelica Model*):
  - \* Analyse the *User-defined* model and identify the inputs and outputs. This can be done either by visual inspection of the model's block diagram or by verifying the declared inputs and outputs in the corresponding dialog windows of the controllers.
  - \* Note: all signals in *PowerFactory* (and in *DSL/Modelica*) are case-sensitive.

---

### Assigning a *DSL Model Type* to a Slot

A *DSL Model Type* (*BlkDef*) can be assigned directly to a slot. This option will simplify the handling of the slot and prevent errors due to mismatched signal names of slot and assigned block.

To assign the external form of a *DSL Model Type* to the selected slot, edit the slot by double-clicking it and choose the *Select* button  for the “*DSL Model Type*” in the dialog. Now the *DSL Model Type* can be selected, e.g. the type of controller or built-in element, which should be assigned to this slot later.

As an example, if the newly-defined slot ought to represent a synchronous machine's automatic voltage regulator (AVR) in the frame diagram, a predefined *DSL Model Type* can be chosen to insert the AVR

model including input and output signals to this slot. A controller should only be assigned to a slot, when only this type of controller is to be inserted into this slot, and no other model can be.

When the *DSL Model Type* is selected, the input and output as well as limiting signals will disappear from the slot dialog. The filter for the class name will automatically be entered. When clicking on the **OK** button, the slot will then show the right inputs and outputs according to the *DSL Model Type*.

---

**Note:** When a *DSL Model Type* is assigned directly to a slot, only the input/output signals are set automatically. The internal equations/definitions of the *DSL Model Type* are not implemented in the slot and the slot itself remains empty. There is always the need to create a *DSL Model*, which is the model inserted into the slot of the composite model. When the slot refers to an outside *DSL Model Type*, beware that this reference is also inside your project. If the reference to the definition is invalid or changed, the slot may be changed as well. Therefore, assign a block very carefully.

---

### 30.2.5.2 Customisation options for *Routed signal lines*

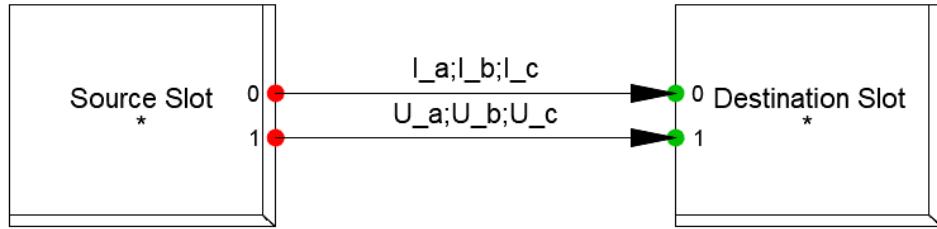
Routed signal lines always connect one *Output node* (a source) with one *Input node* (a destination). Routed signal lines always perform a one-to-one mapping of the declared variable(s) of the *Output node* to the declared variable(s) of the *Input node*. This procedure is valid for any number of variables that define an *Input* or an *Output* node.

In the trivial case in which one node represents one variable of the slot's component, then the one-to-one mapping translates to mapping one single variable to another one. For example, with reference to Figure 30.2.6, the slot *Voltage Controller (VCO)* defines one output node as "Vexc". The slot *Synchronous Machine (SYM)* defines the input nodes as "ve,pt". It logically results that the slot *Voltage Controller (VCO)* will show one output node, whereas the slot *Synchronous Machine (SYM)* will display two input nodes. The connection (manually done) between these slots has resulted in a *Routed signal line* named "Vexc" that connects the single output of the *Voltage Controller (VCO)* to the first input node of the *Synchronous Machine (SYM)* i.e. variable "ve". A one-to-one mapping is obtained via the *Routed signal line* named "Vexc" between the *Voltage Controller (VCO)* variable "Vexc" (which is an output signal of this controller) and the *Synchronous Machine (SYM)* variable "ve" (which is an input signal of this machine element).

A multi-variable *Routed signal line* is defined by connecting a multi-variable *Output node* with a multi-variable *Input node*. Such a choice is useful in highly cluttered *Composite Model Frames*, where the variables exchanged between slots need not be explicitly drawn using individual *Routed signal lines*. For example two slots have six signals to inter-connect:

- Signal source *Slot* has output signals: i\_a, i\_b, i\_c, u\_a, u\_b and u\_c
- Signal destination *Slot* has input signals: I\_a, I\_b, I\_c, U\_a, U\_b and U\_c

The multi-variable *Routed signal line* functionality can be used to link the two slots with only two *Routed signal lines* (instead of six), by grouping the signals based on their functionality (phase currents and phase voltages). Consequently, the signal source *Slot* may state two output nodes by appropriately adjusting the text field *Output Signals* to contain: "i\_a;i\_b;i\_c,u\_a;u\_b;u\_c". The signal destination *Slot* may state two input nodes by appropriately adjusting the text field *Input Signals* to contain: "I\_a;I\_b;I\_c,U\_a;U\_b;U\_c". It is now possible to use two *Routed signal lines* and link the two slots together, as shown in Figure 30.2.9.

Figure 30.2.9: Example of a multi-variable *Routed signal line*

Important notes:

- The order of writing variables of the *Output* and *Input nodes* is important. The mapping between source and destination variables is done by mapping the first variable in the multi-variable string (e.g. “`out_variable_1;out_variable_2;..`”) of the *Output Node* to the first variable found in the multi-variable string (e.g. “`in_variable_1;in_variable_2;..`”) of the *Input node*, the second variable of the *Output Node* is mapped to the second of the *Input Node* and so on.
- The number of variables of the multi-variable string of the *Output Node* must be equal to the number of variables of the multi-variable string of the *Input Node*

### 30.2.6 Creating and configuring a *Composite Model*

The ***Composite Model*** (*ElmComp* ) represents an “instance” of a *Composite Model Frame* in the power system. The high-level control structure is conceptually defined within the *Composite Model Frame*, but without assigning specific controlled equipment and controllers to it. The “Component assignment” process is specific to a *Composite Model*. When a new *Composite Model* is created, a reference to *Composite Model Frame* must be defined in order to obtain a functional system. The *Composite Model* parses through the graphical definition of the *Composite Model Frame* and extracts a list of all defined *Slots*.

To create a ***Composite Model*** (*ElmComp* ), do the following:

- using the *Data Manager* (option 1):
  - Navigate inside a *Network Data* folder
  - Click the *New Object* ( ) icon
  - Select *Composite Model* and click **OK**
  - Assign a *Composite Model Frame* to the *Frame* field in the Edit dialog of the *Composite Model*. Further configure the model.
- using the *Data Manager* (option 2):
  - Navigate inside a *Network Data* folder
  - Right-click in the right side window of the *Data Manager* and from the context menu select *New → Others...*
  - From the dialog select *Composite Model* and click **OK**
  - Assign a *Composite Model Frame* to the *Frame* field in the Edit dialog of the *Composite Model*. Further configure the model.

### 30.2.6.1 Component assignment in a Composite Model

When opening the Edit dialog of a *Composite Model*, users are presented with a tabulated assignment list with two columns, as shown for example in Figure 30.2.10. The first column (entitled “Slots”) is an ordered list of all contained *Slots* within the *Composite Model Frame* (based on the *Sequence* field of each *Slot*). The second column (entitled “Net Elements”) is empty by default i.e. when newly creating a *Composite Model*. To each field within this second column a reference to one object within the power system model can be assigned. The process of referencing objects within the second column of a *Composite Model*, thereby creating mappings between a specific slot and a specific power system component is called the “**Component assignment**” of a *Composite Model*.

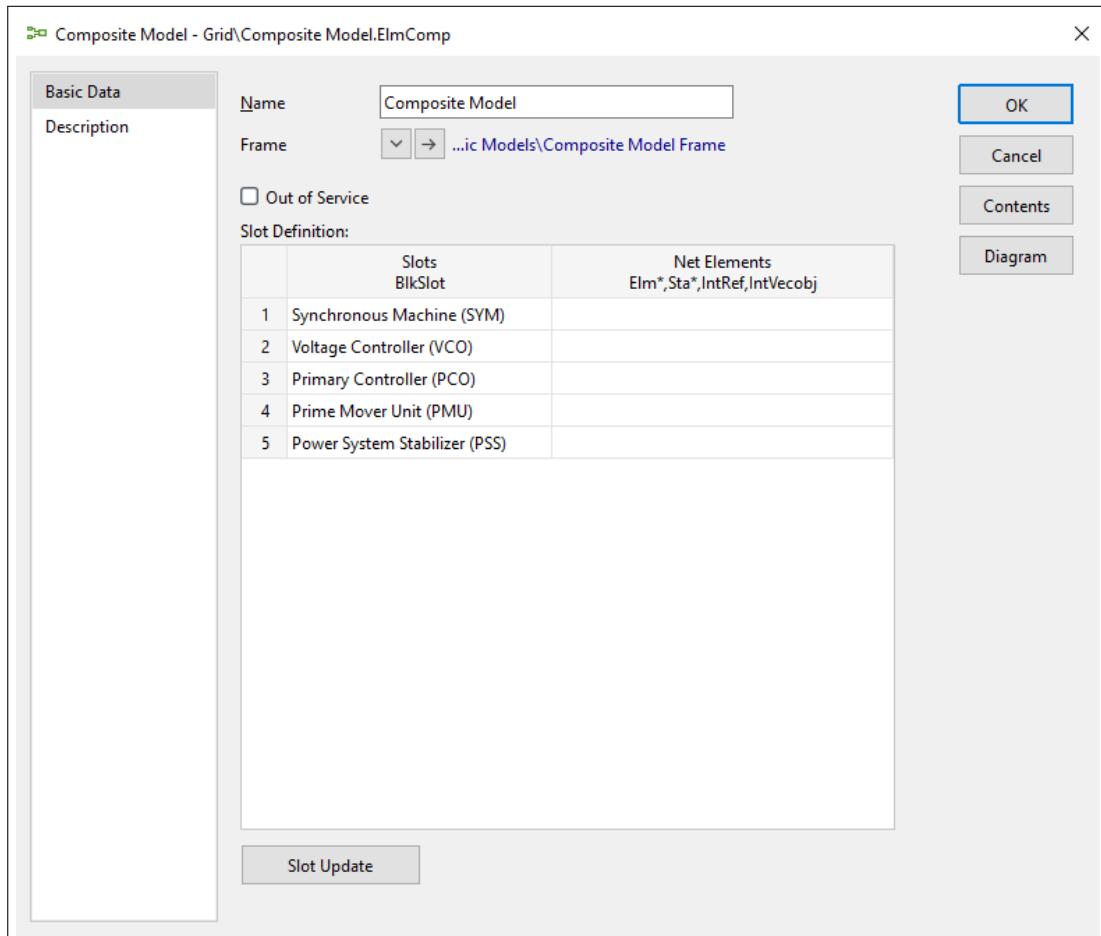


Figure 30.2.10: Example of a Composite Model for Conventional Generating Units

To assign a component to a slot, do the following:

- right-click the *Net Elements* field corresponding to the specific *Slot*
- Select the option *Select Element/Type*. A selection dialog is shown.
- Browse and identify the required component to be assigned and select it (by clicking on it in the right side of the selection dialog)
- Click **OK**. The selected component is assigned to the slot.

#### Important notes:

- Any number of *Composite Models* can be created, each referencing to the same *Composite Model Frame*.

- It is not possible for a *Composite Model* to reference to multiple *Composite Model Frames*.
- Not all slots of a *Composite Model* must necessarily be assigned. There can be empty slots assignments i.e. the *Net Elements* field corresponding to a specific slot can be left empty. In such cases, any inputs/outputs connecting to/from this slot will not be connected by this *Composite Model*.
- *Slots* in the *Composite Model Frame* may be pre-configured to accept (i.e. filter for) only specific component classes or names.
- With reference to Figure 30.2.10, the *Synchronous Machine (SYM)* slot is intended to be assigned a *Synchronous machine* element (built-in element). The slots *Voltage Controller (VCO)*, *Primary Controller (PCO)*, *Prime Mover Unit (PMU)* and the *Power System Stabiliser (PSS)* are, however, user-defined dynamic models. They may be assigned any of the supported *User-defined* model types (refer to Figure 30.1.1 for an overview of allowed model types). For example, it is possible to define corresponding controllers using *DSL* language by creating corresponding *DSL Models* (using *ElmDsl* and *BlkDef* objects) for each functional component. Alternatively, *Modelica Models* (using *ElmMdl* and *TypMdl* objects) can be used, leading to similar results.
- When inserting controller models into a slot, it is often the case that the controller element has not yet been created. To create a new controller element during the component assignment process, select *New Element/Type* from the slot's context menu. *PowerFactory* will automatically jump to the project Library and show a list of available *DSL Model Types* (*BlkDef*). Selecting a *DSL Model Type* from the project library or the global library will open the dialog of the newly-created *DSL Model*, so that its parameters can be defined.
- If an element is already assigned to a slot, it is possible to edit the assigned element by simply right-clicking and selecting *Edit Element/Type*.
- The right-mouse button menu entry *Reset Element/Type* will reset the slot, so that it is empty again.

---

**Note:** Depending on the settings of the individual slot, the menu entry *Reset Element/Type* will not only clear the marked slot but also delete the built-in or *DSL Model*, if it is stored inside the composite model in the Data Manager.

---

### Slot Update

The **Slot Update** button (found in the *Basic Data* page of the Edit Dialog of the *Composite Model*) re-reads the slot information from the *Composite Model Frame*, checks for validity all existing *Component assignments* and tries to re-assign components found inside the *Composite Model*.

If a specific assignment is found to be *invalid* then the assignment is reset. A *Component assignment* is *invalid* when a model has been assigned to a slot which is not suited to receive the already assigned type of model, i.e. a voltage controller cannot be assigned to a slot defined for a primary controller model, or when the name of the model does not fit the *Slot* "Filter for" information.

All built-in models and *DSL Models* which have been created for a specific *Composite Model* are stored in that *Composite Model* itself. The contents of a *Composite Model* is shown in the Data Manager where the *Composite Model* is treated as a normal database folder. Basic power system equipment, such as synchronous machines or static VAr compensators, are normally not stored inside the *Composite Model*, but in the grid itself.

Furthermore, the *Slot update* will try to re-assign each model found in its contents to the corresponding Slot. The options defined for each *Slot* are important, and are described in Section 30.2.5.

### Show Graphic

To show the diagram of the *Composite Model* (bound to it), click the button **Show Graphic** in the *Basic Data* page of the Edit Dialog of the *Composite Model*.

### 30.2.6.2 Simplified creation of pre-defined *Composite Models* for standard elements

A simple method of defining *Composite Models* directly from the single line diagram of a grid is available for a predefined set of *Composite Model Frames*. The method can be applied to the following network elements:

- The synchronous motor and generator;
- The asynchronous motor and generator;
- The static VAr system.

For these elements, if a *Composite Model* is not already defined, then do the following:

- Right-click the element in the single line diagram
- Select *Network Models* from the context menu of the element.
- For a Synchronous machine choose one of the following options:
  - *Excitation System (exc)* → *New...*
  - *Governor and Turbine (gov)* → *New...*
  - *Power System Stabiliser (pss)* → *New...*
- For a Asynchronous machine choose one of the following options:
  - *New Motor Driven Machine (mdm)* → *New...*
  - *Motor Driven Machine (mdm)* → *from Library...*
- For a static VAr system:
  - *Controller* → *New...*

When a standard composite model is available for the selected object, a list of the available controllers is shown. Selecting a controller will add it to the composite model, which is automatically created when no composite model exists for the selected object.

### 30.2.7 Array signal distribution/aggregation in high-level control systems

Within many applications, it is common to use vector-based signals (one-dimensional signal arrays) that are distributed to a high number of network elements having a similar structure. Two examples can be mentioned:

- Individual dispatch of Wind Turbine Generators by a Power Plant Controller - for RMS-/EMT-domain simulation (used as use case example in this Section)
- Dispatch of gate signals to individual MMC modules in a detailed HVDC-MMC based system - for EMT-domain simulation (refer to Section 30.15.2 for an in-depth description)

The *Vector of Objects* element (*IntVecobj*) enables *PowerFactory* users to integrate the following high-level control systems functions:

- Distribution of one or multiple array signals from one source component to several destination components;
- Aggregation of one or multiple array signals from several source components to one destination component.

A summary of the possible configurations for routing array signals (or sets of scalar signals) in a high-level control system is given in Table 30.2.1.

	Output Slot	Output Node Type	Input Slot	Input Node Type
Configuration 1	Model	array(1 : n)	Model	array (1 : n)
Configuration 2	Model	array(1 : n)	$n$ Models (IntVecobj)	scalar
Configuration 3	Model	array(1 : $n \cdot m$ )	$n$ Models (IntVecobj)	array (1 : $m$ )
Configuration 4	Model	array(1 : $n \cdot m$ )	$n$ Models (IntVecobj)	$m$ scalar
Configuration 5	Model	array(1 : n)	Model	$n$ scalar
Configuration 6	$n$ Models (IntVecobj)	scalar	Model	array (1 : n)
Configuration 7	$n$ Models (IntVecobj)	array(1 : $m$ )	Model	array(1 : $n \cdot m$ )
Configuration 8	$n$ Models (IntVecobj)	$m$ scalar	Model	array(1 : $n \cdot m$ )
Configuration 9	Model	$n$ scalar	Model	array (1 : n)
Configuration 10	$n$ Models (IntVecobj)	scalar	$n$ Models (IntVecobj)	scalar
Configuration 11	Model	scalar/array[index]	$n$ Models (IntVecobj)	scalar
Configuration 12	Model	array[index]	Model	scalar

Table 30.2.1: Possible configurations for routing array signals in a high-level control system

**Note:**

- By “scalar” it is meant that a scalar variable is defined at the *Input* or *Output node* e.g. “ua” or “y\_out”;
- By “ $n$  scalar” it is meant that a multi-variable (a set of scalar variables) is defined in the *Input* or *Output node*, as per the details in Section 30.2.5.2 e.g. “ua;ub;uc”;
- By “array(1 : n)” or “array(1 : m)” “array(1 :  $n \cdot m$ )” it is meant that a one-dimensional array variable can be defined in the designated *Input* or *Output node* e.g. “my\_array\_signal”. The array size is declared here for clarity e.g.  $n$ ,  $m$  or  $n \cdot m$ , but it is not intended to be written in the *Input* or *Output Node* entry fields;
- By “array[index]” it is meant that one index (0-based) of a one-dimensional array variable can be defined in the designated *Input* or *Output node* e.g. “my\_array\_signal[2]” extracts the third value of array “my\_array\_signal”.
- The *index* in *array[index]* is an integer number corresponding to the array index (0-based) being extracted from the whole *array* variable; Pay special attention to interconnecting *Modelica Models* in a high-level control system, as array variables (and therefore input and output signals as well) are 1-based in Modelica. For example, in the code of a *Modelica Model*, the second index of an array variable is referred to by “my\_array\_variable[2]” (i.e. 1-based). In a *Slot*, in order to extract the second index of the *Modelica Model* array variable “my\_array\_variable” the *Input* or *Output Node* must contain “my\_array\_variable[1]” (i.e. 0-based).
- By “Model” it is meant that an individual element is placed in the designated *Slot* e.g. a synchronous machine element (*ElmSym*), a voltage measurement (*StaVmea*);
- By “ $n$  Models (IntVecobj)” it is meant that a *Vector of Objects* element is assigned to the designated *Slot*. The *Vector of Objects* element defines (via its Edit dialog) a list of  $n$  components (models) which are to be used in the array signal routing process.

### 30.2.7.1 Configuration 1: From an array signal to an array signal

The connection of two array signals in a high-level control system is supported as shown in Figure 30.2.11.

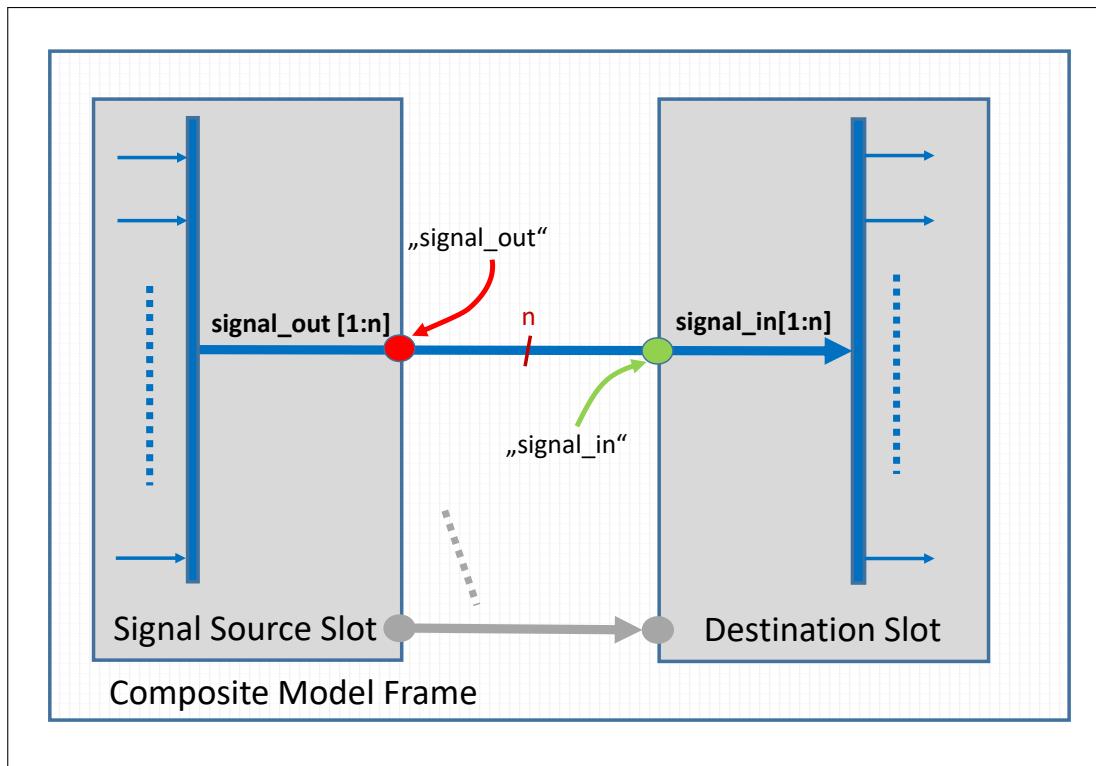


Figure 30.2.11: Configuration 1: From an array signal to an array signal

**Note:**

- The *Output Signals* text field of the *Signal source Slot* should contain the name of the source array signal (e.g. “signal\_out”);
- The *Input Signals* text field of the *Destination Slot* should contain the name of the destination array signal (e.g. “signal\_in”);
- The *Signal source Slot* and *Destination Slot* can be linked with multiple such connections by adding similarly defined *Output / Input nodes* separated by commas. This is graphically suggested by the gray coloured *Output / Input nodes* and their corresponding *Routed signal line*.
- If the size of the input and output arrays does not match, an error message is issued and the simulation is stopped.

**Example 1: Linking two models each having an array signal**

In this scenario, “Model 1” has an output signal *signal\_out* of size *n* and “Model 2” has an input signal *signal\_in* of size *n*.

To connect the two signals, observe the following:

- The *Output Signals* text field of the *Source Slot* should contain “signal\_out”;
- The *Input Signals* text field of the *Destination Slot* should contain “signal\_in”.

**Example 2: Linking one model containing multiple scalar outputs with one model having an array signal input**

In this scenario, “Model 1” has three scalar output signals  $ua$ ,  $ub$  and  $uc$  and “Model 2” has an input signal  $u$  of size 3.

To connect the signals, observe the following:

- The *Output Signals* text field of the *Source Slot* should contain “[ $ua;ub;uc$ ]”;
- The *Input Signals* text field of the *Destination Slot* should contain “ $u$ ”.

**Example 3: Linking one model containing one array output signal with one model having multiple scalar inputs**

In this scenario, “Model 1” has one output signal  $gate$  of size 6 and “Model 2” has six input signals  $g1$ ,  $g2$ ,  $g3$ ,  $g4$ ,  $g5$  and  $g6$ .

To connect the signals, observe the following:

- The *Output Signals* text field of the *Source Slot* should contain “ $gate$ ”;
- The *Input Signals* text field of the *Destination Slot* should contain “[ $g1;g2;g3;g4;g5;g6$ ]”.

### 30.2.7.2 Configuration 2: From an array signal to a scalar signal belonging to a set of $n$ models

The connection of an array signal with a scalar signal belonging to a set of  $n$  models is supported as shown in Figure 30.2.12.

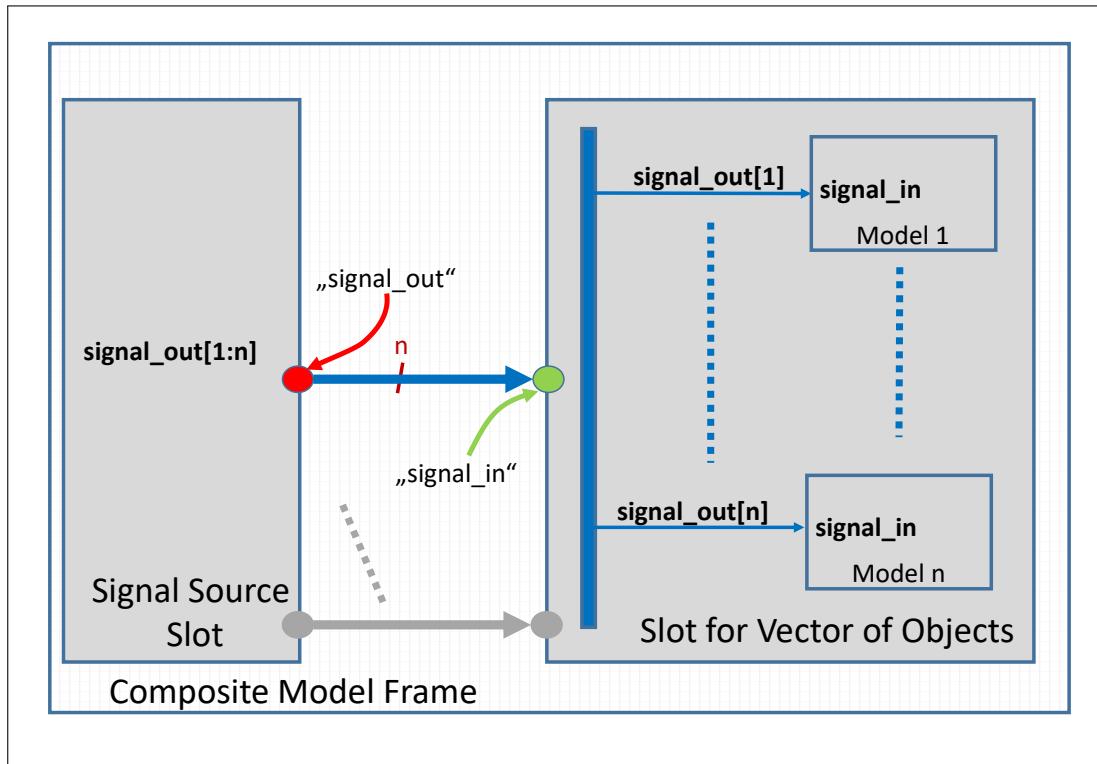


Figure 30.2.12: Configuration 2: From an array signal to a scalar signal belonging to a set of  $n$  models

**Note:**

- The *Output Signals* text field of the *Signal source Slot* should contain the name of the source array signal (e.g. “`signal_out`”);
- The *Input Signals* text field of the *Slot for Vector of Objects* should contain the name of the destination signal (e.g. “`signal_in`”). The destination signal name must be the same for all  $n$  models of the set.
- The *Signal source Slot* and the *Slot for Vector of Objects* can be linked with multiple such connections by adding similarly defined *Output / Input nodes* separated by commas. This is graphically suggested by the gray coloured *Output / Input nodes* and their corresponding *Routed signal line*.
- If the size of the output array is less than  $n$ , an error message is issued and the simulation is stopped.

### 30.2.7.3 Configuration 3: From an array signal to an array signal belonging to a set of $n$ models

The connection of an array signal with an array signal belonging to a set of  $n$  models is supported as shown in Figure 30.2.13.

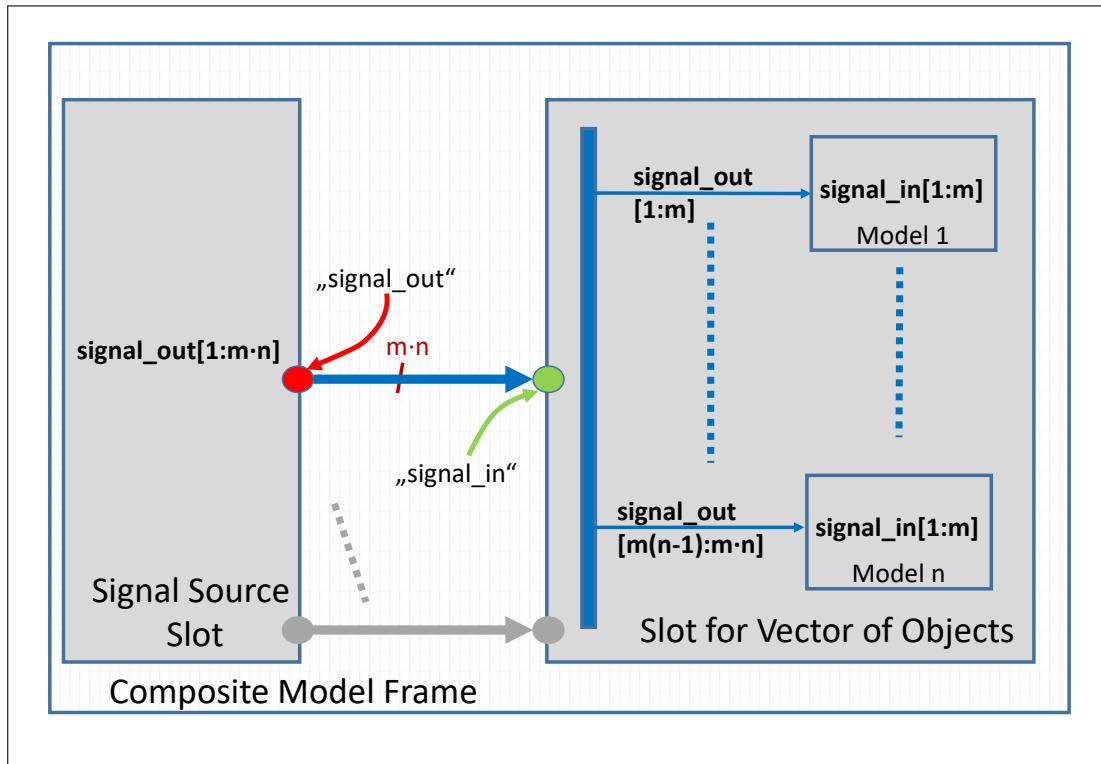


Figure 30.2.13: Configuration 3: From an array signal to an array signal belonging to a set of  $n$  models

**Note:**

- The *Output Signals* text field of the *Signal source Slot* should contain the name of the source array signal (e.g. “`signal_out`”);
- The *Input Signals* text field of the *Slot for Vector of Objects* should contain the name of the destination array signal (e.g. “`signal_in`”); The destination array signal name must be the same for all  $n$  models of the set.
- The *Signal source Slot* and the *Slot for Vector of Objects* can be linked with multiple such connections by adding similarly defined *Output / Input nodes* separated by commas. This is graphically suggested by the gray coloured *Output / Input nodes* and their corresponding *Routed signal line*.
- An error message is issued when the dimension of the output signal is smaller than the dimension of the input signal ( $k$ ) multiplied by the number of selected elements. Further, an error message is issued when the dimension of the output signal is not an integer of the input signal dimension.

### 30.2.7.4 Configuration 4: From an array signal to a set of scalar signals belonging to a set of $n$ models

The connection of an array signal with a set of scalar signals (a multi-variable *Node*) belonging to a set of  $n$  models is supported as shown in Figure 30.2.14.

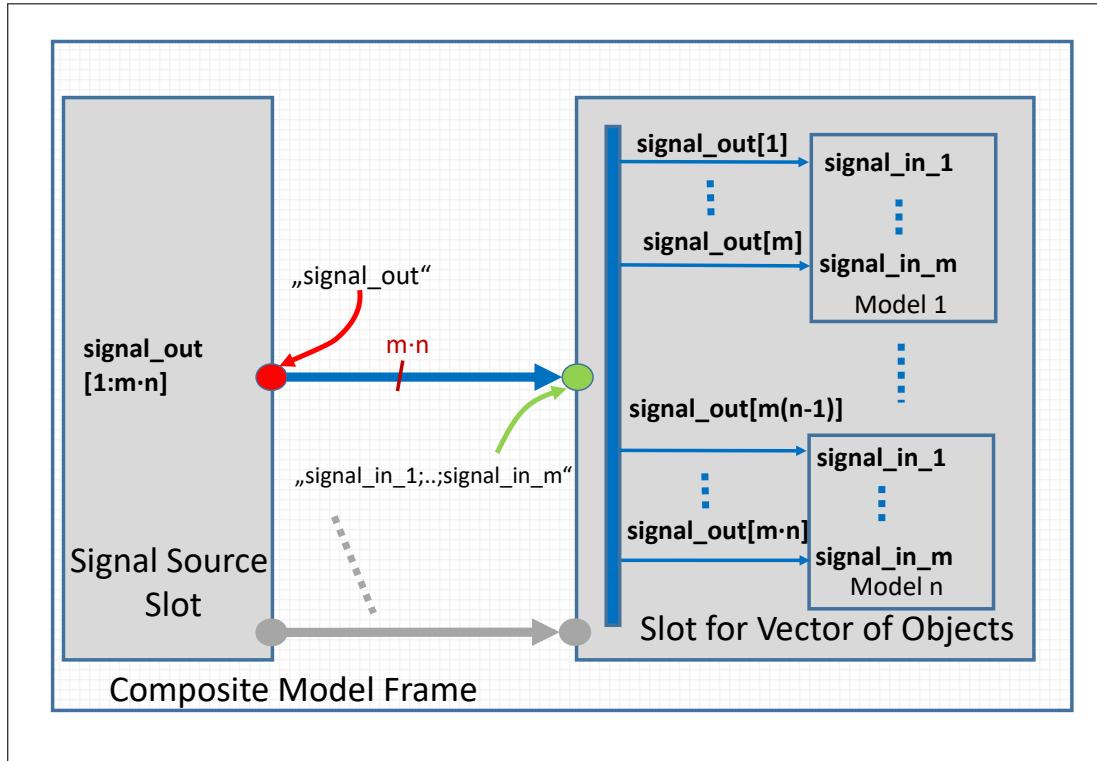


Figure 30.2.14: Configuration 4: From an array signal to a set of scalar signals belonging to a set of  $n$  models

- Note:**
- The *Output Signals* text field of the *Signal source Slot* should contain the name of the source array signal (e.g. “*signal\_out*”);
  - The *Input Signals* text field of the *Slot for Vector of Objects* should contain the name of the destination set of scalar signals (a multi-variable *Node* e.g. “*signal\_in\_1;..;signal\_in\_m*”). The names and order of the destination set of scalar signals must be the same for all  $n$  models of the set.
  - The *Signal source Slot* and the *Slot for Vector of Objects* can be linked with multiple such connections by adding similarly defined *Output / Input nodes* separated by commas. This is graphically suggested by the gray coloured *Output / Input nodes* and their corresponding *Routed signal line*.
  - An error message is issued when the dimension of the output signal is smaller than the total dimension of the input signal ( $k$ ) (for one element) multiplied by the number of selected elements. Further an error message is issued when the dimension of the output signal is not an integer of the input signal dimension.

### 30.2.7.5 Configuration 5: From an array signal to a set of scalar signals belonging to one model

The connection of an array signal with a set of scalar signals (a multi-variable *Node*) belonging to a one model is supported as shown in Figure 30.2.15.

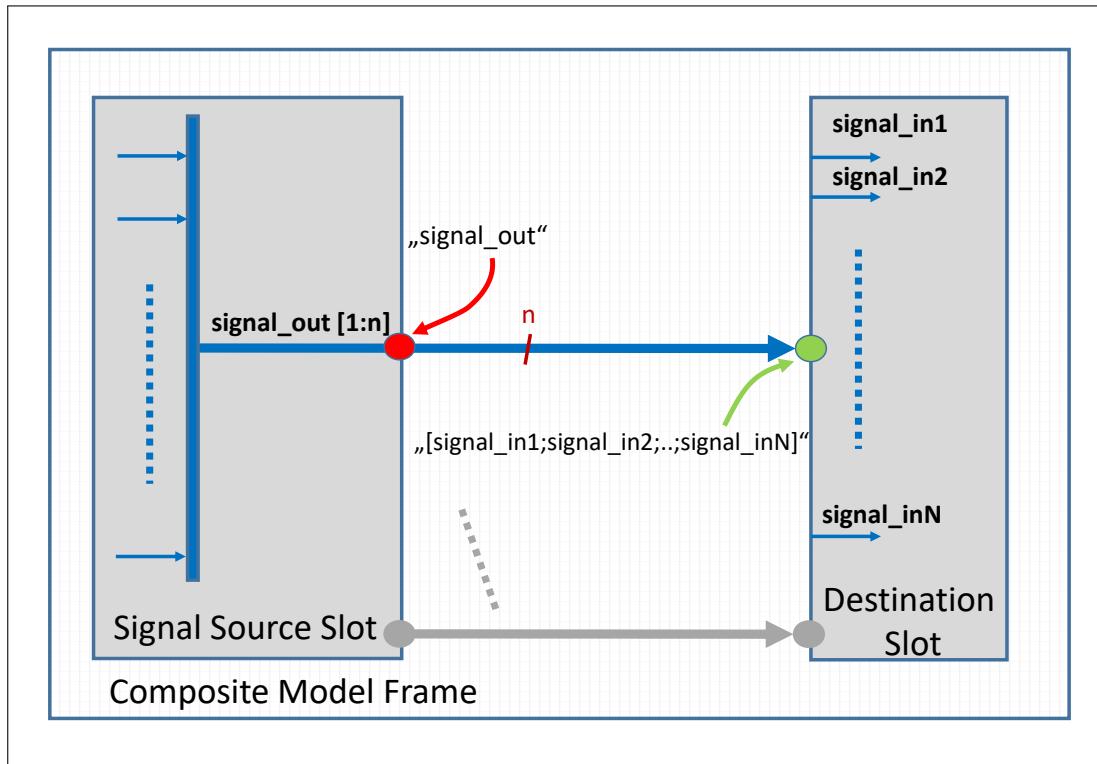


Figure 30.2.15: Configuration 5: From an array signal to a set of scalar signals belonging to one model

**Note:**

- The *Output Signals* text field of the *Signal source Slot* should contain the name of the source array signal (e.g. “signal\_out”);
- The *Input Signals* text field of the *Destination Slot* should contain the name of the destination set of scalar signals (e.g. “[signal\_in1;..;signal\_inN]”);
- The *Signal source Slot* and *Destination Slot* can be linked with multiple such connections by adding similarly defined *Output / Input nodes* separated by commas. This is graphically suggested by the gray coloured *Output / Input nodes* and their corresponding *Routed signal line*.
- An error message is issued if the dimension of the output signal is smaller than the total dimension of all defined input signals.

### 30.2.7.6 Configuration 6: From a scalar signal belonging to a set of $n$ models to an array signal of a model

The connection of a scalar signal belonging to a set of  $n$  models with an array signal of a model is supported as shown in Figure 30.2.16.

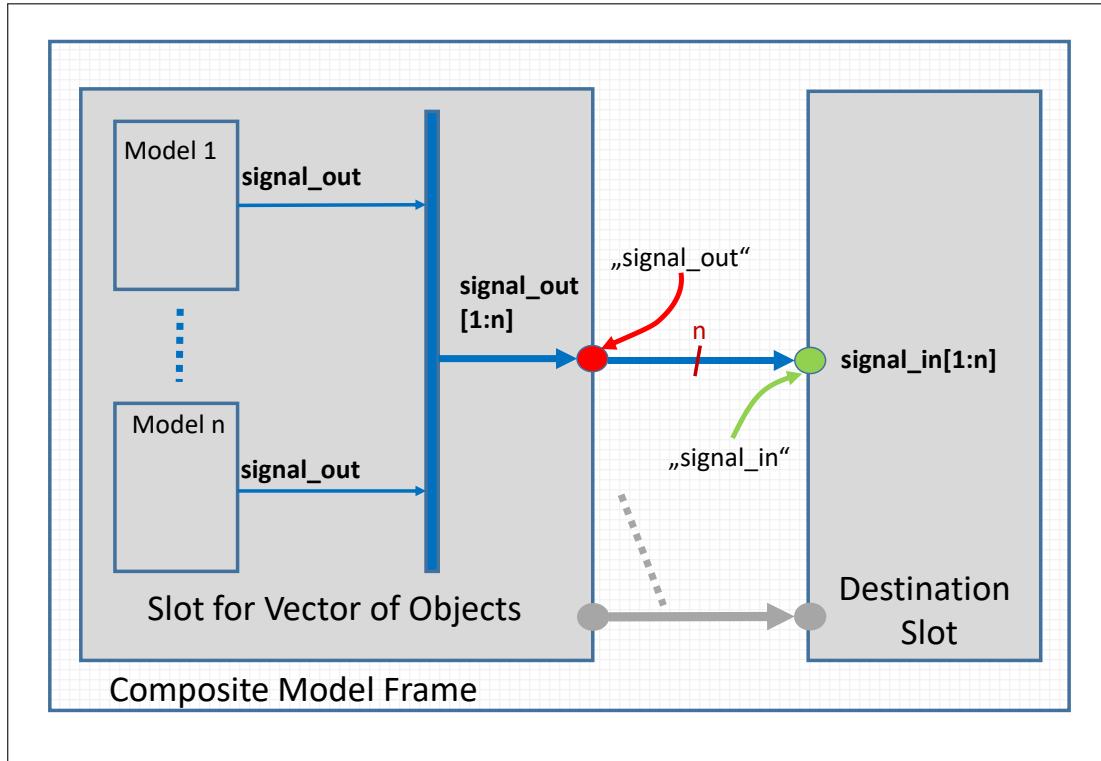


Figure 30.2.16: Configuration 6: From a scalar signal belonging to a set of  $n$  models to an array signal of a model

- Note:**
- The *Output Signals* text field of the *Slot for Vector of Objects* should contain the name of the source array signal (e.g. "signal\_out"). The name of the source signal must be the same for all  $n$  models of the set.
  - The *Input Signals* text field of the *Destination Slot* should contain the name of the destination array signal (e.g. "signal\_in");
  - The *Slot for Vector of Objects* and *Destination Slot* can be linked with multiple such connections by adding similarly defined *Output / Input nodes* separated by commas. This is graphically suggested by the gray coloured *Output / Input nodes* and their corresponding *Routed signal line*.
  - If  $n$  is greater than the size of the input array, an error message is issued and the simulation is stopped.

### 30.2.7.7 Configuration 7: From an array signal belonging to a set of $n$ models to an array signal of a model

The connection of an array signal belonging to a set of  $n$  models with an array signal of a model is supported as shown in Figure 30.2.17.

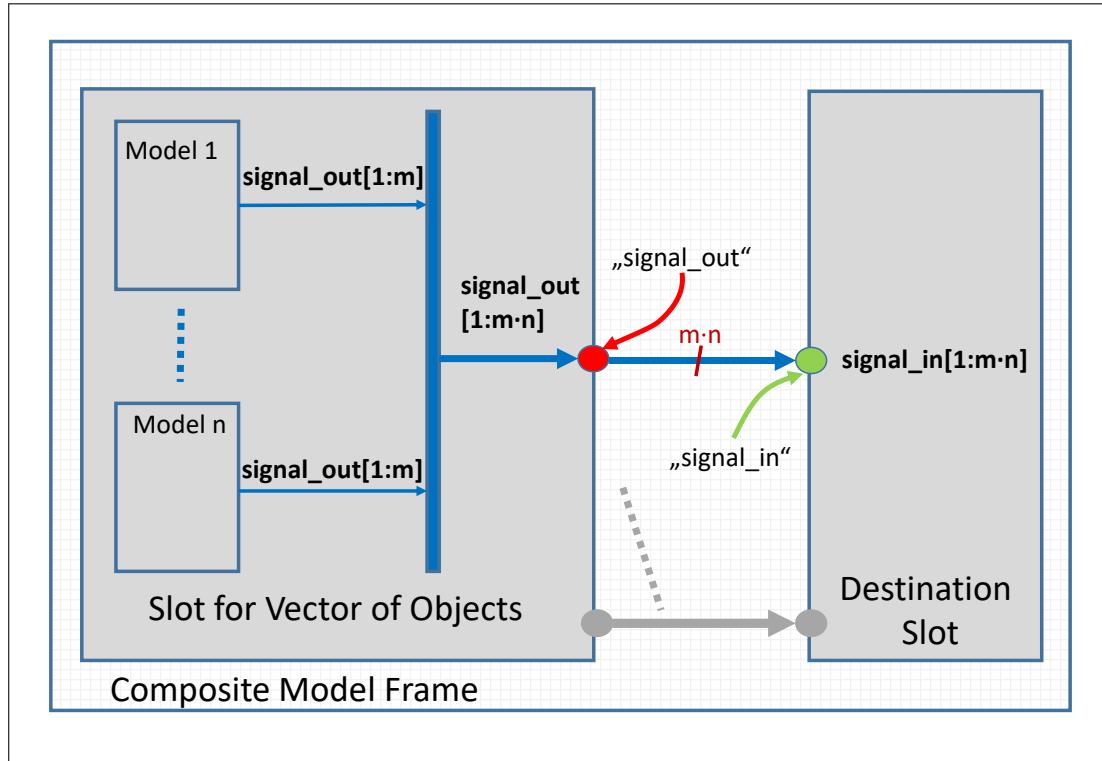


Figure 30.2.17: Configuration 7: From an array signal belonging to a set of  $n$  models to an array signal of a model

- Note:**
- The *Output Signals* text field of the *Slot for Vector of Objects* should contain the name of the source array signal (e.g. "signal\_out"). The name of the source signal must be the same for all  $n$  models of the set.
  - The *Input Signals* text field of the *Destination Slot* should contain the name of the destination array signal (e.g. "signal\_in");
  - The *Slot for Vector of Objects* and *Destination Slot* can be linked with multiple such connections by adding similarly defined *Output / Input nodes* separated by commas. This is graphically suggested by the gray coloured *Output / Input nodes* and their corresponding *Routed signal line*.
  - An error message is issued when the dimension of the input signal is smaller than the total dimension of the output signal (for one element) multiplied by the number of selected elements. Further, an error message is issued when the dimension of the input signal is not an integer of the output signal dimension ( $k$ ). ( $n_{in} \bmod k <> 0$ ).

### 30.2.7.8 Configuration 8: From a set of scalar signals belonging to a set of $n$ models to an array signal of a model

The connection of a set of scalar signals (a multi-variable *Node*) belonging to a set of  $n$  models with an array signal of a model is supported as shown in Figure 30.2.18.

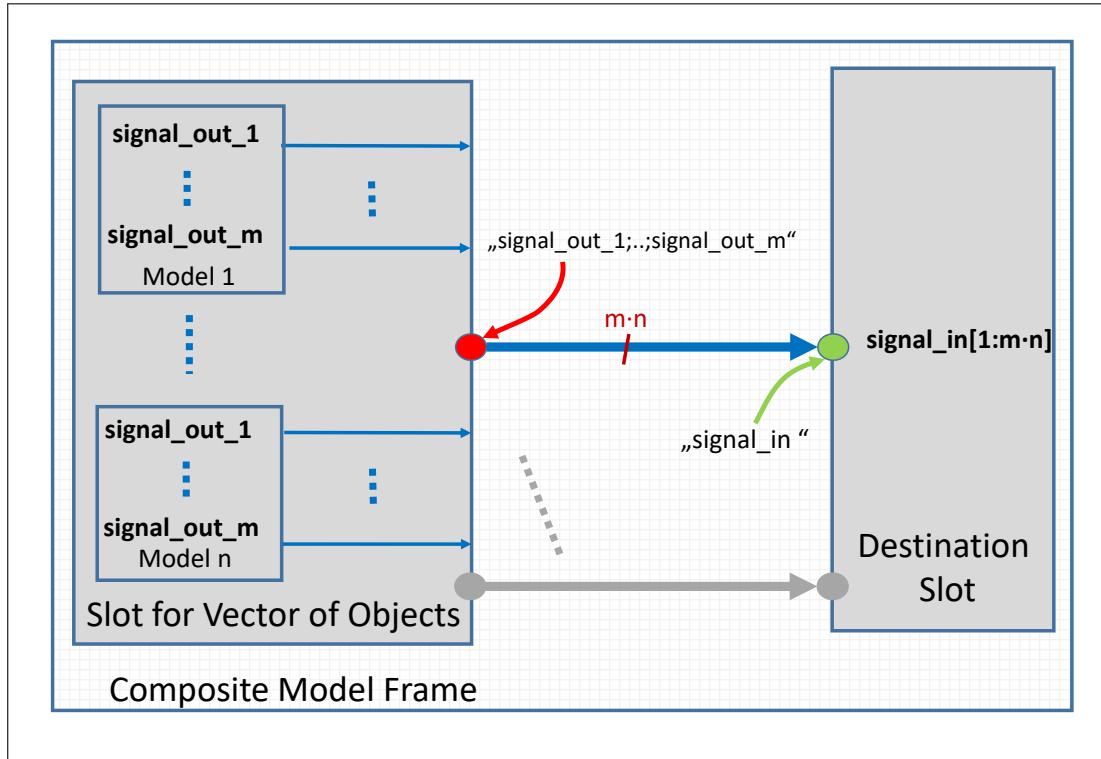


Figure 30.2.18: Configuration 8: From a set of scalar signals belonging to a set of  $n$  models to an array signal of a model

- Note:**
- The *Output Signals* text field of the *Slot for Vector of Objects* should contain the name of the source set of scalar signals (e.g. "signal\_out\_1;..;signal\_out\_m"). The names of the source set of scalar signals must be the same for all  $n$  models of the set.
  - The *Input Signals* text field of the *Destination Slot* should contain the name of the destination array signal (e.g. "signal\_in");
  - The *Slot for Vector of Objects* and *Destination Slot* can be linked with multiple such connections by adding similarly defined *Output / Input nodes* separated by commas. This is graphically suggested by the gray coloured *Output / Input nodes* and their corresponding *Routed signal line*.

### 30.2.7.9 Configuration 9: From a set of scalar signals belonging to a model to an array signal of a model

The connection of a set of scalar signals (a multi-variable *Node*) belonging to a model with an array signal of a model is supported as shown in Figure 30.2.19.

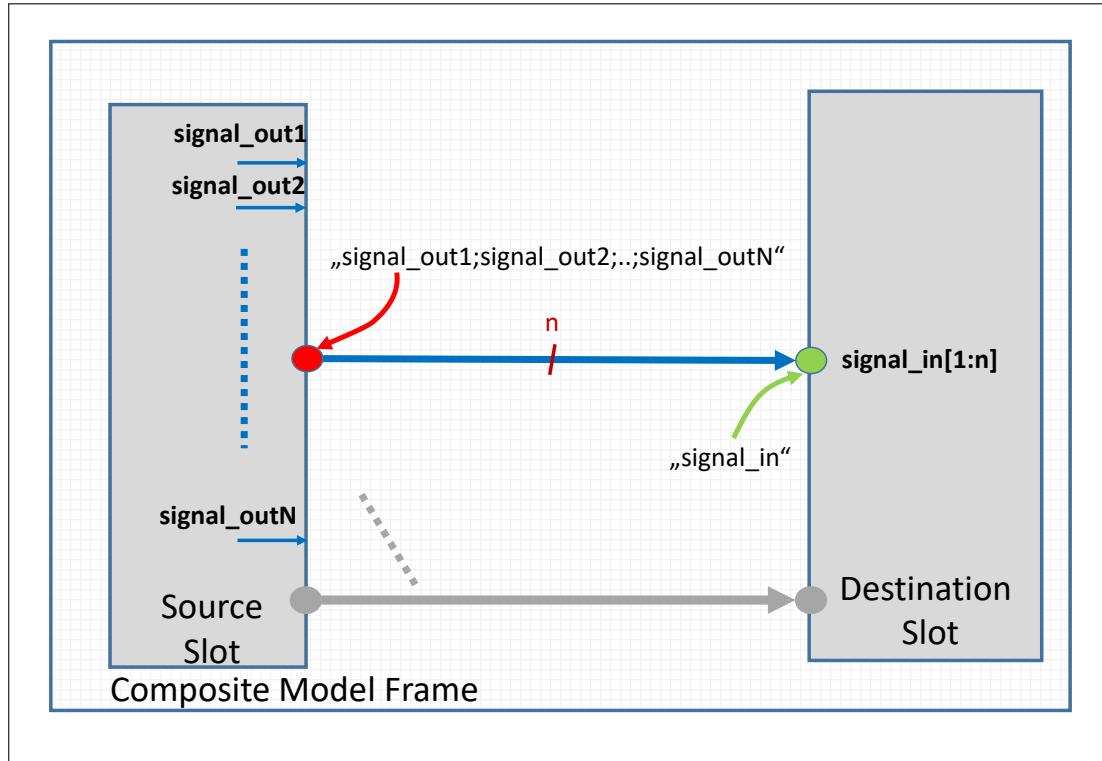


Figure 30.2.19: Configuration 9: From a set of scalar signals belonging to a model to an array signal of a model

- Note:**
- The *Output Signals* text field of the *Signal source Slot* should contain the name of the source set of scalar signals (e.g. “`signal_out1;..;signal_outN`”);
  - The *Input Signals* text field of the *Destination Slot* should contain the name of the destination array signal (e.g. “`signal_in`”);
  - The *Signal source Slot* and *Destination Slot* can be linked with multiple such connections by adding similarly defined *Output / Input nodes* separated by commas. This is graphically suggested by the gray coloured *Output / Input nodes* and their corresponding *Routed signal line*.
  - An error message is issued if the total dimension of the output signal does not match the dimension of the input signal.

### 30.2.7.10 Configuration 10: From scalar signals belonging to a set of $n$ models to scalar signals of a set of $n$ other models

The connection of scalar signals belonging to a set of  $n$  models with scalar signals belonging to a set of  $n$  other models is supported as shown in Figure 30.2.20.

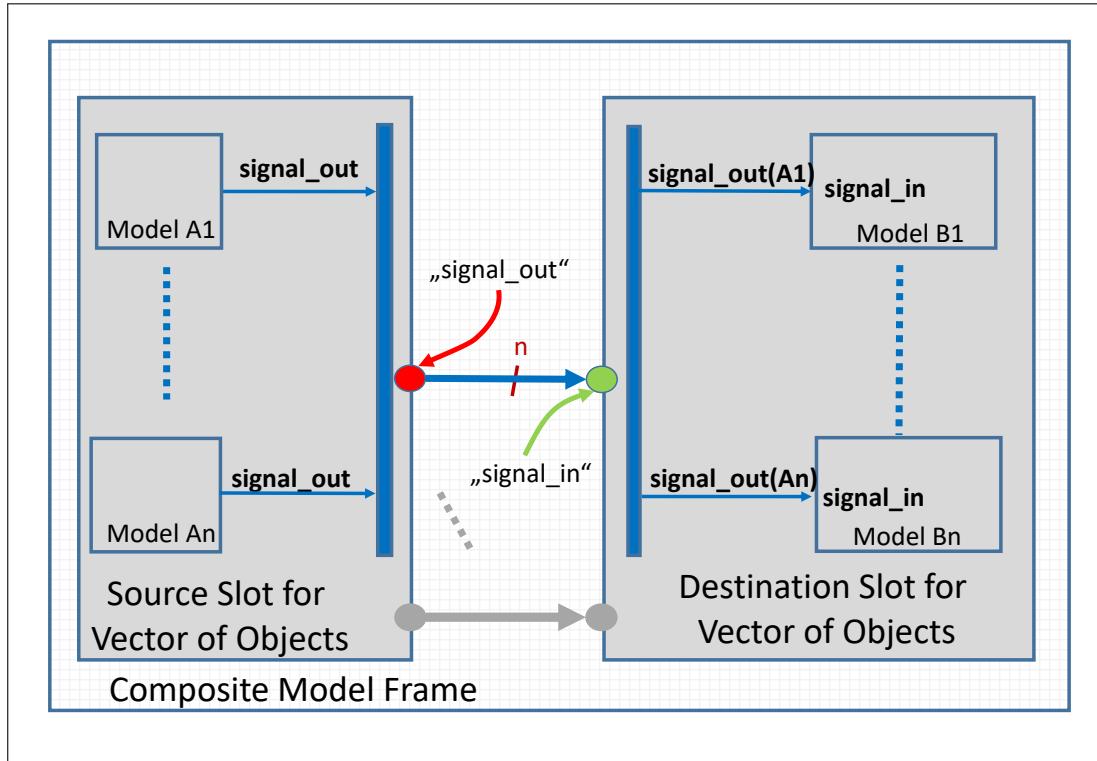


Figure 30.2.20: Configuration 10: From scalar signals belonging to a set of  $n$  models to scalar signals of a set of  $n$  other models

**Note:**

- The *Output Signals* text field of the *Source Slot for Vector of Objects* should contain the name of the source scalar signal (e.g. “signal\_out”). The signal name must be the same for all  $n$  models of the set.
- The *Input Signals* text field of the *Destination Slot for Vector of Objects* should contain the name of the destination scalar signal (e.g. “signal\_in”). The signal name must be the same for all  $n$  models of the set.
- The *Source Slot for Vector of Objects* and the *Destination Slot for Vector of Objects* can be linked with multiple such connections by adding similarly defined *Output / Input nodes* separated by commas. This is graphically suggested by the gray coloured *Output / Input nodes* and their corresponding *Routed signal line*.
- An error message is issued if one of the signals is an array (source or destination).

### 30.2.7.11 Configuration 11: From one scalar signal (or one index of an array signal) of one model to one scalar signal belonging to a set of $n$ models

The connection of one scalar signal (or one index of an array signal) of one model with one scalar signal belonging to a set of  $n$  models is supported as shown in Figure 30.2.21.

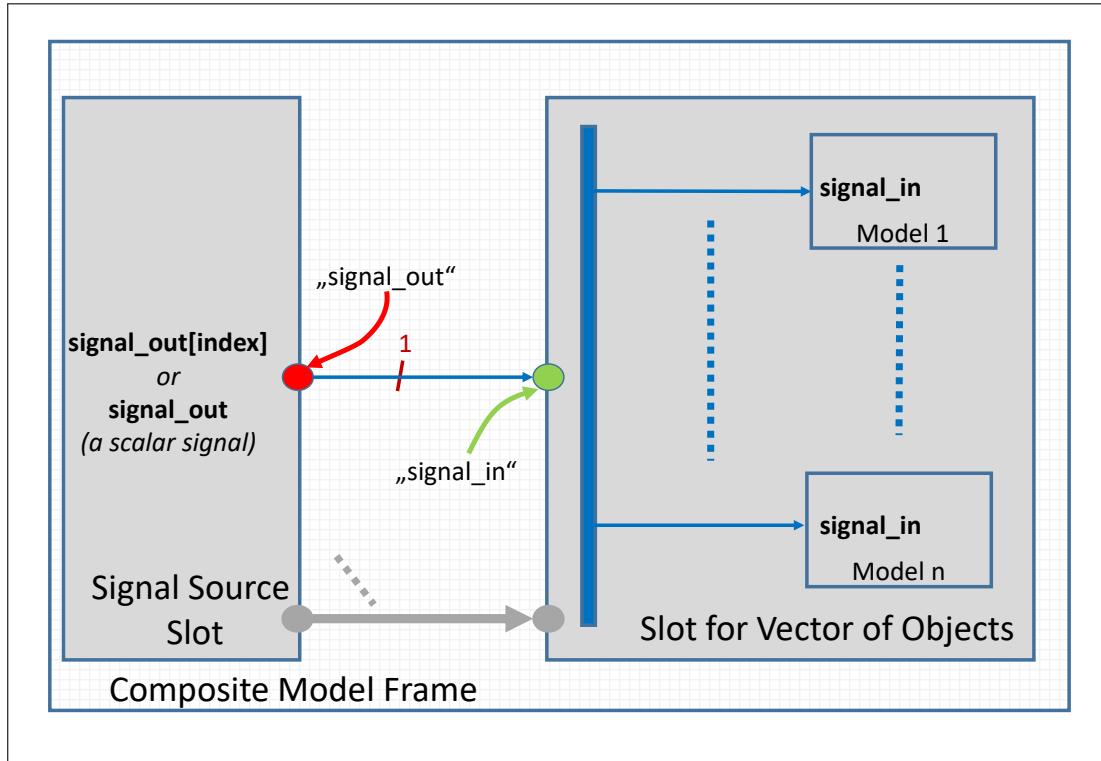


Figure 30.2.21: Configuration 11: From one scalar signal (or one index of an array signal) of one model to one scalar signal belonging to a set of  $n$  models

- Note:**
- The *index* in *array[index]* is an integer number corresponding to the array index (0-based) being extracted from the whole *array* variable; Pay special attention to interconnecting *Modelica Models* in a high-level control system, as array variables (and therefore input and output signals as well) are 1-based in Modelica. For example, in a Modelica model, the second index of an array variable is referred to by “my\_array\_variable[2]” (i.e. 1-based). In a *Slot*, in order to extract the second index of the *Modelica Model* array variable “my\_array\_variable” the *Input* or *Output Node* must contain “my\_array\_variable[1]” (i.e. 0-based).
  - The *Output Signals* text field of the *Signal source Slot* should contain the name of the source array signal (e.g. “signal\_out” (if a scalar) or “signal\_out[3]” (if an array whose fourth value is extracted));
  - The *Input Signals* text field of the *Slot for Vector of Objects* should contain the name of the destination scalar signal (e.g. “signal\_in”). The destination signal name must be the same for all  $n$  models of the set.
  - The *Signal source Slot* and the *Slot for Vector of Objects* can be linked with multiple such connections by adding similarly defined *Output / Input nodes* separated by commas. This is graphically suggested by the gray coloured *Output / Input nodes* and their corresponding *Routed signal line*.
  - An error message is issued if the input signal is an array or no index is defined when the output signal is an array.

### 30.2.7.12 Configuration 12: From one index of an array signal of one model to one scalar signal belonging to one model

The connection of one index of an array signal of one model with one scalar signal belonging to one model is supported as shown in Figure 30.2.22.

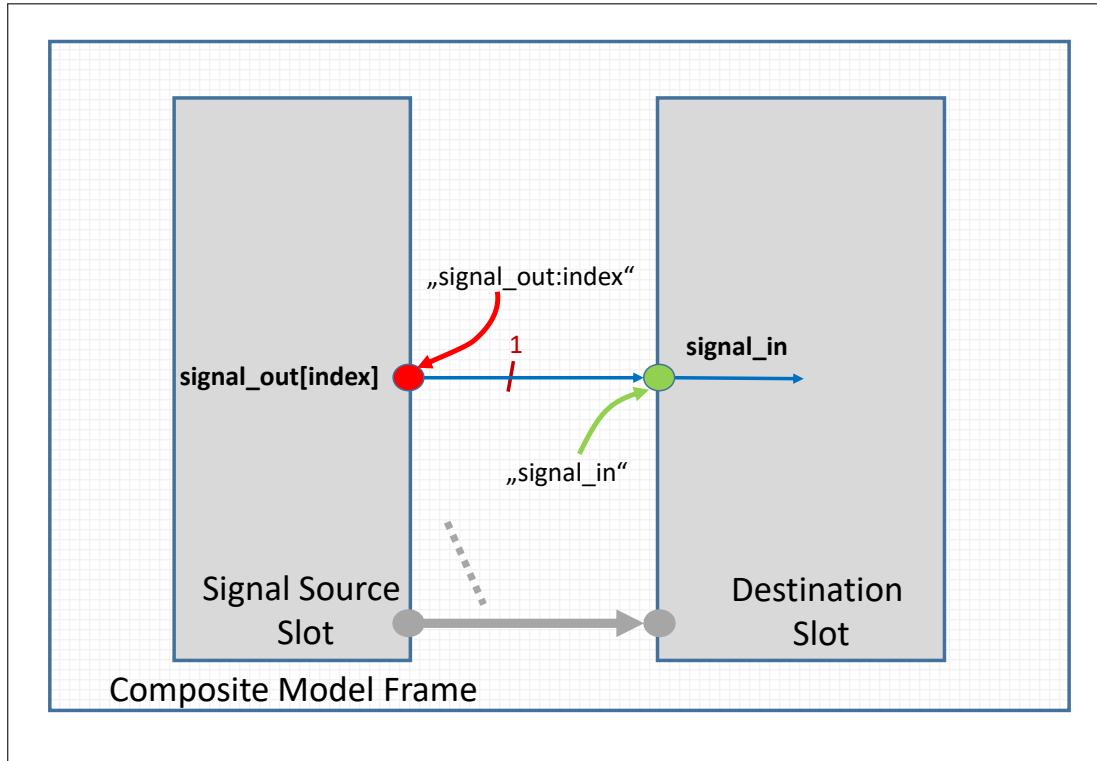


Figure 30.2.22: Configuration 12: From one index of an array signal of one model to one scalar signal belonging to one model

- Note:**
- The *index* in *array[index]* is an integer number corresponding to the array index (0-based) being extracted from the whole *array* variable; Pay special attention to interconnecting *Modelica Models* in a high-level control system, as array variables (and therefore input and output signals as well) are 1-based in Modelica. For example, in a Modelica model, the second index of an array variable is referred to by “*my\_array\_variable[2]*” (i.e. 1-based). In a *Slot*, in order to extract the second index of the *Modelica Model* array variable “*my\_array\_variable*” the *Input* or *Output Node* must contain “*my\_array\_variable:1*” (i.e. 0-based).
  - The *Output Signals* text field of the *Signal source Slot* should contain the name of the source array signal and its index (e.g. “*signal\_out:2*”);
  - The *Input Signals* text field of the *Destination Slot* should contain the name of the destination scalar signal (e.g. “*signal\_in*”);
  - The *Signal source Slot* and *Destination Slot* can be linked with multiple such connections by adding similarly defined *Output / Input nodes* separated by commas. This is graphically suggested by the gray coloured *Output / Input nodes* and their corresponding *Routed signal line*.
  - An error message is issued if no array index is defined.

### 30.2.7.13 Example of an individually dispatched Power Plant Controller for wind turbines

As an example, a typical use case scenario is the individual dispatch of Wind Turbine Generators (WTGs) by a Power Plant Controller (PPC). The WECC standard PPC system is used as reference in this example. The standard WECC PPC high-level control system is by design intended for non-individual dispatch of WTGs by a PPC, as it is apparent in Figure 30.2.23, which depicts the diagram of the corresponding *Composite Model Frame*. The output signals  $Q_{ext}$  and  $P_{ref}$  are scalar quantities, sourced from the *Plant Level Control* slot. This slot is a placeholder for a *Power Plant Controller Model*, in this case implemented using DSL. From a *PowerFactory* integration perspective, the design of the shown *Composite Model Frame* implies that the relevant PPC *Composite Model* must be assigned to each WTG within the corresponding WTG's *Composite Model* (a diagram of the WTG's *Composite Model Frame* is shown in Figure 30.2.4, where the "Plant Control" slot is intended for the PPC). Furthermore, the output signals  $Q_{ext}$  and  $P_{ref}$  are not individual dispatches for each WTG (they are single-valued quantities).

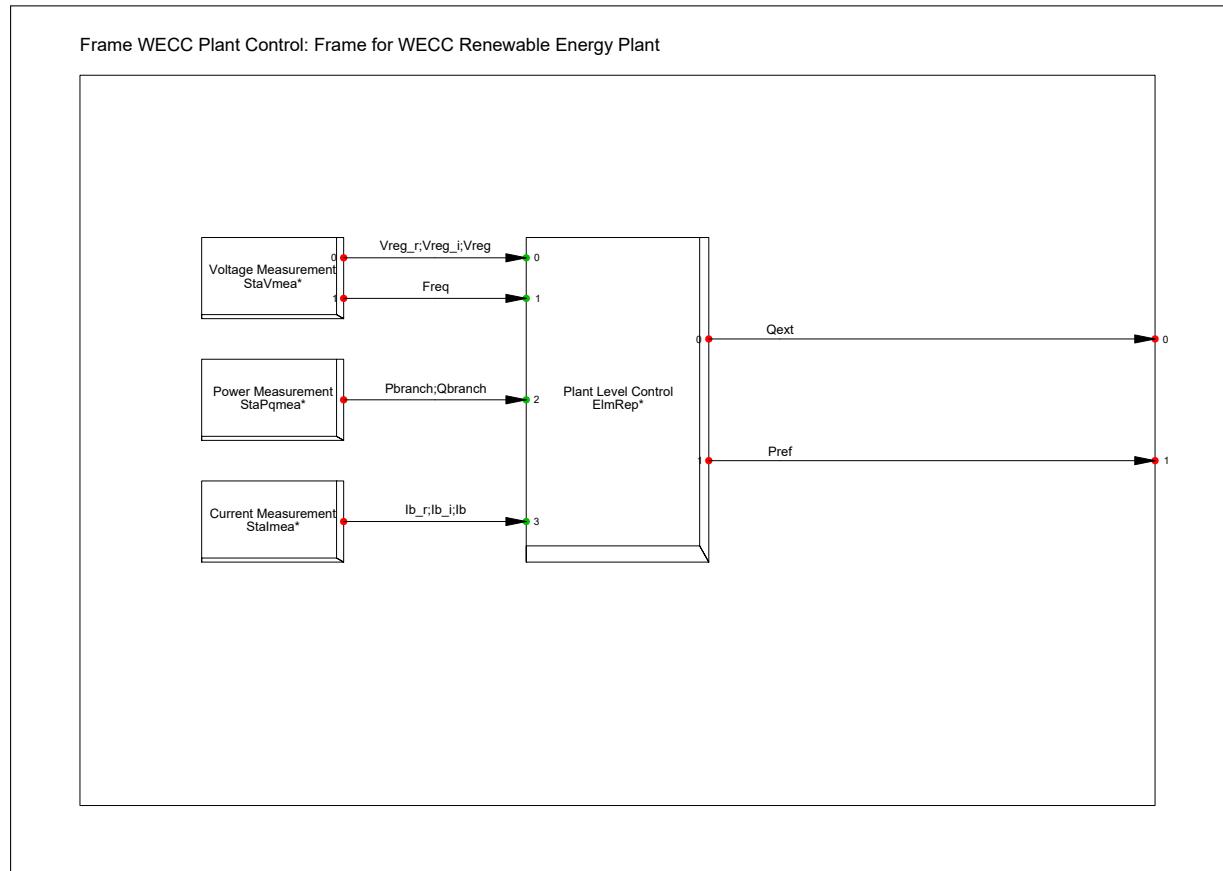


Figure 30.2.23: WECC (standard) PPC frame with non-individual dispatch of WTGs

For detailed, manufacturer specific PPC implementations, an individual generator dispatch is nevertheless typical. A modified version of the standard WECC PPC high level control system (*Composite Model Frame*) is shown in Figure 30.2.24. The modified version of the WECC PPC system demonstrates the individual dispatching of WTGs by using the *Vector of Objects* element (*IntVecobj*). An additional slot is created in the new *Composite Model Frame*, which is a placeholder for the *Vector of Objects* element.

The signals which are to be conveyed (distributed) to the WTGs are to be declared in the *Input* field of this new *Slot*. Note that, although identically depicted in the diagram, the signals to be distributed (in the example, signals  $Q_{ext}$  and  $P_{ref}$ ) are required to be vectors (one-dimensional array signals). Such vector signals can be generated using a *Modelica Model* (DSL Models support only scalar signal outputs). Their size must be equal to  $n \cdot m$ , i.e. the number of elements assigned in the *Vector of Objects* ( $n$ ) times the size of each individual destination signal ( $m$ ). In the trivial case of a destination

signal being a scalar, the size of each vector must be  $n$ .

The signals which are to be extracted from the WTGs are to be declared in the *Output* field of the *Slot*. The extracted signals (in the example, signals  $QMIN$  and  $QMAX$ ) are vectors (one-dimensional array signals). Their size is equal to  $n \cdot m$ , i.e. the number of elements assigned in the *Vector of Objects* ( $n$ ) times the size of each individual source signal ( $m$ ).

It is apparent that in this use case scenario, [Configuration 2](#) was used to convey array signals from the PPC to the WTGs and [Configuration 6](#) was used to extract signals from the WTGs and send them to the PPC.

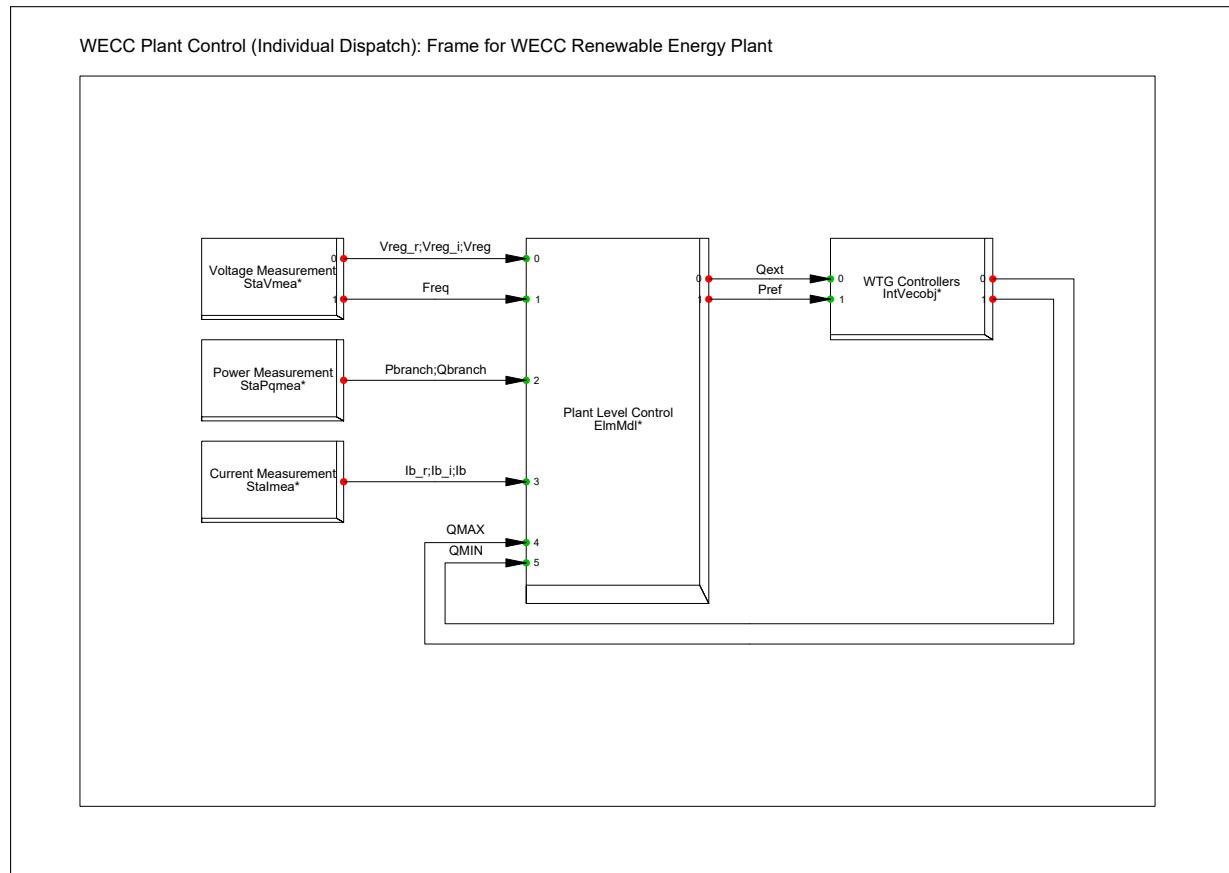


Figure 30.2.24: WECC (modified) PPC frame with individual dispatch of WTGs

The *Vector of Objects* element is created in the corresponding *Composite Model* (of the PPC) and assigned to this new *Slot*. The *Vector of Objects* must be configured to contain those control elements that are to receive the signals. In the case of the WECC standard Type 4A wind turbine model, the relevant model is the *DSL Model* “REEC\_A Electrical Control Model”. The configuration of the *Vector of Objects* element for a wind power plant containing four WTGs is shown in Figure 30.2.25. The order within the configuration dialog of the *Vector of Objects* element is important! The demultiplexing logic parses all these objects in ascending order (i.e. first element in the table is processed first, second element is processed next, and so on).

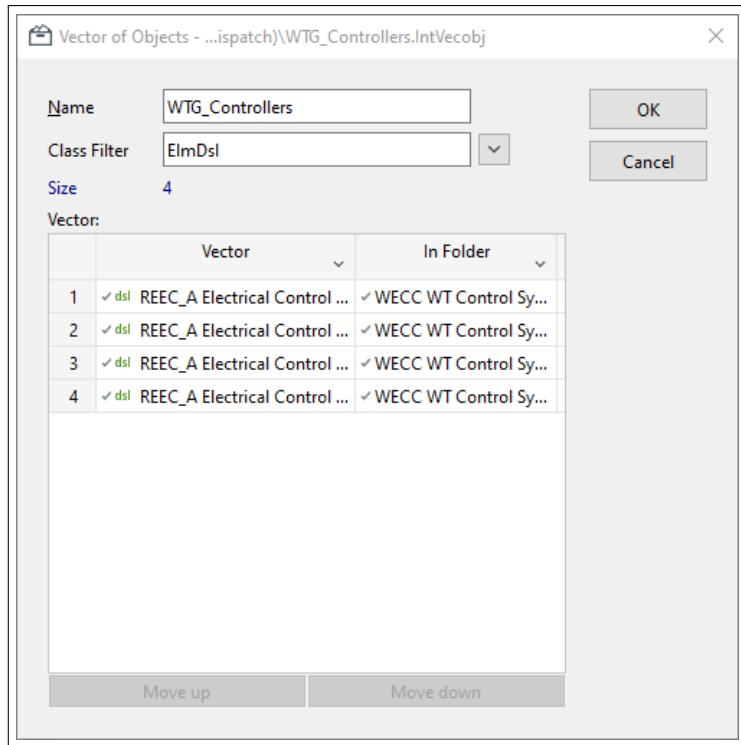


Figure 30.2.25: Assignment of WTG controllers to the *Vector of Objects*

**Note:** The tabular entry list of the *Vector of Objects* dialog has the following functionality:

- Multiple subsequent rows can be selected by holding down the **Shift** key and then clicking rows of interest;
- The order can be changed using buttons **Move up** and **Move down**: select a row and then click on one of these buttons - the selected element will consequently be moved up/down.
- Selected rows can be moved via *Drag & drop* (to start dragging, the mouse cursor has to be placed over a cell of one of the two columns).

In the case at hand, the following applies:

- The source signals are the vector signals *Qext* and *Pref* of the *PPC Model*, having each a size  $4 \cdot 1 = 4$ . The *PPC Model* is implemented using a *Modelica Model* (*ElmMdl*).
- The destination signal is stated within the *Input Signals* text field of the slot assigned to the corresponding *Vector of Objects* element within the *Composite Model Frame*. The signals are named *Qext* and *Pref* and represent the active and reactive power dispatches of the WTG controller, having a size 1;
- The destination elements are programmed within the configuration dialog of the *Vector of Objects* element (refer to Figure 30.2.25). Within this dialog there are 4 individual *WTG Controller* elements added (*DSL Models*).
- The signal demultiplexing logic is as follows:
  - the first destination element is the “*REEC\_A Electrical Control Model*” (a *DSL Model*) of the first WTG; the first index of each source signal *Qext* and *Pref* will be assigned (in the same order) to the destination components *Qext* and *Pref* (which are scalars) of the “*REEC\_A Electrical Control Model*” of the first WTG.
  - the second destination element is the “*REEC\_A Electrical Control Model*” (a *DSL Model*) of the second WTG; the second index of each source signal *Qext* and *Pref* will be assigned (in the same order) to the destination components *Qext* and *Pref* (which are scalars) of the “*REEC\_A Electrical Control Model*” of the second WTG.

- the procedure continues until: (a) all destination elements have been parsed or (b) the last index of the source signal has been reached.

### 30.2.8 Using power system measurements in a high-level control system

Retrieving power system measurements (e.g. current, voltage and power) from specific points in the network and linking them with a “Control system” is done by creating (and configuring) measurement devices. The following measurement devices are typically used in a *high-level control system*:

- Current Measurement (*Stalmea*)
- Voltage Measurement (*StaVmea*)
- PQ Measurement (*StaPqmea*)
- PLL Phase/Frequency Measurement (*StaPll*)

As described in Section 30.2.2, a *User-defined* model can be connected with *built-in* elements via predefined sets of input or output signals using a *Composite Model Frame*. Each element class allows for a specific set of signals that can be connected within a high-level control system. Although not specifically forbidden, linking variables which are not input or output signals of elements should be avoided as much as possible. This applies in particular to attributes which change during simulation. For example, the electrical quantities available at busbar or branch elements should never directly be used in a *Composite Model Frame*. Instead, the measurement devices should be used.

To create a new measurement device, do the following:

- from the single line diagram:
  - locate the measurement point. For measuring voltage, current and power, a *cubicle* of a terminal should be selected.
  - Right-click the target cubicle and choose *Devices* → <Measurement Type> → *New...*, where the “<Measurement Type>” is one of the allowed device types, as provided in the selection menu.
  - The object is created inside the respective cubicle. Note that, if this measurement device is moved to a different location, the measurement point is not automatically saved. This implies that the corresponding field will need to be manually updated.
- from the Data Manager:
  - Navigate to the location where the measurement device is to be placed. Typically, this can either be the targeted measurement point i.e. a terminal cubicle or the *Composite Model* object (which in this sense acts as a folder that stores the measurement devices).
  - From the Data Manager toolbar, click *New Object* ().
  - Select the measurement device of interest from the options shown, which can be filtered if necessary using the device or class name of the measurement device.
- Apply measurement device configuration settings according to the specific application. Refer to the [Technical Reference documentation](#) for detailed information.

---

**Note:** The object for PLL Phase/Frequency Measurements *StaPll* replaces the *ElmPhi\_pll* object. It is recommended to use the new *StaPll* object, which encompasses all previously available functionalities of the *ElmPhi\_pll* object as well as some additional enhancements. Please refer to the [Technical Reference documentation](#) for a detailed description of the model. The outdated *ElmPhi\_pll* object can still be used in *PowerFactory* but will not be further developed.

---

To include a measurement device in a high-level control system (*Composite Model Frame*), do the following:

- Identify the output signal(s) name(s) of the measurement device required to be linked in the designed *Control System*.

**Note: How to identify the output signals (and their names) of a measurement device in order to correctly set the outputs of a corresponding “measurement” slot?**

- Check the Technical Reference documentation of the specific measurement device (refer to the “Measurement Devices” section of the [Technical Reference Overview](#)) document. Alternatively, the *Technical Reference Overview* is accessible in *PowerFactory* by selecting from the main menu *Help* → *Technical References* → *Models*.
  - Alternatively, use the Data Manager or the Network Model Manager to navigate to an element of the same class as the targeted measurement device.
  - Use the *Flexible Data* tab and open the *Variable Selection* window (e.g. right-click on the *Flexible Data* tab and choose *Variable selection*).
  - From the *Variable Selection* window, use the filter *Group* to get all the *Signals and States*.
  - A listing of all available signals is provided. From the available list, use only the outputs (type *OUT*). The outputs (type *OUT*) can be set in the *Output Signals* field of a *Slot*. These signals can therefore be taken from the measurement device and further provided to other components placed in *Slots*. Note, all signals in *PowerFactory* are case sensitive.
- 
- Create a corresponding *Slot* in the *Composite Model Frame* to which the measurement device is to be later assigned.
  - Update the slot field *Output* with a list of output signals that are to be extracted from the measurement device (e.g. for a voltage measurement, the signals *ur* and *ui* represent the real and imaginary part of the positive sequence voltage in an RMS simulation - therefore, the *Output Signals* field could be updated with the text “*ur,ui*”).
  - Link the measurement signals of this slot with any other relevant input signals of slots in the control system.
  - When the *Composite Model Frame* is finalised, the typical approach of assigning the measurement device element to a corresponding slot of a *Composite Model* is to be followed (refer to Section [30.2.6](#)).

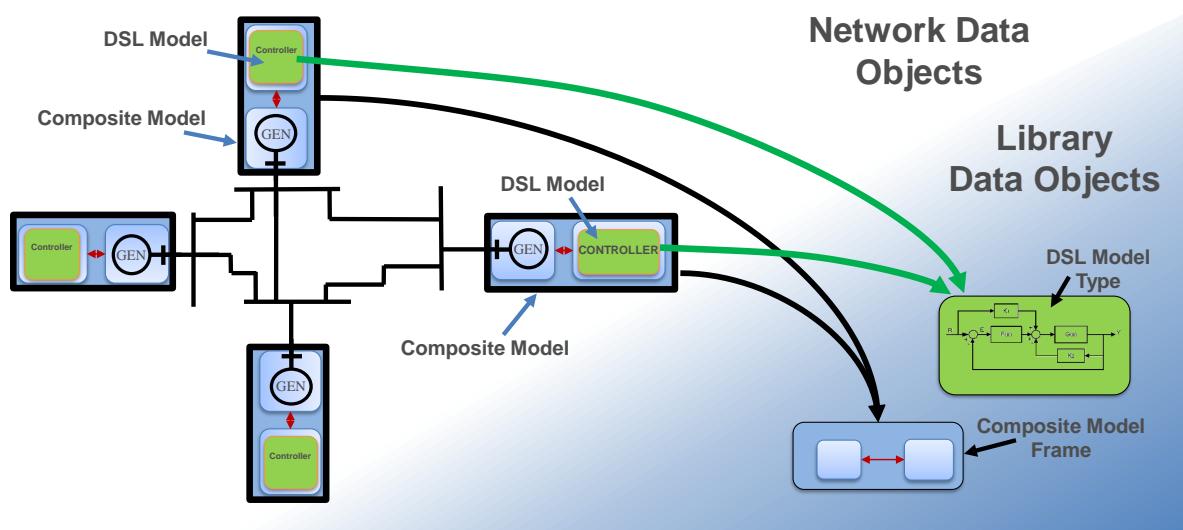
## 30.3 DSL: Integrating *DSL Models* into a Simulation

### 30.3.1 *DSL Models* and *DSL Model Types*

Dynamic models, as discussed in Section [30.1](#), can be of various types. Invariably, the implementation of *User-defined* dynamic models is realised using one of the following two options:

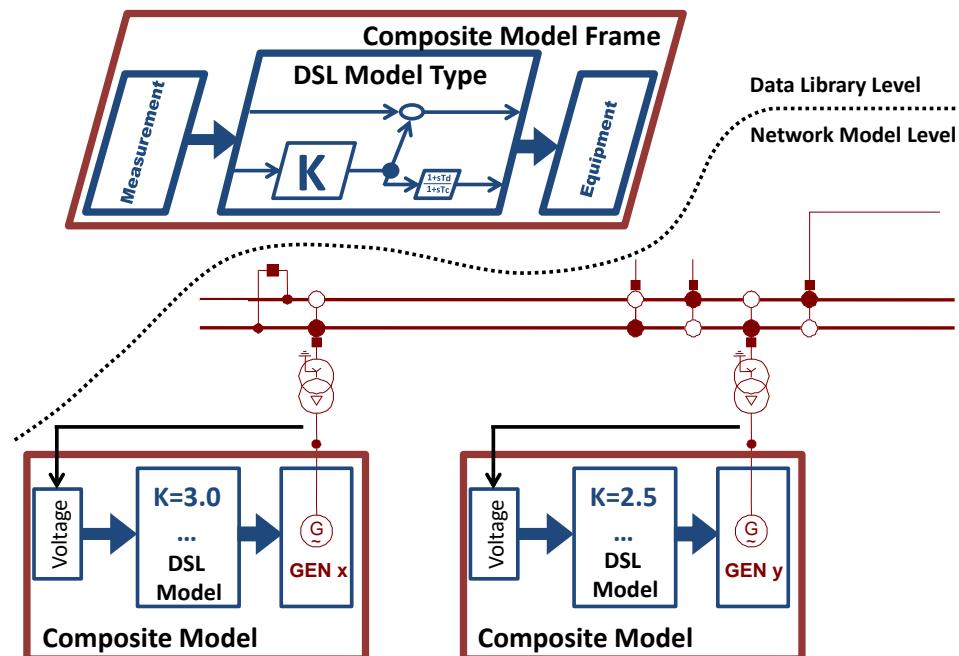
- using DSL specific objects: *DSL Model* (*ElmDsl* `dsl`) and *DSL Model Type* (*BlkDef* `dsl`), or
- using Modelica specific objects: *Modelica Model* (*ElmMdl* `mdl`) and *Modelica Model Type* (*TypMdl* `mdl`)

If discussing the first option, the implementation of a *User-defined* dynamic model follows the modelling paradigms defined by the *DIGSILENT Simulation Language*, as explained in detail in Section [30.4](#). The integration of a DSL dynamic model conceptually follows the same *Type/Element* principle found throughout *PowerFactory*, and is graphically depicted in Figure [30.3.1](#). It is apparent that the *DSL Model* (*ElmDsl*, `dsl`) represents one instance in the power system of a user-defined *DSL Model Type* (*BlkDef* `dsl`). A *DSL Model* is integrated within a slot of a *Composite Model* (a high-level control system) in order to be interconnected with other control relevant components, with which the *DSL Model* will exchange input and output signals, as defined in the *Composite Model Frame*.

Figure 30.3.1: Integration of *DSL Models* into a Simulation

A *DSL Model* instantiates a *DSL Model Type* with a specific set of parameter values. The initial parameter values are either zero or the default ones, if existing. Default parameter values are those pre-configured parameter values of a *DSL Model* saved (as a copy) inside the corresponding *DSL Model Type*.

The benefit of using a *DSL Model* that stores the actual model parameter values is illustrated in Figure 30.3.2. It is apparent that two individual *DSL Models* can be defined and individually parameterised while referencing the same controller structure (the same *DSL Model Type*) and controlling different generators.

Figure 30.3.2: Individual model parameterisation using *DSL Models*

Furthermore, *DSL Models* have the following characteristics:

- Any number of *DSL Models* can be defined in a power system model.

- A *DSL Model* can reference only *DSL Model Type*
- A *DSL Model Type* can be referenced multiple times, by any number of *DSL Models*
- A single *DSL Model* can be assigned to one *Slot* of a given Composite Model
- Although rarely the case, one single *DSL Model* can be assigned to *Slots* belonging to multiple Composite Models. In such a case, the *Model developer* needs to make sure that the input signals that are connected to the *DSL Model* are unique e.g. it is not allowed to connect the same input signal twice, once with a component of one Composite Model, then with another one of a different Composite Model.

Two types of parameters are supported by a *DSL Model*:

- Scalar parameters e.g. gains, time constants, set-points,
- Array parameters: one- and two-dimensional array parameters.

If the referenced *DSL Model Type* defines one or more arrays (characteristics), then these arrays can be customised within the *DSL Model*, using references to *IntMat* objects (recommended approach). The *IntMat* objects will contain the actual arrays. Alternatively, arrays can be customised using the tab “Advanced 1” (for one-dimensional arrays) and “Advanced 2” (for two-dimensional arrays) of the Edit Dialog (legacy feature).

#### **Characteristic parameterisation using references to *IntMat* objects (recommended approach)**

DSL lookup Arrays/Matrices may be defined externally, using *IntMat* objects, provided that the array variable names are declared using the prefix *oarray\_* or *omatrix\_*:

- *oarray\_* - (e.g. “*oarray\_K*”), external array, this option will use an external array characteristic for a one dimensional function  $y = f(x)$ , where  $x$  and  $y$  characteristic value pairs are defined in two columns. The array characteristic must be defined within an external object of type *IntMat*. The *IntMat* object has to be stored inside the *DSL Model* (i.e. as child object). The object name must match the string following the *\_*. If this is done, then *PowerFactory* automatically assigns the array object to the variable *oarray\_*.... For example, if “*oarray\_K*” is declared in the DSL block definition parameter list, then the *IntMat* object must be named “*K*” and stored inside the *DSL Model* which defines variable “*oarray\_K*”. A combination of internal and external arrays (i.e. mixing array\_ and oarray\_ variables) within a single *DSL Model Type / DSL Model* is not supported. The “*IntMat*” object must have suitable dimensions; this means two columns and at least two rows for linear approximation based functions and at least three rows in case of spline approximation functions.
- *omatrix\_* - (e.g. “*omatrix\_WindPowerCoefficient*”), external matrix, this option will use an external matrix characteristic for a two-dimensional function  $y = f(xr, xc)$ , where  $xr$  and  $xc$  are the row and column arguments respectively, while  $y$  is the function output value. The matrix characteristic must be defined within an external object of type *IntMat*. The *IntMat* object has to be stored inside the *DSL Model* (i.e. as child object). The name has to match the string, following the *\_*. For example, in the case of “*omatrix\_WindPowerCoefficient*” DSL variable, the *IntMat* object would have to be named “*WindPowerCoefficient*” and stored inside the *DSL Model*. If this is done, then *PowerFactory* automatically assigns the matrix object to the variable *omatrix\_*.... A combination of internal and external arrays (i.e. mixing matrix\_ and omatrix\_ variables) within a single *DSL Model Type / DSL Model* is not supported. The “*IntMat*” object must have suitable dimensions, i.e. at least three columns and at least three rows for linear approximation based functions and at least four rows and four columns in case of spline approximation functions. The top row must contain the  $xr$  values, the left column must contain the  $xc$  values. Both rows and columns must be defined in ascending order for proper operation.

#### **Characteristic parameterisation using tabular field entries of the *DSL Model* (legacy feature)**

DSL lookup Arrays/Matrices may be defined internally, using the available tabular field entries, provided that the array variable names are declared using the prefix *array\_* or *matrix\_*:

- *array\_* - (e.g. “array\_K”), internal array, this option will create an array characteristic for a one dimensional function  $y = f(x)$ , where  $x$  and  $y$  characteristic value pairs are defined in two columns. The array characteristic is defined within the *DSL Model* (n.b. not within the *DSL Model Type*) should at least one such variable name exist. Internal arrays may be defined in the tab “Advanced 1” of the “Basic Data” page of the *DSL Model* edit dialog. The length of the array is defined via the entry in the left cell of the first row. After adding the dimension an update is needed; this can be done by switching back to the “Basic Data” page and then again to the “Advanced 1” page.
- *matrix\_* - (e.g. “matrix\_WindPowerCoefficient”), internal matrix, this option will create a two-dimensional characteristic for a two-dimensional function  $y = f(xr, xc)$ , where  $xr$  and  $xc$  are the row and column arguments respectively and  $y$  the function output value. The matrix characteristic is defined directly within the *DSL Model* (n.b. not within the *DSL Model Type*) should at least one such variable name exist. Internal matrices are defined in the tab “Advanced 2” of the “Basic Data” page of the *DSL Model* edit dialog. The dimension of the matrix is defined via the entry in the first two left cells of the first row. After adding the dimension an update is needed; this can be done by switching back to the “Basic Data” page and then again to the “Advanced 2” page.

The characteristics are defined as follows:

**One-Dimensional Characteristic** In the row labelled *Size*, insert the number of rows in the first cell; the number of columns is set automatically. If the number of rows is changed, jump to the previous page and back again to update the characteristic.

**Two-Dimensional Characteristic** In the row labelled *Size*, insert the number of rows in the first cell and the number of columns in the second cell. If one of these numbers is changed, jump to the previous page and back again to update the characteristic.

In Figure 30.3.3 an example is shown for an array parameterisation.

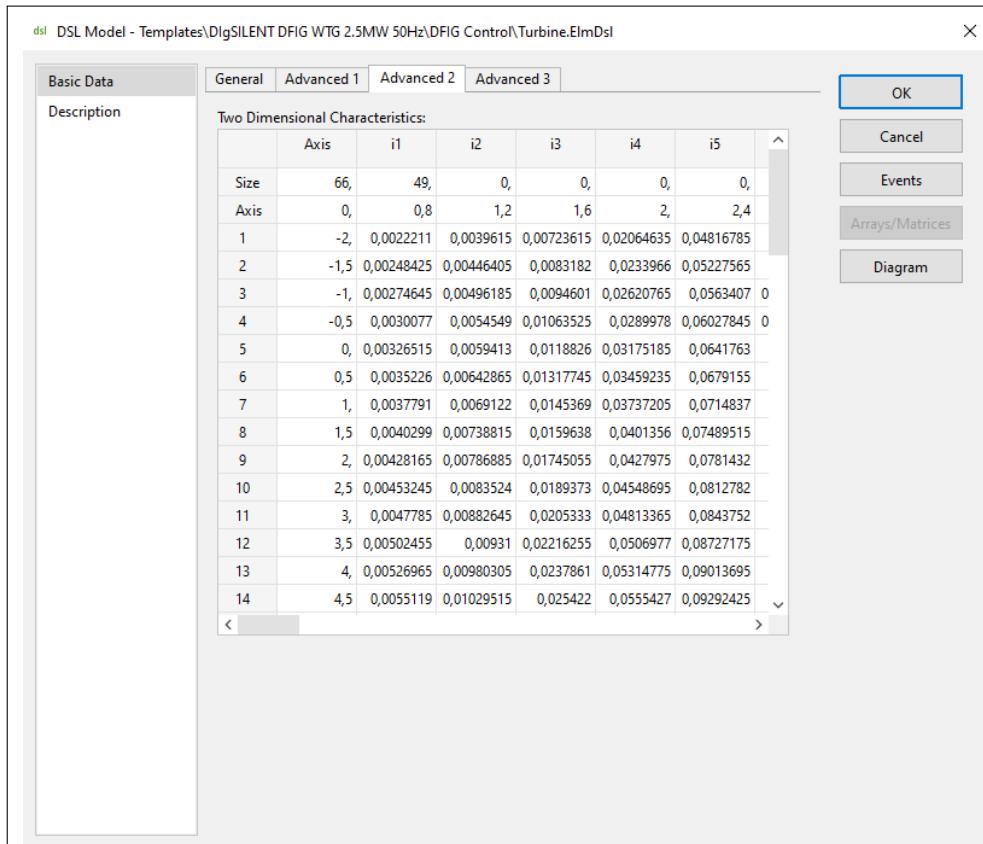


Figure 30.3.3: *DSL Model* containing a two-dimensional characteristic

### 30.3.2 Creating a *DSL Model*

To create a new *DSL Model* (*ElmDsl dsl*), do the following:

- using the Data Manager:
  - Make sure that the Data Manager active directory is inside the folder “Network Model”
  - Click the *New Object* (  ) icon and select *DSL Model*.
  - Give the model a relevant name.
  - Assign to the “Type” field a corresponding *DSL Model Type*

### 30.3.3 Saving and plotting model variables

It is many times useful to record *DSL Model* variables for various purposes e.g. plotting variable against time. To record one or several *DSL Model* variables, they need to be explicitly defined as *monitored variables* belonging to a *Results* object (*ElmRes*). To display the current values of model variables, the *Network Model Manager* can be used. The process of monitoring results in a dynamic simulation is described in detail in Section 29.5.1. The process of plotting the monitored variables is detailed in Section 29.8.

#### Overlaying results of model variables in the block diagram of the *DSL Model*

For graphically defined *DSL Models* (refer to Section 30.6), it is possible to show the values of model variables directly on the block diagram, as follows:

- Use the *Data Manager* or the *Network Model Manager* to navigate and identify to the *DSL Model* of interest.
- Right-click the *DSL Model* and choose *Diagrams* → *Show Diagram*. The block diagram (if existing) of the referenced *DSL Model Type* is shown, along with the current values of the graphically labeled model variables. By default, the shown variables are: input and output signals, internally routed signals, model parameters.

---

**Note:** The displayed values are relevant to the current simulation time. That is, if the simulation is reset  or the simulation has not been started yet, then no values are displayed. If the simulation has been initialised, then the initial conditions values are shown. If the simulation is run up to  $x$  seconds, then the values of the model variables at the simulation time  $x$  seconds are shown.

---

### 30.3.4 Example of a *DSL Model* implementation

An example of a graphically defined *DSL Model Type* is shown in Figure 30.3.5, where a simple excitation voltage controller is implemented.

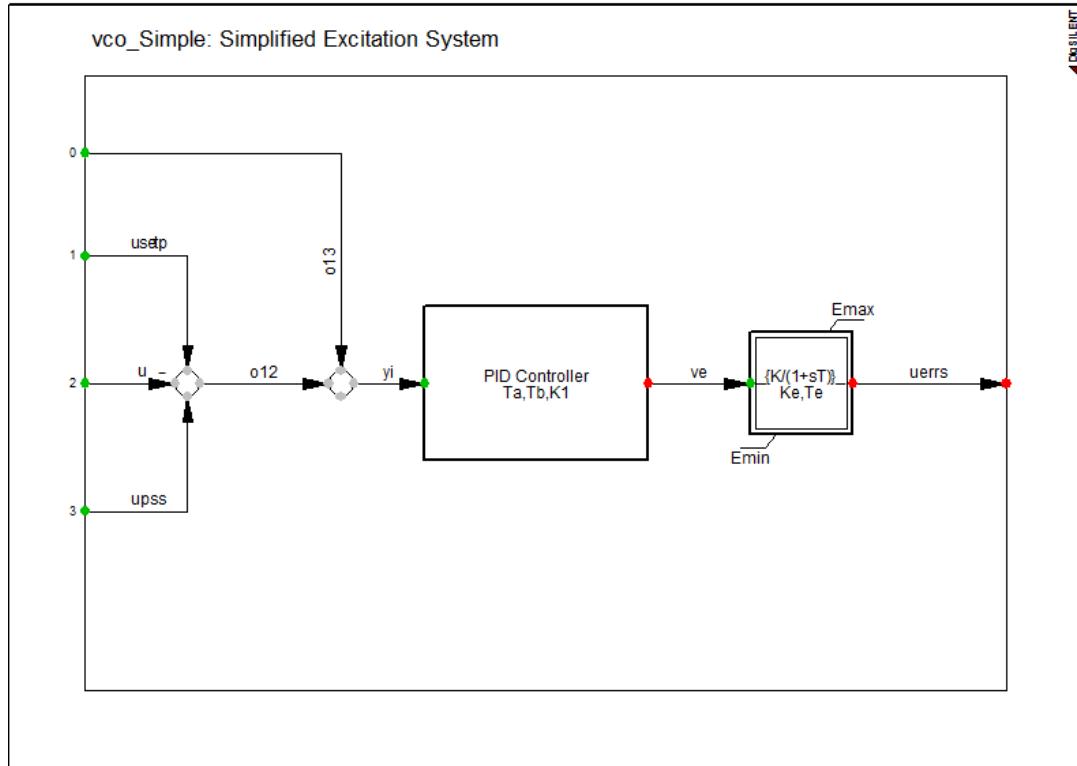
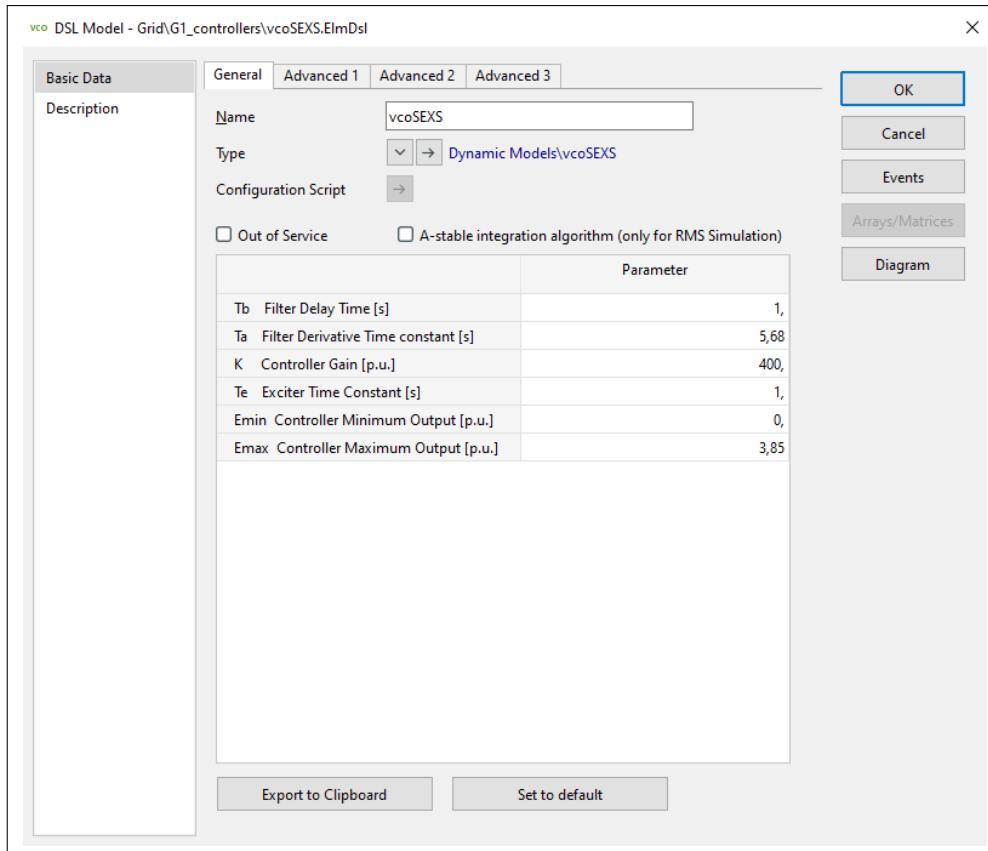


Figure 30.3.4: *DSL Model Type* of the voltage regulator, defined using graphical block diagrams

A *DSL Model* referencing the previously exemplified model thus “instantiates” the controller in the power system. As a result, the *DSL Model* can be assigned to a slot of a relevant *Composite Model* (e.g. see Figure 30.2.3) and thus be connected to a synchronous machine element (to be controlled). Furthermore, whereas parameters of a *DSL Model Type* are literals, the *DSL Model* allows the users to customise the parameter values and apply them to the controller, as shown in Figure 30.3.5.

Figure 30.3.5: *DSL Model* for the synchronous machine voltage regulator

A graphically defined *DSL Model Type* may contain sub-blocks (known as *Block References*) pointing either to a *DSL Macro* (see Figure 30.3.6) or to another graphically defined *DSL Model Type* (see Figure 30.3.7). The structure of the *DSL Model Type* is thus recursive and the user should check that this recursive structure does not contain circular references. A *DSL Macro* contains one or more DSL expressions and forms a basic block for more complex transient models.



Figure 30.3.6: Implementation of the first-order low-pass filter with gain and limits block, using DSL

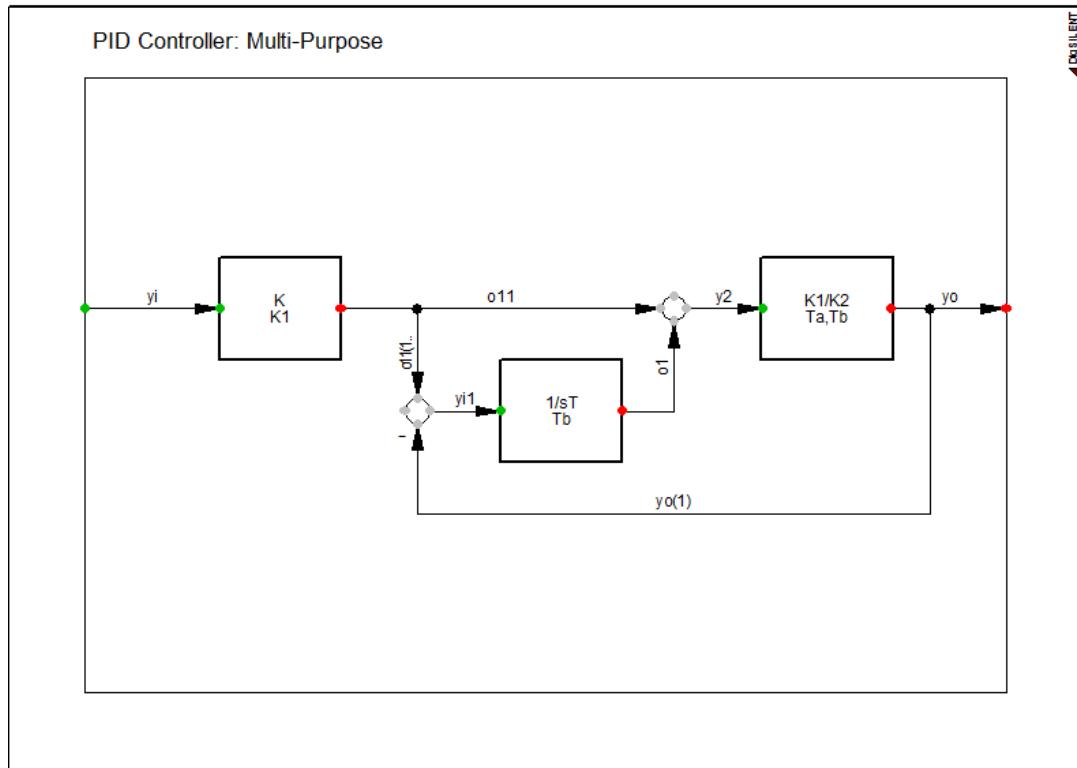


Figure 30.3.7: Implementation of the PID controller block, using a block diagram

It is also possible to implement the *DSL Model Type* by directly using DSL code, as it was the case of the simple *DSL Macro* shown in Figure 30.3.6.

## 30.4 DSL: The *DIGSILENT Simulation Language*

The *DIGSILENT Simulation Language* (DSL) is used to design and simulate dynamic models for representing various dynamic systems, including controllers of electrical equipment as well as other components used in electric power systems.

The structure of a dynamic model using DSL is explained in Section 30.4.6

As for any other simulation or programming language, a special syntax is provided for the model formulation. The DSL syntax is explained in Section 30.4.4.

### 30.4.1 Introduction to DSL

A decisive factor for reliable simulation results is the accuracy and completeness of system model representation. In order to provide a very flexible modelling and simulation tool, which forms part of a dynamic simulation program, a control system based simulation language is developed. The following describes the main features of DSL:

- The simulation tool falls into the category of Continuous System Simulation Languages (CSSL);
- DSL includes a complete mathematical description of (time-) continuous linear and non-linear systems;
- The simulation tool is based upon common control and logic diagrams, leading to a non-procedural language, as the sequence of elements can be chosen arbitrarily. In other words, a DSL model

can be converted into a graphical representation;

- Provision of flexible definition of macros, which could be: algebraic equations, basic control elements like PID, PTn or even complete physical subsystems like valve groups or excitation systems;
- Provision of various intrinsic functions such as: “select”, “lim”, “limits”, “lapprox”, “picdro” in order to provide a complete control of models;
- Provision of various formal procedures for error detection and testing purposes such as: algebraic loop detection, reporting of unused and undefined variables and missing initial conditions

The *DIGSILENT* Simulation Language is used to define *User-defined* dynamic controllers, which receive input signals from the simulated power system and which react by computing various output signals. During the simulation, the model equations of DSL models are combined with those describing the dynamic behaviour of the power system components. These equations are then evaluated together, leading to an integrated dynamic simulation.

The DSL main interfacing functions are:

**Signal input and output channels:** a large set of system variables can be used as input and outputs

**Events:** Conditions evaluated by DSL models may cause events to be sent to the program kernel where they will be scheduled within the event queue.

**Output and Monitoring:** Conditions may trigger user-defined messages to be displayed in the output window.

### 30.4.2 Structure of a dynamic model using DSL

A dynamic model implemented using DSL can be designed using two fundamentally different approaches:

- The dynamic model is built entirely using a single block that contains all model equations, using DSL language. The resulting model is regarded as a monolithic block, whereas *PowerFactory* directly interprets the model equations.
- The dynamic model is built using a series of interconnected sub-blocks, each sub-block being responsible for a given model function. The sub-blocks are interconnected using a graphical block diagram approach. *PowerFactory* is able to parse through the model’s block diagram and pack all model equations belonging to various sub-blocks, equations which are subsequently interpreted together as a monolithic block.

For simple models, the first approach is possible, whereas whenever models tend to be more complex, there is a need to separate basic block functionality. Therefore, a hierarchical approach is made available, by designing sub-blocks and sub-sub-blocks, where so called *DSL macros* form the lowest-level building block of these models. A simplified example is provided in Figure 30.4.1.

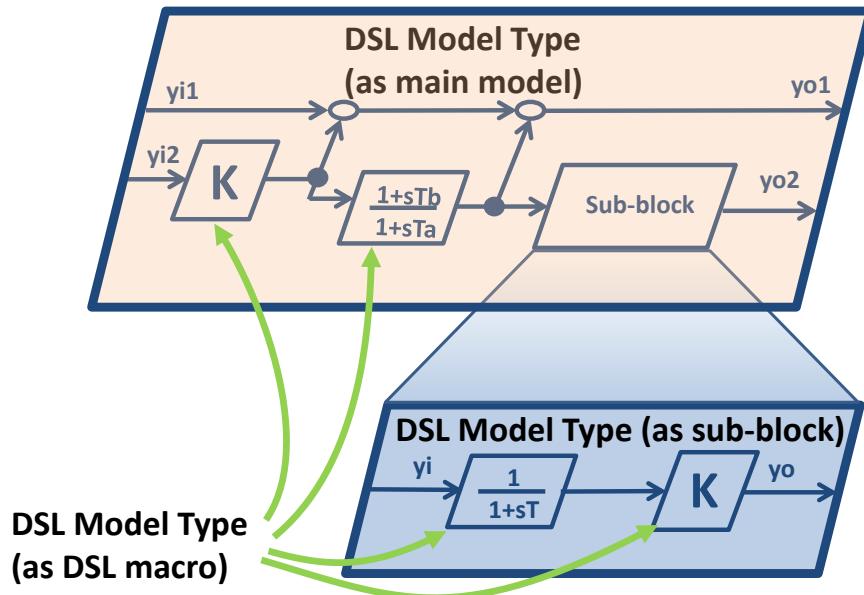


Figure 30.4.1: Simplified example of a hierarchically designed *User-defined* dynamic model

By inspection of Figure 30.4.1, it can be deduced that the *DSL Model Type* object (*BlkDef*) is a highly versatile functional block, that can be used in the following ways:

- Main Model (*BlkDef dsl*, graphically defined) - the *DSL Model Type* is the top-level object of a controller and its block diagram contains all control components
- Model used as a sub-block (*BlkDef dsl*, graphically defined) - the *DSL Model Type* is included either in the diagram of the *Main Model* or in that of a sub-block of the *Main Model*. This model is only representing a part of the *Main Model*'s functionality.
- DSL Macro (*BlkDef +*, defined using DSL code) - the *DSL Model Type* is included either in the diagram of the *Main Model* or in that of a sub-block of the *Main Model*. *DSL Macros* typically represent basic dynamic model functionality e.g. a gain, a first-order low-pass filter, an integrator, etc. They are distinctly marked using the *Macro* flag in the edit dialog of the *DSL Model Type*. Refer to Section 30.4.10 for more information.

- Note:**
- It is the *DSL Model Type* representing the *Main Model* which is used as reference when instantiating a *DSL Model* (*ElmDsl*) in the power system, and not the sub-blocks or the *DSL macro* components
  - The *DSL Model Type* representing the *Main Model* must contain initialisation conditions for all models' state variables, the relevant inputs and outputs. Typically, *DSL Macros*, although containing state variables, do not contain relevant initial conditions. This is done on purpose, in order to give flexibility to the model developer in the model initialisation process.
  - The *DSL Model Type* representing the *Main Model* decides the *DSL Level* applied for the entire model (including all sub-blocks or *DSL Macros*). More information regarding the *DSL Level* is detailed in Section 30.5.2.2.

### 30.4.2.1 Defining DSL Models

A new DSL model is created either by entering the DSL code in the equation part of a *DSL Model Type* (*BlkDef*) object, or by creating a new Graphical Block Diagram. Both methods will result in a *DSL Model Type Object* which holds the definition of the DSL model.

The *DSL Model Type* objects thus serve two purposes in the process of constructing a DSL model:

- They hold the definitions and parts of a graphically constructed *DSL Model Type*, and the diagram graphic which was used to define the model (discussed in detail in Section [30.6](#));
- They provide the framework in which new *DSL Macros* (discussed in Section [30.4.10](#)) or *DSL Model Type* (as model sub-blocks) can be defined.

### 30.4.3 Terms and Abbreviations

The following terms and abbreviations are used to describe the DSL syntax:

#### **expr**

- arithmetic expression, not to be terminated with a ';
- arithmetic operators: +, -, \*, /
- constants: all numbers are treated as real numbers
- standard functions: sin(x), cos(x), tan(x), asin(x), acos(x), atan(x), sinh(x), cosh(x), tanh(x), exp(x), ln(x), log(x) (base 10), sqrt(x) (square root), sqr(x) (power of 2), pow(x,y), abs(x), min(x,y), max(x,y), modulo(x,y), trunc(x), frac(x), round(x), ceil(x), floor(x).  
These standard functions are described in detail in Section [30.16](#) (DSL Reference).
- Parenthesis: (arithmetic expression)

All trigonometric functions are based on radians (RAD).

#### Example:

```
A = x1+2.45*T1/sin(3.14*y)
```

#### **boolexpr**

- logical expression, not to be terminated with a ';
- Logical relations: <, >, < > (inequality), >=, <=, =.
- Unary operators: .not.
- Binary operators: .and. .or. .nand. .nor. .eor.
- Parentheses: logical expression

#### Example:

```
A = x1>0.and..not.x2 <= 0.7.or.T1=0.0
```

#### **string**

anything within '...' (single quotation marks).

#### Example:

```
vardef(Ka)='p.u.';'Controller Gain'
```

### 30.4.4 General DSL Syntax

**Line length:** The maximum line length is 80 characters. Longer lines have to be broken by using the '&' sign in the first column of the continuing line. A '&' sign in the first column joins the current row and its preceding row.

#### Example:

```
x1. = select({at<>0} .and. {bt>=10},
& (1-sqr(x1)/sqr(at))/Tw, 0)
```

Line breaking cannot be used within names or strings.

**Case sensitivity:** All keywords, names, functions, variables, models, macros, etc. are case sensitive.

**Blanks:** All blanks are removed when the DSL code is processed. Exception: blanks in strings are kept.

**Comments:** The “!” sign causes the subsequent text in a line to be interpreted as a comment. Comments are not considered when the DSL code is processed.

**Example:**

```
! comments may start at the beginning of a line
x1. = select(at<>0, ! comments may be used in broken lines
& (1-sqr(x1)/sqr(at))/Tw, 0)
```

### 30.4.5 DSL Variables

The *Model Variables* are declared in the *Basic Options* page of the *DSL Model Type* Edit dialog - see Section 30.5.2 for more information. A DSL model may use five different types of variables:

**Output variables:** An *Output variable* represents a variable that is available outside the scope of this model. It can be connected externally to a different DSL model or a network element.

- *Output variables* are declared in the *DSL Model Type* Edit Dialog, page *Basic Options* (see Section 30.5.2.1). If the *DSL Model Type* is graphically defined (see Section 30.6) then the *Output variables* are automatically identified from the block diagram.
- Output variable names follow the DSL variable naming guidelines, detailed in Section 30.5.2.1.
- All output variables are real, internally represented using *double* variable types.
- If the *Output variable* is defined in a *DSL Model Type* used as a *Main Model* (see Section 30.4.2), then an initialisation statement is required for this *Output*. Whether the initialisation is done within the model itself or externally, by another model, depends on the practical arrangement.

**Input variables:** Input variables may originate from other DSL models or from power system elements. In the latter case, currents and voltages, as well as any other signal available in the analysed power system, become available to the DSL model.

- *Input variables* are declared in the *DSL Model Type* Edit Dialog, page *Basic Options* (see Section 30.5.2.1). If the *DSL Model Type* is graphically defined (see Section 30.6) then the *Input variables* are automatically identified from the block diagram.
- Input variable names follow the DSL variable naming guidelines, detailed in Section 30.5.2.1.
- All input variables are real, internally represented using *double* variable types.
- Parameter events (*EvtParam*) can be applied on an *Input variable*. The input value is changed by this event at the event's execution time provided that the input variable is not connected (i.e. there exists no model that has an equation for this variable).
- If the *Input variable* is defined in a *DSL Model Type* used as a *Main Model* (see Section 30.4.2), then an initialisation statement is required for this *Input*. Whether the initialisation is done within the model itself or externally, by another model, depends on the practical arrangement.

**State variables:** State variables (time-continuous) are time-dependent signals which are implicitly defined using their time derivative equation. The *PowerFactory* program evaluates the *State* derivative equations at each time step and solves them for the next value of the *State*.

- *State variables* are declared in the *DSL Model Type* Edit Dialog, page *Basic Options* (see Section 30.5.2.1). If the *DSL Model Type* is graphically defined (see Section 30.6) then the *State variables* are automatically identified from the block diagram.

- State variable names follow the DSL variable naming guidelines, detailed in Section [30.5.2.1](#).
- All *State* variables are real, internally represented using *double* variable types.
- A *State* variable cannot be used as an *Output* variable; whenever required, the use of an assignment like  $y=x1$  is recommended (i.e. *Output* variable equals *State* variable).
- The only time a *State* variable can be written on the left side of an equation is for when defining its first-order derivative equation e.g. “ $x.=yi$ ”. This means that only the derivative of a *State* can be assigned an expression. It is not allowed to directly assign a value to a *State*.
- Parameter events (*EvtParam*) can be applied on a *State*. The *State* value is changed by this event at the event’s execution time.
- *States* can be changed discontinuously from within the same *DSL Model* by using the DSL special function `reset`.
- An initialisation statement is required for a *State variable*. Whether the initialisation is done within this model or in any of the higher hierarchical levels of the model (if used as a sub-block), depends on the practical arrangement.

**Parameters:** Parameters are constant variables which are used to customise the behaviour of a dynamic model. They enable *Model developers* to use non-hardcoded variables within the control structure while still allowing *Model users* to assign preferential values to these variables. Parameters are declared in the *DSL Model Type* and visible and configurable in the *DSL Model*.

- *Parameters* are declared in the *DSL Model Type* Edit Dialog, page *Basic Options* (see Section [30.5.2.1](#)). If the *DSL Model Type* is graphically defined (see Section [30.6](#)) then the *Parameters* are automatically identified from the block diagram. Additional parameters can be defined using the corresponding field in the *Equations* page.
- Parameter names follow the DSL variable naming guidelines, detailed in Section [30.5.2.1](#).
- All parameters are real, internally represented using *double* variable types.
- A parameter declared with the name “**oarray\_nnn**” and “**omatrix\_nnn**” (with “nnn” being a user defined suffix containing characters defining the characteristic name), is automatically interpreted to be either a one- (“**oarray\_nnn**”) or a two-dimensional (“**omatrix\_nnn**”) characteristic, being instantiated externally of the *DSL Model* using *IntMat* objects (see Section [30.3](#)). Parameter names “**oarray\_nnn**” and “**omatrix\_nnn**” follow the DSL variable naming guidelines, detailed in Section [30.5.2.1](#).
- Parameter events (*EvtParam*) can be applied on a parameter. The parameter value is changed by this event at the event’s execution time. The change is non-persistent (i.e. the parameter value is changed only during the simulation, but not permanently in the *DSL Model*)
- Parameters are not allowed to have initialisation statements.

**Internal variables:** Internal variables are defined and used in the DSL model to ease the construction of a set of DSL equations.

- *Internal variables* are declared in the *DSL Model Type* Edit Dialog, page *Basic Options* (see Section [30.5.2.1](#)). If the *DSL Model Type* is graphically defined (see Section [30.6](#)) then the *Internal variables* are automatically identified from the block diagram. Additional internal variables can be defined using the corresponding field in the *Equations* page.
- Internal variable names follow the DSL variable naming guidelines, detailed in Section [30.5.2.1](#).
- All internal variables are real, internally represented using *double* variable types.
- Parameter events (*EvtParam*) can be applied on an *Internal variable*. The variable value is changed using this event to the prescribed value.
- *Internal variables* can be discontinuously changed during the simulation from within the same *DSL Model* by using the DSL special function `reset`. This method is effective only if the *Internal variables* does not have an equation statement (but only an initialisation statement).

### 30.4.6 DSL Model Structure

DSL models are typically composed of several functional parts:

- The *Model Interface*, which states, among other things, the model name, title, classification and declares the model variables. The *Model Interface* part is configured in the *Basic Options* page of the *DSL Model Type* Edit dialog - see Section 30.5.2 for more information;
- Initialisation statements. The *Initialisation* of a model can be programmed in the *Equations* page of the *DSL Model Type* - see Section 30.4.7 for more information; The direct (non-iterative) initialisation of a model is discussed in Section 30.4.7.1. Furthermore, a *Configuration Script* can be added to the *DSL Model Type* in order to facilitate a more flexible initialisation of the model (refer to Section 30.4.7.3. The iterative model initialisation is discussed in Section 30.4.7.2.
- Equation code. The *Equation* code can be programmed in the *Equations* page of the *DSL Model Type* - see Section 30.4.8 for more information;
- Various model definitions. Various other definitions (e.g. statements on limits or description of parameters) are programmed in the *Equations* page of the *DSL Model Type* - see Section 30.4.9 for more information.

### 30.4.7 Initial Conditions

#### 30.4.7.1 Direct Definition of Initialised Variables (Non-iterative)

##### **inc(varnm) = expr**

Definition of the initial condition of variable varnm. If inc(varnm) is not defined, the normal assignment expression will be evaluated (only possible if varnm is of the intern or input type). If inc(varnm) is defined, it will be evaluated when the model is reset.

##### **inc0(varnm) = expr**

Definition of the initial condition of variable varnm, for unconnected output or input variables. This variant of the inc() statement is used only when the variable varnm could not be initialised through the initial condition of the connected input or output signal. The inc0() statement is thus used to make open input or output terminals possible.

##### **incfix(varnm) = expr**

This variant of the inc() statement is valid only in connection with automatic initialisation and is used to determine the initial values in ambivalent situations. With incfix, one or more variables can be directly initialised so that other variables can be initialised automatically.

Example:

An AVR model has two inputs, [upss , usetp ], and one output, [uerrs ]. Both inputs cannot both be initialised automatically by the single output value, which is determined by the connected machine. Therefore one of the inputs must be initialised as fixed, e.g. by incfix(upss)=0. The initial value of usetp is now automatically determined, using upss=0.

#### 30.4.7.2 Iterative calculation of initial conditions

Iterative solvers are available for determining initial values. The functions *loopinc* and *intervalinc* are used to find the initial value for one set of parameters if the initial values of another set of parameters, which are functions of the first set of parameters, are known. The iterative functions are used to find the (approximated) values for the unknown parameters for which the known parameter take their initial value.

- Note:**
- The iterative calculation of initial conditions and its underlying functions are not-supported by the C Interface compiler (Section 30.14.3) using DSL level 6 or earlier. DSL models containing one of these functions cannot be compiled.
  - In case of algebraic loops in a model using DSL level 7, the system of equations needs to be solved via a non-linear solver. The solver settings can be found in *Basic Options → Advanced* when the “Allow iterative solver at initialisation” flag is set. Additionally, the solver allows the initialisation of the derivative of a state variable.

### **inc(varnm.) = expr**

Within DSL Level 7 and upwards, the derivative of state variables can be assigned. Definition of the initial condition of a state variable varnm, in which the value of the state's derivative is defined by expr. An iterative method of solving this equation is automatically applied at initialisation.

#### **loopinc(varnm, min, max, step, eps)**

Performs a simple linear search for a single value for which the parameter varnm is closest to its known initial value.

varnm = target variable, whose initial value is known

min = lower limit

max = upper limit

step = step size

eps = maximum error

**Example:**

```
inc(a) = loopinc(b, -5, 5, 0.01, 0.001)
```

- The initial value of variable a is searched for by evaluating parameter b, beginning at a=-5, ending at a=5, with an increment of 0.01.
- Return value: the value of a for which the deviation of b from its known initial value, takes the smallest value. A warning is given if the smallest deviation is greater than eps.
- Restriction: can only be used on the right side of an inc() statement

#### **intervalinc(varnm, min, max, iter, eps)**

Performs an “interval-division search” for a single value for which the parameter varnm is closest to its known initial value.

varnm = target variable, whose initial value is known

min = lower limit, max = upper limit

iter = maximum number of iterations

eps = maximum error

**Example:**

```
inc(a) = intervalinc(b, -5, 5, 40, 0.001)
```

**Explanation:**

The initial value of the variable a is searched for, within the interval [-5,5] by successively dividing the interval as long as the deviation of the variable b from its initial value is less than *eps*. The iteration stops if the maximum number of iterations is reached, and a warning is given if the smallest deviation is greater than *eps*.

Restriction:

May only be used on the right side of an inc() statement

**newtoninc (initexpr, start, [iter], [eps])** - DSL level 7 and onwards. Function requires either two (initexpr, start) or four (initexpr, start, iter, eps) input arguments.

**newtoninc (initexpr, start, iter, eps)** - DSL level 6 and earlier. Function requires four input arguments (initexpr, start, iter, eps).

Performs a Newton iterative search for one or more parameters by minimising the errors in a set of coupled equations.

- initexpr = the expression which must equal the parameters whose initial value is sought
- start = the starting search value for the parameter whose initial value is sought
- iter
  - DSL level 6 and earlier: the maximum allowed number of iterations
  - DSL level 7 onwards: optional argument (supported for simplified migration to DSL level 7 from an earlier level). If provided, then the argument is ignored. Instead, the maximum allowed number of iterations is taken from the *DSL Model type (BlkDef)*, parameter *maxiteration* (available from the DSL Model type edit dialog, *Basic Options* → *Advanced* → *Iterative solver options* pane).
- eps
  - DSL level 6 and earlier: the maximum allowed absolute error between initexpr and the parameter whose initial value is sought.
  - DSL level 7 onwards: optional argument (supported for simplified migration to DSL level 7 from an earlier level). If provided, then the argument is ignored. Instead, the maximum allowed absolute error is taken from the *DSL Model type (BlkDef)*, parameter *maxerror* (available from the DSL Model type edit dialog, *Basic Options* → *Advanced* → *Iterative solver options* pane).

Example (DSL level 6):

```
qt0 = 0.5
eps = 0.000001
maxiter = 100
inc(hedr) = newtoninc(hw-sqrt(qedr)*(Rds+Rdr), hw,
maxiter, eps)
inc(qt1) = newtoninc(Pt1/(4*dh*eta1), qt0, maxiter, eps)
inc(qt2) = newtoninc(Pt2/(4*dh*eta2), qt0, maxiter, eps)
inc(qt3) = newtoninc(Pt3/(4*dh*eta3), qt0, maxiter, eps)
inc(qt4) = newtoninc(Pt4/(4*dh*eta4), qt0, maxiter, eps)
```

Example (DSL level 7):

```
qt0 = 0.5
inc(hedr) = newtoninc(hw-sqrt(qedr)*(Rds+Rdr), hw)
inc(qt1) = newtoninc(Pt1/(4*dh*eta1), qt0)
inc(qt2) = newtoninc(Pt2/(4*dh*eta2), qt0)
inc(qt3) = newtoninc(Pt3/(4*dh*eta3), qt0)
inc(qt4) = newtoninc(Pt4/(4*dh*eta4), qt0)
```

The above examples show a part of the initial value definitions for a model where the initial values of 5 parameters (*hedr*, *qt1*, ..., *qt4*) are sought simultaneously by setting up a system of coupled equations and solving that system by the Newton method so that, eventually:

$$hedr \approx hw - \sqrt{qedr} \times (Rds + Rdr) \quad (30.1)$$

$$qt1 \approx Pt1/(4 \times dh \times eta1) \quad (30.2)$$

$$qt2 \approx Pt2/(4 \times dh \times eta2) \quad (30.3)$$

$$qt3 \approx Pt3/(4 \times dh \times eta3) \quad (30.4)$$

$$qt4 \approx Pt4/(4 \times dh \times eta4) \quad (30.5)$$

The following guidelines should be considered:

- Add the initial conditions to the complex block, as opposed to each primitive (such as a first-order time lag).
- The general initialisation “direction” is from right to left, i.e. the outputs are normally known and the inputs (or setpoints) have to be determined.
- If initial conditions are not defined for a certain variable, the simulation equations are used instead. It should be therefore enough to specify the initial conditions of the state variables and input variables.
- The option Automatic Calculation of Initial Conditions requires configuring, but does not require correct initial conditions for each state/input variable. The initial values are only used to initialise the iteration process. The incfix-function can be used to determine the initial values in ambiguous situations.
- Use the option Verify Initial Conditions to check if the initial conditions lead to the correct result.

#### **Initialisation logic for multiple DSL blocks containing inter-dependent variables**

Complex initialisation procedures involving multiple DSL blocks are supported.

##### **30.4.7.3 Configuration scripts for initialisation**

A DPL based *Configuration Script* can be optionally added to any DSL model or an IEC 61400-27 based external interface model defined within the *DSL Model Type* object. The script is executed automatically by the command *Calculation of Initial Conditions (ComInc)* after the successful execution of the load flow calculation and prior to the actual model initialisation. Therefore a *Configuration Script* can only access and use load flow results, element and type parameters, but not signal values.

The functionality of the *Configuration Script* is two-fold:

- it provides a flexible way of configuring DSL model parameters;
- it allows DSL models to issue customised output window messages i.e. via DPL output window methods e.g. *Info()*.

The Configuration Script is linked within a DSL model as shown in Figure 30.4.2. It can be set up by following these steps:

- Creating a DPL script within the *DSL Model Type (BlkDef)* object. Within the script, users may define various *Result* variables. Those results whose names match DSL model parameters will be considered for overwriting the values assigned to DSL parameters. That is, during the calculation of initial conditions, the pre-existent values of the DSL model parameters will be overwritten by the *Configuration Script*.
- Optional: the DPL script can be programmed to contain both *Input parameters* as well as External objects.
- Referencing the DPL script to the *DSL Model Type (BlkDef)* object, using the corresponding field “Configuration Script” within *Basic Options* → *Advanced* tab of the Edit dialog of the Block definition object.
- Creating a new *DSL Model (ElmDsl)* within the *Network Model* folder of the project (e.g. typically within one of the “Grid” folders). This action will automatically create a corresponding DPL script that references the previously defined *Configuration Script*.
- Optional: It is possible to assign instance-dependent attributes to the *Configuration Script*. Each *DSL Model* contains a DPL script that references the remote script of the *DSL Model Type*. Those *Input parameters* and *External objects* which were previously defined within the remote script (refer to the optional step above), will be available for editing for each DSL model that shares the same *DSL Model Type*. This provides increased flexibility, allowing each model to be differently parameterised. For example, it is possible to link different external objects to each *Configuration Script*.

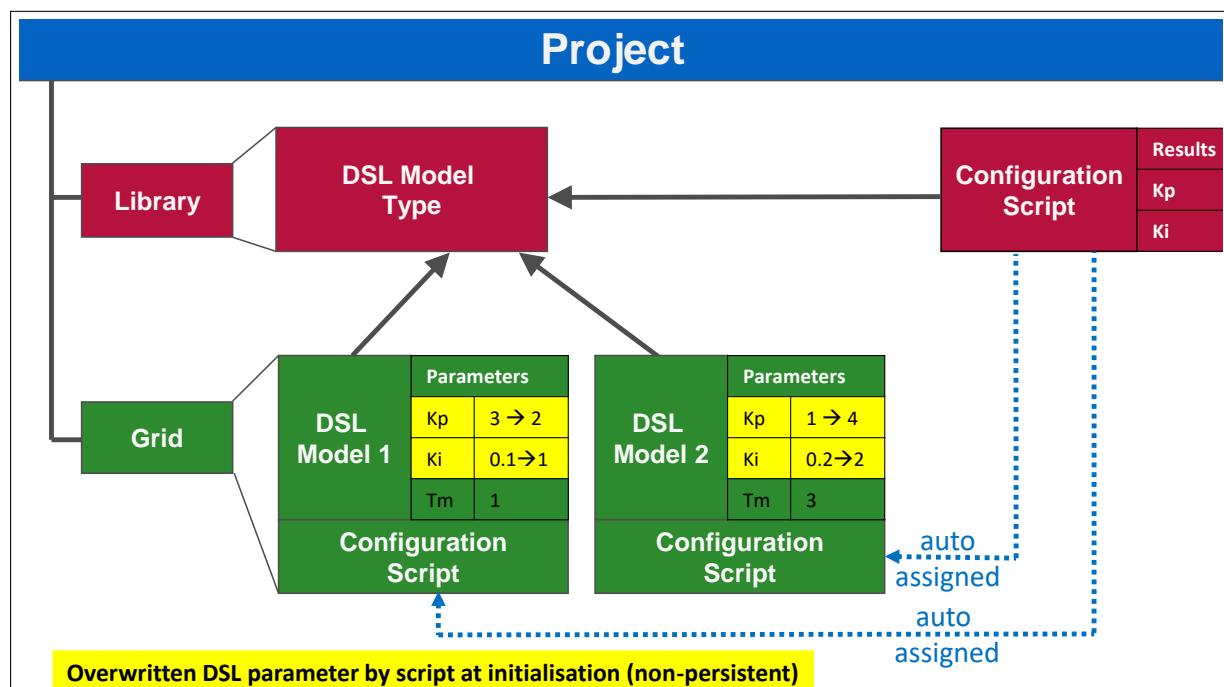


Figure 30.4.2: Configuration Script functionality and inter-operation

**Notes:**

- The scripting language used for the Configuration Script is *DlgSILENT Programming Language* (DPL). Refer to Chapter 23 for a comprehensive description of DPL. Further documentation is available within the *DPL Reference* which provides a full description of available functions. The *DPL Reference* can be accessed in the *PowerFactory* program from the main menu at *Help* → *Scripting References*→ *DPL*. Functions that can be used within the *Configuration Script* are marked appropriately in the *DPL Reference* documentation with a star (\*) symbol (e.g. *GetFrom-StudyCase\**). There are functions (without a star (\*) symbol) that are not available, e.g. the execution of a load flow (*ComLdf*) command is not allowed.

- Parametric changes made by the *Configuration Script* are non-persistent. When the simulation is reset (e.g. by clicking the button *Reset Calculation*), the values of the modified DSL parameters will be set back to their initial values - meaning that the *Configuration Script* does not permanently modify project data.
- The values of modified DSL parameters can be displayed using the Data Manager. After the successful execution of the *Calculation of Initial Conditions*, do the following:
  - Using the *Data Manager* or the *Network Model Manager*, navigate to and identify the *DSL Model* that contains the modified parameter;
  - Left-click the DSL model and activate the *Detail Mode* from the toolbar;
  - Left-click on the *Flexible Data* tab and then on the *Variable Selection* button from the toolbar. The *Variable Selection* window will be shown;
  - Use the filter *Group* to get the *Results* and the filter *Calculation* to get *Simulation RMS* or *Simulation EMT* (depending on the case).
  - Select the modified parameter(s) in order to display them;
  - Click *OK* and observe the parameter values being used (overwritten by the *Configuration Script*).
- In Figure 30.4.2 above, an example is provided for the operation of a script. In this case, the *Configuration Script* declares two result variables *Kp* and *Ki*. The Block definition contains three parameters: *Kp*, *Ki* and *Tm*. This means that the script will overwrite only two model parameters: *Kp* and *Ki*. In the figure, exemplary values are provided: for “*DSL Model 1*” parameter *Kp* has an initial value of 3 and the script will overwrite it with the value 2. Similarly, parameter *Ki* has an initial value of 0.1 and the script will overwrite it with the value 1. The overwritten values of the parameters will be used for all subsequent calculations (e.g. initial conditions and simulation). Further value modifications of these parameters will still be possible during the simulation as for any other DSL parameter e.g. via Parameter events (*EvtParam*). The script allows assignment of instance-dependent attributes. It implies that, depending on the script’s code, each *DSL Model* may be configured with individual values for the two parameters. Consequently, the parameters *Kp* and *Ki* of “*DSL Model 2*” will have been assigned different values. Note that at the end of the simulation all modified parameters are automatically set back to their initial values.

### 30.4.8 Equation Code

Within the equation code, all equations necessary to build up the simulation models are included. The set of equations defines a set of coupled differential equations which describe the dependency between the input and output signals. The equations may describe simple linear, single-input single-output functions, or highly complex non-linear, non-continuous, multi-input, multi-output functions. DSL Standard Functions (described in Section 30.16.1) and DSL Special Functions (described in Section 30.16.2) can be used within the model equations.

DSL is used to describe the direct relationship between signals and other variables. Expressions may be assigned to a variable, or to the first derivative of a state variable. Higher order differential equations have to be split up into a set of single order differential equations by the introduction of intermediate state variables.

Therefore, as far the calculation of time-continuous state variables is concerned, a DSL model can be conceptually described using state-space notation, as follows:

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{A}'\mathbf{x} + \mathbf{B}'\mathbf{u} \\ \mathbf{y} &= \mathbf{C}'\mathbf{x} + \mathbf{D}'\mathbf{u}\end{aligned}\tag{30.6}$$

where:

$\mathbf{A}'$  – System state matrix.

$\mathbf{B}'$  – Input matrix

$\mathbf{C}'$  – Output matrix

D' – Feedforward matrix

x – Time continuous state variables vector

u – Input variables vector

y – Output variables vector

The elements of matrices  $A'$ ,  $B'$ ,  $C'$  and  $D'$ , can be linear/non-linear, time dependent/independent, continuous/discontinuous.

### 30.4.8.1 Equation Statement

The equation statements are used to assign expressions to variables, thus relating all variables in a set of differential equations.

Syntax:

**varnm = expr**

Assigns expression “expr” to variable “varnm”.

Examples:

```
y = sin(a)+3*x1  
y = .not. x1>2 .or. a<=3
```

**varnm. = expr**

Assigns expression expr to the first order derivative of the variable varnm.

Examples:

```
x1. = (xe-x1)/T1  
x2. = x1
```

### Remarks

- DSL assignments may occur in any sequence. The sequence does not influence the evaluation of the assignments.
- All DSL variables are *Real*, represented using double precision floating point numbers. This is the case even if assigned to a boolean expression, in which case the value for logical false is 0.0 and for logical true is 1.0.
- When a variable z is used in a logical expression (i.e.  $y = .not.z$ ), the logical 1 of z is tested by evaluating ( $z > 0.5$ ):  
 $y = .not.z$  is interpreted and equal to  $y = (z < 0.5)$   
There is no warning against mixing logical and non-discrete variables in expressions. Consequently the following code will not cause a message to be emitted: depending on y, z will take the value  $x1 + 4.0$ , or just  $x1$ :  

```
y = .not. x1>2 .or. a<=3  
z = 4.0*y + x1
```
- The assignment of a value to a variable takes place in an order which recognises the connections between these variables. In the case of the following example, the second line will be evaluated first, then line 1:
  1.  $a = b+5$
  2.  $b = x1$
  3.  $x1. = 1$
- Algebraic loops are not supported. In the following example, an error message will be displayed:  

```
a = b+5  
b = 2*a
```
- If there is no assignment to a variable varnm, varnm will keep its initial value. The right side expression may not contain derivatives. Derivatives may only appear on the left side of the equal

sign. The first example is correct; the second is incorrect.

```
x1. = asin(a) ! Correct
a = sin(x1.) ! Not accepted
```

- Division by zero: the simulation solver automatically treats a division by zero as a division by a small number (value used is 0.000001). On the other hand, within a compiled DSL model (see Section 30.14.3), the division by zero is executed according to C-language specifications. As such, *NaN* (not a number) values may be generated for such cases. It is recommended to treat within the DSL model any possible divisions by zero directly in the model equations e.g. replace ".../Ve" with ".../max(0.000001, Ve)".

## 30.4.9 Various model definitions

The *Equation* section of a DSL model is used to define parameter properties and initial conditions.

### 30.4.9.1 Unit and Parameter Description

#### **vardef(varnm) = unitstring;namestring**

Unit and name for variable varnm.

Examples:

```
vardef(Ton) = 's';'Pick up time for restart' !defines unit and name
vardef(Ton) = ;'Pick up time for restart' !only defines name
vardef(Ton) = 's'; ! only defines unit
```

#### **[varnm] = unitstring**

Unit for variable varnm, maximum 10 characters.

*Remark:* A macro call causes error messages if the units of the substituted variables do not match the defined units.

Example:

```
[Ton] = 's' ! defines unit
```

### 30.4.9.2 Valid Value Ranges

#### **limits(varnm) = [/| minimum value, maximum value |/)**

Defines the valid interval for variable varnm. Violations of the interval limits during simulation will be reported:

```
limits(yt)=[,1] is equivalent to output(yt>1,
'Maximum exceeded: yt=yt>1')
```

The "(" and ")" brackets exclude the minimum or maximum value from the interval; the "[" and "]" brackets include them.

Examples:

```
limits(x)=[min,max] ! min <= x <= max
limits(x)=(min,max] ! min < x <= max
limits(x)=[,max] ! x <= max
limits(x)=(min,) ! min < x
```

If required and if possible, the program automatically determines the smallest interval under several intervals of the same variable.

Example:

`limits(x)=(1,3) and limits(x)=(2,4]` results in  $2 < x < 3$ .

Macro models often define limits for certain variables. The model which uses the macro might also define limits for the variables which are used in the macro calls. The “smallest interval” method thus gives the calling model the freedom to redefine parameter limits without violating the internal macro limit definitions.

The `limfix(varnm)` function is a variant of the `limits(varnm)` function which is evaluated only at initialisation. Its use is encouraged for performance reasons whenever `varnm` is constant throughout the simulation.

### 30.4.10 DSL Macros

A *DSL Macro* is a *DSL Model Type*, which is meant to be included as a building block in a higher level *DSL Model Type* (`BlkDef`, shown as ). *DSL Macros* are typically predefined objects, available in a model library e.g. the *DlgSILENT Library*. The Edit dialog of the *DSL Model Type* provides a checkbox for identifying the object as a “Macro” (see Section 30.5.2 for more information).

A *DSL Macro* is included in a higher level DSL model by creating a *Block Reference* in the block diagram of the higher level model.

To define a *DSL Macro*, do the following:

- Create the *DSL Model Type* (refer to Section 30.5.1).
- Make sure to leave the newly created diagram empty. It is recommended to close it.
- Open the Edit dialog of the newly created *DSL Model Type* and from *Basic Options* → *General* tab set the flag “Macro”. The *DSL Model Type* is now interpreted as a *DSL Macro*.
- Customise the macro by declaring variables and writing the relevant equations. The procedure is similar with the creation of a non-graphically defined *DSL Model Type* (refer to Section 30.7).
- Interact with the *DSL Macro* only via the Data Manager and do not edit or add components in the block diagram of the *DSL Macro*.

---

**Note:** Although less usual, it is still allowed to define a *DSL macro* using a graphical definition. A graphically defined *DSL macro* is atypical because such representations are not very often re-usable. Moreover, such constructs are regarded as sub-blocks of a larger model (refer to Section 30.4.2).

---

A *DSL macro*, for example, might implement a low-pass filter, which may then be used to graphically construct complex controllers which include this filter as a sub-component. A *DSL Macro* can be referred to in more than one complex *DSL Model Types* at the same time. The benefits of using a *DSL Macro* as a basic block of complex models are obvious:

- The developer of the complex model needs only to focus on the high level implementation, while making use of the readily available *DSL Macros* from the *DlgSILENT Library*
- Creation of customised *DSL macros* is possible, with model developers being able to implement and test the low-level functionality of the *DSL macro* using a separate test setup
- Code re-use: it is generally recommended that the code of a given component which is re-used many times is unique.

### 30.4.10.1 DSL Macro Handling

A preparser substitutes each macro call with the equation code of the macro. The variables of the macro DSL model are then replaced by the variables used in the macro call. The local variable names of macros thus disappear after the preparation process.

### 30.4.10.2 DSL Macro Example: Time-Continuous Integrator

Continuous equation:

$$y_{out} = y_0 + \int y_{in} dt \quad (30.7)$$

Rewriting the continuous equation using the state variable x:

$$\frac{d}{dt} y_{out} = y_{in} \quad (30.8)$$

$$\begin{cases} \frac{d}{dt} x = y_{in} \\ y_{out} = x \\ inc(y_{out}) = y_0 \end{cases} \quad (30.9)$$

Finally the implementation in *PowerFactory* using DSL:

```
inc0(yin) = 0 !starts with 0 if no input signal is connected
inc0(yout) = y0 !starts with y0 if no output signal is connected
inc(x) = yout !starts with steady state
x. = yin
yout = x
```

### 30.4.11 Events and Messages

The DSL language provides procedures for the generation of an interrupt event and for sending messages to the output window:

- **event(*enable*, *trigger*, *ConfigurationString*)** generates a user defined event. Further information is found in Section 30.16.2.
- **reset(*var*, *rst*, *val*)** generates DSL model internal events (user defined). Further information is found in Section 30.16.2.
- **output(*boolexp*, *message\_string*)** outputs a message during the simulation. Further information is found in Section 30.16.2.

---

**Note:** The events are accessed or created by opening the edit dialog of the *DSL Model* (double-click on the DSL model *dsl* in the Data Manager), and then pressing the button **Events** in the dialog. A list of events already defined inside this model is displayed. The events generated via procedure *event()* are added to the project's global event list. The events generated via procedure *reset()* are not added to the project's global event list (as they are considered internal DSL events).

---

### 30.4.12 Advanced DSL Features

The numerical integration of DSL models, interrupt scheduling and input-output signal processing is handled automatically by the program kernel. In addition, if the output of a DSL model is an electric current being added to the appropriate total bus current - which is the case if a load or generator model is created - all Jacobian elements necessary for the iterative simulation procedure will be calculated automatically.

Another useful feature of DSL is the algorithm implemented for numerical setup of the system matrix for eigenvalue calculation purposes. Consequently, any model implemented at the DSL level will be automatically taken into consideration when calculating the system eigenvalues or when applying the modal network reduction approach (MRT). Of course, any signal limiting functions will be disabled automatically for this calculation procedure.

### 30.4.13 Graphically defined DSL Model Types

To define graphically a DSL Model type, follow the instructions provided in Section 30.6.1. To access the dialog of the *DSL Model Type* (*BlkDef* *dsl*), double-click on the frame box surrounding the diagram. *DSL Model Type* objects are conceptually similar to “Grid Folders” in the *PowerFactory* database tree. They are defined by graphically defining a controller block diagram of which they will store the graphical information and all logic parts. These parts include signals, small standard components (adders, multipliers, etc.) or DSL Macros.

Although a *DSL Model Type* object is created graphically, it allows for additional DSL equations to define those aspects of the controller that would be otherwise difficult to enter in a graphical way.

## 30.5 DSL: Overview of the *DSL Model Type*

### 30.5.1 Creating a new *DSL Model Type*

To create a new *DSL Model Type* (*BlkDef* *dsl*):

- from the main menu:
  - Click *Insert* → *Dynamic Model* → *DSL Model Type*
- from the Data Manager using the **New Object** button:
  - Focus the Data Manager on the library subfolder *Library* → *Dynamic Models*
  - Click *New Object*
  - Select *DSL Model Type*
- from the *Dynamic Models* folder using the context menu:
  - right-click on the *Dynamic Models* folder on the left side of the Data Manager and select *New* → *Dynamic Model* → *DSL Model Type*
- in each case, an empty diagram is created for drawing the model details

### 30.5.2 The Edit Dialog of the *DSL Model Type*

The Edit Dialog of the *DSL Model Type* (*BlkDef* *dsl*) contains a large set of relevant information about:

- Basic Options - General: selection of model type, model input and output variables, parameters, state variables and limiting signals,

- Basic Options - Advanced: various advanced configuration options e.g. DSL level, use of DSL configuration scripts, compilation options,
- Equations: initial conditions of variables as well as names and units of parameters,
- Description: text based model description which can be customised,
- Version: further customisation regarding versioning.

### 30.5.2.1 Basic Options page - General tab

*Name*: the object's name

*Title*: a user defined title can be provided using this field

*Model type pane*:

- *DSL model*: This is the standard DSL model type. In this case the *Macro* option is used to identify the *DSL Model Type* as a macro inside the library.
- *Compiled model*: Enabling this option will require the selection of the corresponding DLL file (of the compiled model) in an input dialog. For more information see Section 30.14.3 (C Interface).

---

**Note:** *PowerFactory* is supplied based on a 64-bit architecture; hence 64-bit compiled DLL files of all interfaced external models are typically required. *PowerFactory* is capable of loading both 64-bit and 32-bit DLL files of the DSL-C Interface or of the IEC 61400-27 Interface but, for simulation performance reasons, it is recommended to use 64-bit versions whenever possible. In particular for 32 bit DLLs using DSL-C Interface, only the DSL-C Interface version 220001 can be loaded. The DSL-C Interface version is displayed below the *Compiled model* field upon loading the DLL file.

The file assigned in the file path of the *Compiled model* can be automatically included in the project export/import process by means of the *PowerFactory* specific \*.pfpx file format, as documented in Section 9.1.5. As such, the external file can be packed within the \*.pfpx project file along with all necessary data and shared as a single package.

- 
- *Macro flag*: Check this flag if the block definition is meant to be a DSL macro (refer to Section 30.4.10 for more information on DSL macros).
  - *Display name*: If the *Macro* flag is checked then this text field can be freely defined by the user. The *Display name* is shown in the graphic representation of models which make use of this macro (i.e. in block diagrams). If the *Display name* is empty then the object name of the *DSL Model Type* is shown in the block diagrams.

*Variables pane*:

- *Input and output signals* have to be defined for internal use inside the *DSL Model Type*. The number and their name will then appear in the graphical diagram when the block is used.
  - *State variables* are needed when not only linear, but also differential equations are used. Then, for every first-order derivative one state variable must be specified.
  - *Parameters* will appear in the *DSL Model* dialog and can then be specified. The parameter defined in the *DSL Model Type* will automatically be inserted in the *Block Reference*. The names of the parameters can be different in the *Block Reference* and in the *DSL Model Type*. Only the order must be identical.
  - *Internal variables* are only used inside the *DSL Model Type* but can not be set from outside.
- Limiting parameters/input signals pane*:

- *Upper/Lower limiting parameters*: Constant parameters may be defined here for representing a limitation parameter; a corresponding graphical representation of the limiting parameter is applied when using a graphical block diagram which includes this model.
- *Upper/Lower limiting input signals*: Input signals may be defined here for representing a limitation signal; a corresponding graphical representation of the limiting input signal is applied when using a graphical block diagram which includes this model.

---

**Note: Guidelines for constructing variable names:**

- characters allowed :
  - \* small letters ( a – z )
  - \* CAPITAL letters ( A – Z )
  - \* digits ( 0 – 9 )
  - \* underscore( \_ )
- special symbols **are not allowed** e.g. blank spaces and commas. Exception: underscore( \_ ) is allowed.
- first character should be an alphabet or underscore (digits are not allowed). If the first character is an underscore, then the variable name must contain additional characters, and at least one letter or digit.
- variable name should not be reserved word (e.g. name of DSL standard function or DSL special function, refer to Section [30.16](#))

---

*DLL info* pane (only for compiled model types):

- When a *Compiled model* is selected, this pane is shown and it will provide overview information contained within the DLL. The information depends on the DLL file and interface type (e.g. C-Interface or IEC-Interface)
- if a C-Interface based DLL file is used, then a *Checksum* field is also provided. This field, when properly configured, represents the DSL checksum information of the original DSL block definition used for generating the C-Interface source files.

*DSL info* pane:

- *Checksum*: The checksum of this model is shown. For compiled models their pre-programmed checksum is shown. Whenever changes are applied to the model, *PowerFactory* will compare the checksums of the compiled and the DSL model. If the two checksums are different, a corresponding information message will be shown.

### 30.5.2.2 Basic Options page - Advanced tab

*Initialisation options* pane:

- The *Automatic calculation of initial conditions* option enables *PowerFactory* to process any existing “incfix()” statements within DSL models. More information about the “incfix()” statement can be found [here](#).
- *Partial initialisation in case of deadlock*:
- *Configuration Script*: A configuration script (DPL based) can be referenced here. The script should reside within the *DSL Model Type(BlkDef)* object and will be run during the execution of the *Calculation of Initial Conditions* command (*ComInc*). More information on its use is provided in Section [30.4.7.3](#).

*Classification* pane (only for DSL model types):

- *Level* of the model enforces a certain prescribed behaviour of the DSL model equations. For backwards compatibility reasons, DSL models may have a different level. The highest level that can be selected represents the current (recommended) DSL level. For newly created models the highest level should always be used. The existing DSL levels are described in Table [30.5.1](#).

For DSL macros, this field will decide the behaviour of the macro syntax checks (e.g. “reset” keyword is reserved in DSL level 6 and upwards for the *reset* function, whereas in lower levels, the keyword will be allowed for variable declarations). For *DSL Models*, the DSL level of the referenced *DSL Model Type* decides the behaviour both at runtime as well as for the syntax checks. Any DSL level settings contained in various DSL macros or lower-level sub-model components of the highest level *DSL Model Type* are ignored.

- *Linear*: This user-defined flag is used to state whether the model is linear or non-linear. The graphical representation of the block is changed accordingly in block diagrams i.e. a linear block is illustrated using a single rectangle while a non-linear block is depicted using two concentric rectangles. This field has no influence on the behaviour of the dynamic model during dynamic simulation. Nevertheless, it can provide graphical information on the linear behaviour of the constructed model, this field being especially useful for DSL macros.

<b>DSL Level</b>	<b>Description</b>
0	Model created in <i>PowerFactory</i> Build 57 or previous
1	List of all internal variables in header
2	DSL Level 1 plus user-defined sorting of parameters
3	DSL Level 2 plus <a href="#">lim</a> function being precise in time
4	DSL Level 3 plus <a href="#">event</a> function using boolean expression as condition: <code>event(boolean expression, ...)</code> , see Section 30.16.2. DSL Level 4 was introduced with <i>PowerFactory</i> version 13.2.
5	DSL Level 4 plus additional DSL syntax checks and improved warnings. DSL Level 5 was introduced with <i>PowerFactory</i> versions 15.2.9, 2016 SP4 and 2017.
6	<p>DSL Level 5 plus newly developed additional DSL functions</p> <ul style="list-style-type: none"> <li>• <a href="#">picontrol</a> with variable non-windup limits,</li> <li>• <a href="#">reset</a>,</li> </ul> <p>and improvements in behaviour of DSL functions</p> <ul style="list-style-type: none"> <li>• <a href="#">loopinc</a>, <a href="#">intervalinc</a> and <a href="#">newtoninc</a>,</li> <li>• <a href="#">limstate</a> and <a href="#">limstate_const</a>,</li> <li>• <a href="#">picdro</a> and <a href="#">picdro_const</a>,</li> <li>• <a href="#">picontrol_const</a>,</li> <li>• and all DSL functions having min/max limits (internal check of max not being smaller than min).</li> </ul> <p>The improvements are linked with the DSL level to ensure that the dynamic behaviour of existing models using previous DSL levels remains unchanged. DSL Level 6 was introduced with <i>PowerFactory</i> version 2020.</p>
7	<p>DSL Level 6 plus the following improvements</p> <ul style="list-style-type: none"> <li>• Improved DSL parser with significant performance improvement at initialisation, especially for large DSL models,</li> <li>• DSL allows the initialisation of the derivative of state variables e.g. “<code>inc(x.)=0</code>” ,</li> <li>• Internal model variables must always be declared, otherwise an error is generated,</li> <li>• <code>inc()</code> statements cannot include the following DSL special functions: <ul style="list-style-type: none"> <li>– <a href="#">select</a> and <a href="#">select_const</a> (instead, use <a href="#">selfix</a> or <a href="#">selfix_const</a>)</li> <li>– <a href="#">lim</a> and <a href="#">lim_const</a> (instead, use <a href="#">selfix</a> or <a href="#">selfix_const</a>)</li> <li>– <a href="#">limstate</a> and <a href="#">limstate_const</a> (instead, use these functions outside <code>inc()</code> statements, the state variables will be limited at initialisation as well)</li> <li>– <a href="#">picontrol</a> and <a href="#">picontrol_const</a></li> <li>– <a href="#">picdro</a> and <a href="#">picdro_const</a></li> <li>– <a href="#">gradlim_const</a>, <a href="#">movingavg</a>, <a href="#">lastvalue</a> and <a href="#">delay</a></li> </ul> </li> <li>• <a href="#">newtoninc</a> - models containing this function can be compiled,</li> <li>• <a href="#">selfix</a>, <a href="#">aflipflop</a>, <a href="#">limstate</a>, <a href="#">loopinc</a> and <a href="#">intervalinc</a> - improved implementation</li> </ul> <p>The improvements are linked with the DSL level to ensure that the dynamic behaviour of existing models using previous DSL levels remains unchanged. DSL Level 7 was introduced with <i>PowerFactory</i> version 2022.</p>

Table 30.5.1: Description of DSL levels

*Interpolation of internal variables:* Set this option in order to update DSL internal variables during the occurrence of interpolation events.

---

**Note:** By default, *PowerFactory* does not update the internal variables of a DSL model during the occurrence of interpolation events. It is only inputs, outputs and state variables that are updated by default. This behaviour is highly useful for improving the performance of the simulation, especially in event-rich simulation cases. Nevertheless, if stopping a simulation just after the occurrence of an interpolation event, the recorded values of internal variables may be inaccurate. By activating this option, the *PowerFactory* solver will ensure that internal variables are recalculated always, thus delivering accurate results of DSL internal variables - at the cost of simulation performance. This option is particularly useful during the DSL model development stage, in which the internal variables are observed when debugging model behaviour.

---

*Compilation options* pane:

- *Author, Company, Copyright* and *Version* fields are used to store corresponding information in the compiled model file.
- *Check for Compilation:* this button is functional only for DSL non-macro models of the highest level. It checks that the syntax of the model complies with the requirements for compilation.
- *Compile...:* this button is functional only for DSL non-macro models of the highest level which are not encrypted. This button starts the automatic DSL-to-C Interface Converter. A syntax check is first performed and then, if successful, the corresponding C and Microsoft Visual Studio files are written. These files must be compiled with an external compiler, then the *Compiled model* option must be selected in the *Model type* frame and the resulting DLL file must be specified in the dialog. For simplicity, these models will be referred to hereafter as 'compiled models'. For more information see Section 30.14.3 (DSL C Interface).

**Buttons** - There are several buttons on the right side of the dialog, as follows.

**Check:**

"Check" will verify the model equations and output error messages if errors have occurred. Otherwise the following message will be seen:

```
Check '\TestUser.IntUser\Windparks.IntPrj\Library
\\emph{DSL Model Type}s\DFIG\Voltage Control.BlkDef':
Block is ok.
```

**Equations:**

The "Equations" button will print the DSL equations to the output window, regardless of whether they are defined graphically or on the *Additional Equations* page, as well as variable definitions.

**Macro equat.:**

This button prints all *DSL Model Type* equations (including the equations in the used DSL macros) to the output window.

**Diagram:**

This button is used to show the *DSL Model Type* diagram (if existing).

**Pack:**

Pack will copy all hierarchically lower referenced DSL sub-models (or DSL macros) used by a *DSL Model Type* to the folder "Used Macros" inside the object itself. In this way there will no

longer be any external references. Beware: Once the *Pack* function is used, any further changes to the previously referenced objects will have no influence on the current model anymore - the hierarchically lower referenced DSL sub-models (or DSL macros) are no longer linked to the external objects. Therefore, if an error occurs in a specific macro it must be fixed separately in each packed component.

#### **Pack-> Macro:**

This command will reduce the entire model (including DSL blocks and additional equations and macros) to a single DSL model containing only equations. All graphical information will be lost. It should be noted that this command is irreversible.

#### **Encrypt:**

DSL Model Types (*BlkDef*, usage: DSL Model) can be encrypted. The *Encrypt* command encrypts all existing model equations and generates a non-editable encrypted text of the Equations page of the DSL Model Type.

---

**Note:** The user should be aware that encryption can never guarantee complete security. The chosen technology balances the requirements for security with the usability and performance of encrypted models. Generally, users are advised to share models only with trusted partners.

The **Encrypt** requires that the entire DSL Model Type is contained within itself (i.e. completely packed: no external macros, all model code within the object). To easily achieve this the subsequent execution of the *Pack* and *Pack -> Macro* commands can be performed.

---

**Note:** If the **Encrypt** button is deactivated, then the execution of *Pack* and *Pack->Macro* should enable the encryption. The Encryption function is functional only for licences containing the “DPL/D-SL/QDSL Encryption” module.

---

Upon a successful encryption, a DSL model type can be shared with any third party in an encrypted format. Note, the **Pack->Macro** command deletes any existing graphics (i.e. Block Diagram) of the DSL Model Type. It should be noted that the encryption command is irreversible and that no decryption function is available.

In order to **encrypt a DSL model type** perform the following actions:

- Create a back-up copy of the *PowerFactory* project, because the next steps cannot be undone; alternatively export your project (as backup and for later use) before you perform the encryption of the model;
- Activate the project;
- Open the dialog window of the DSL Model Type (*BlkDef*) you want to encrypt;
- Click the **Pack** button: this action copies all used macros into a sub-folder of the DSL Model Type (*BlkDef*);

---

**Note:** The *Pack* command dialog provides the option *Include objects from the DlgSILENT Library*. If this option is enabled, all referenced *DlgSILENT Library* objects will be packed, and their references will be updated accordingly to ensure consistency.

- Then click the **Pack->Macro** button: this action removes the graphics and packs all equations into one single DSL Model Type (*BlkDef*) object. Any pre-existing DSL macro references are removed.
- Afterwards, click the **Encrypt** button in order to encrypt the DSL Model Type (*BlkDef*).

- Note:**
- Composite Model Frames cannot be encrypted. In order to obfuscate the internal realisation of a composite model frame, it is possible to delete the Graphic sub-folder (folder type *IntGrfnet*) located inside the Composite Model Frame object (*BlkDef*, usage: Composite Model Frame).
  - The DSL *Configuration Script* is not encrypted using this function. To encrypt the *Configuration Script* assigned to a DSL Model Type, please perform the encryption process from within the script itself.

On the *Equations* page the (additional) equations of the DSL model can be provided. Further information such as the initial conditions of state variables and the name and unit of parameters can be specified. For DSL macros, this field is used to provide all the DSL equations that describe the macro functionality. Figure 30.5.1 shows the additional equations for the DSL model of the basic first-order limiter block.



Figure 30.5.1: *DSL Model Type* dialog, *Equations* page

### 30.5.2.3 Description page

A general purpose *Description* page is available for each *DSL Model Type*.

### 30.5.2.4 Version page

A *Version* page is available for each *DSL Model Type*.

*Copyright* field: This field allows to set a watermark/prove the ownership of a model. The field is password protected and can be customised by the user. The following information is relevant:

- The copyright can be configured by clicking the Set button. If done so, the copyright information can be entered. After typing in the copyright text, click OK. A password entry dialog is afterwards provided, in which a user defined password can be entered.

**Note:** The Set button is not available for *DSL Model Type* objects which have been created in a *PowerFactory* version prior to *PowerFactory* 2019.

- The entered copyright information is protected via the password. This object and any further copies of this object inherit the copyright information and the corresponding password.
- Copyright information can be changed at a later point in time by clicking the Change button and providing the original password;

- *DSL Model Type* objects which have been originally created in a *PowerFactory* version prior to *PowerFactory* 2019 are not supporting the *Copyright* field (even if used/imported in *PowerFactory* 2019 or later).
- *DSL Model Type* objects which are imported from a *PowerFactory* version equal to or greater than *PowerFactory* 2019 inherit the *Copyright* field information.

The *DIGSILENT* library objects (e.g. DSL macros, DSL control models) support a versioning system. For these objects, the following fields are relevant:

- *Version*: a field that states the current object version.
- *Change log*: a text field that provides a history change log of all previous versions.

## 30.6 DSL: Creating *DSL Model Types* using Block Diagrams

A block diagram of a *DSL Model Type* (*BlkDef dsl*) is a graphical representation of a multi-input multi-output (MIMO) time differential equations system including non-linearities. The block diagram defines one or more output signals as a function of one or more input signals.

A block diagram may thus be described as:

$$(y_0, y_1, \dots) = f(u_0, u_1, \dots) \quad (30.10)$$

where  $y_0, y_1, \dots$  represent output signals of index 0, 1, ... and  $u_0, u_1, \dots$  represent the input signals of index 0, 1, ... .

Block diagrams consist basically of the following elements:

- **Block References** which are used to include other *DSL Model Types*.
- **Summation Points** which produce the single output  $y=(u_0+u_1+\dots)$
- **Multipliers** which produce the single output  $y=(u_0*u_1*\dots)$
- **Divisors** which produce the single output  $y=(u_0/u_1/\dots)$
- **Switches** which produce the single output  $y=u_0$  or  $y=u_1$
- **Input/Output nodes** used to define an input (green color) or an output (red color) of relevant components in the block diagram (e.g. inputs/outputs of Block References, Summation points, switches etc.). The nodes are color coded:
  - **Red**: Output node (shown on the right side of an element e.g. *Block Reference*)
  - **Green**: Input node (shown on the left side of an element e.g. *Block Reference*)
  - **Magenta**: Input node, maximum limiting signal (shown on the top side of an element e.g. *Block Reference*)
  - **Blue**: Input node, minimum limiting signal (shown on the bottom side of an element e.g. *Block Reference*)
  - **Gray**: Undefined node type
- **Routed signal lines** used to link an output signal node with an input signal node
- **Signal labels** used to link signals without creating a routed signal

*Block references* can be looked upon as macros that insert a low-level *DSL Model Type* in the graphical representation of the hierarchically higher level *DSL Model Type*. A *Block reference* always points to another *DSL Model Type* (which can be graphically defined or using DSL code e.g. a DSL macro). *PowerFactory* is shipped with a large set of *DSL Macros* useful for most common controller elements like PID-controllers, dead-bands, valve characteristics, etc., and can be found in the *PowerFactory* tree

under DlgSILENT Library → Dynamic Models → Macros. These macros are provided in an open format, where the DSL code is readily accessible. A *Block reference* is added to the block diagram by using the  icon in the *Drawing Tools*. This creates an empty block which can then refer to any existing *DSL Model Type* in the global/project library. When the *Type* of the *Block reference* is assigned a *DSL Model Type*, *PowerFactory* inserts all available parameters of the referred type. The user may change the name of any parameter, however ensure that the order of the parameters is not changed. The order is important so that the right parameter is assigned to the parameters inside the *DSL Model Type*.

After a *Block reference* has been created, it will show any relevant input, output nodes. Routed signal lines may then be connected to these points. It is allowed to refer to the *DSL Model Type* more than once in the same block diagram. This way, it is possible to use a particular PID-controller, for instance, twice or more in the same model. Routed signal lines are directional branches, connecting one output with one or multiple input signals.

An example of a simple block diagram, comprising a multiplier, a summation point and a standard PI block, is shown in Figure 30.6.1.

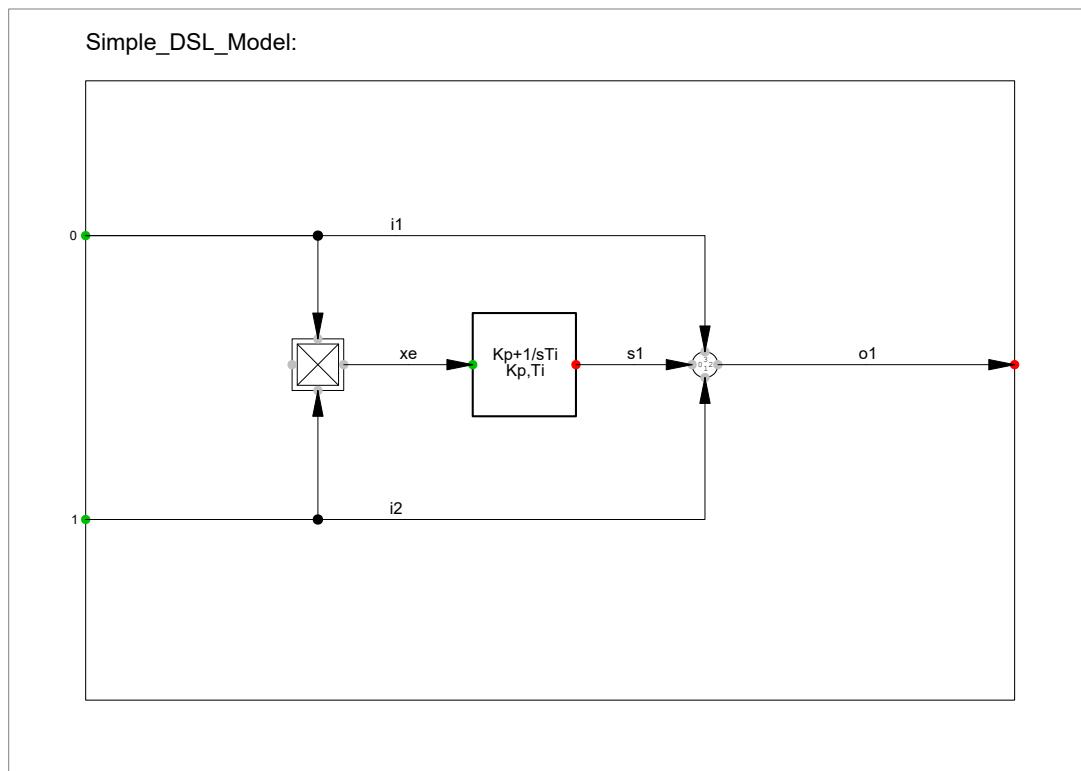


Figure 30.6.1: Example of a Simple Block Diagram

When rebuilding a diagram (by pressing the  icon), the DSL representation of the block diagram is written to the output window. For the example block diagram in Figure 30.6.1, this results in the following output:

```
o1 = 'Simple_DSL_Model'(i1,i2;x;Kp,Ti;s1,xe)
s1 = '    Kp+1/sTi'(xe;x;Kp,Ti)
xe = i1*i2
o1 = s1+i2+i1
```

This simple example shows the whole meaning of the block diagram graphics: it is a convenient way to define specific controllers, based on standard components.

However, it is possible to define exactly the same block diagram by entering similar DSL code in the

*Equations* page of an empty *DSL Model Type* and thereby create the model without having a graphical representation.

### 30.6.1 Drawing Diagrams of *DSL Model Types*

The creation of graphical *DSL* models is based on an intuitive and straightforward process. The development process involves using the available *Drawing Tools* by selecting building blocks from the *Library Browser* and drag-and-dropping the required blocks into the model diagram. Interconnecting these functional blocks is done using any of the specialised *Blocks and Signals* elements.

Inside the diagram of a *DSL Model Type* the following elements are allowed:

- *Block references*
- *Summation points*
- *Multipliers*
- *Divisors*
- *Switches*
- *Signal labels*
- *Routed signal lines*

These objects can be selected from the *Drawing Tools*. Additionally, the *Drawing Tools* contains graphical annotations as well (lines, polygons, rectangles, texts, etc.) as shown in Figure 30.6.2. The diagram itself must be taken out of *Freeze mode* ( ) in order to enable the *Drawing Tools*.

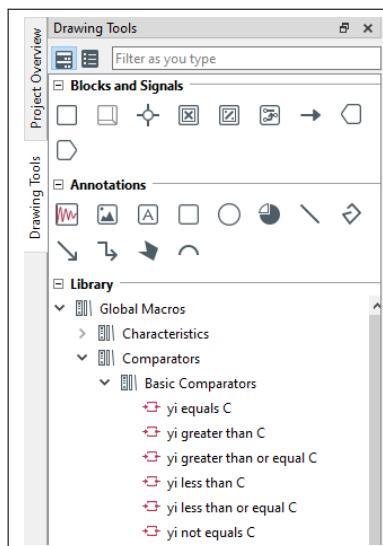


Figure 30.6.2: *Drawing Tools* toolbar, *Blocks and Signals* and *Annotations* sections

#### 30.6.1.1 Inserting a *DSL Macro* into a Diagram of a *DSL Model Type*

A *DSL macro* is inserted into the block diagram of a *DSL Model Type* by means of adding a *Block Reference*, which is the actual instance that links to the *DSL macro*. As a *DSL macro* can be used several times within the same *DSL Model Type* or in different *DSL Model Types*, the *Block Reference* maps the parameters, state variables, internal variables and signals, which are defined in the *DSL macro* with the associated parameters, state variables, internal variables and signals of the *DSL Model*.

Type. Therefore, after adding a *DSL macro* via *Block Reference*, the parameters of the *Block Reference* usually have to be customised to fit within the higher-level *DSL Model Type*; see next Section 30.6.1.2.

To add a *DSL macro* (i.e. a *Block Reference* that links to the *DSL macro*) in the diagram of a *DSL Model Type*:

- Option 1 - inserting pre-defined *DSL macros* using the *Drawing Tools*

The *Drawing Tools* toolbar contains a library browser that lists all *DSL macros* from the global *DlgSILENT Library* (see Section 14.10), from an additional custom global library (if existing, see Section 14.3) and any user-defined *DSL macros* from the project library's *Dynamic Models* folder. Sub-folders and their contents are displayed in a tree structure. For quick access, it is possible to search for particular macros using the filter function of the *Drawing Tools* toolbar. With a right-click, it is possible to show the contents of a selected *DSL macro*.

- Open the *Drawing Tools* toolbar and find the *Library* section. Navigate to a folder which contains the *DSL macro* of interest. You may use the filter function of the *Drawing Tools* toolbar.
- Insert a *DSL macro* into the block diagram of a *DSL Model Type* using *drag & drop*:
  - \* Select the *DSL macro* which should be inserted.
  - \* Left-click on the *DSL macro* and keep the left mouse button pressed.
  - \* Drag the *DSL macro* into the diagram.
  - \* Drop it at the intended location by releasing the left mouse button.
- Or: Insert a *DSL macro* into the block diagram of a *DSL Model Type* using *select & click*:
  - \* Select the *DSL macro* which should be inserted.
  - \* Left-click the *DSL macro* and release the left mouse button. The *DSL macro* is now “stuck” to the mouse cursor.
  - \* Move the mouse to the location in the block diagram, where the *DSL macro/Block Reference* should be deployed.
  - \* Left-click again to deploy the *DSL macro* (i.e. to create a *Block Reference* that refers to the *DSL macro*). This action can be repeated to insert the *DSL macro* at several locations within the block diagram.
  - \* Right-click or press **ESC** on your keyboard to release the *DSL macro* from the mouse cursor (i.e. to finish the mode of inserting the *DSL macro*).

- Option 2 - manually creating the block using the *Drawing Tools*:

- Click on the *Block reference* item in the *Drawing Tools*.
- Click in an empty area of the diagram to deploy it.
- Use the *Select* button (▼ in Figure 30.6.3) to select a *Type*. The type of a *Block Reference* is a *DSL macro*. Pre-defined *DSL macros* are located in the folder *DlgSILENT Library* → *Dynamic Models* → *DSL Macros* (see Section 14.10). User-defined *DSL macros* are typically stored in the local library of a project (*Project* → *Library* → *Dynamic Models*), or in an additional custom global library (if one exists; see Section 14.3).

---

**Note:** The associated object classes and their interdependencies are:

- *DSL Model Type*: *BlkDef* (Block Definition). The *DSL Model Type* can be defined graphically or by pure *DSL* text code.
- *Block Reference*: *BlkRef*. The *Block Reference* is the instance of a *DSL macro* within a higher-level *DSL Model Type*.
- *DSL Macro*: *BlkDef*. A *DSL macro* is a *DSL Model Type* with the option *Macro* enabled.

---

The edit dialog of a *Block Reference* is shown by double clicking the block (refer to Figure 30.6.3).

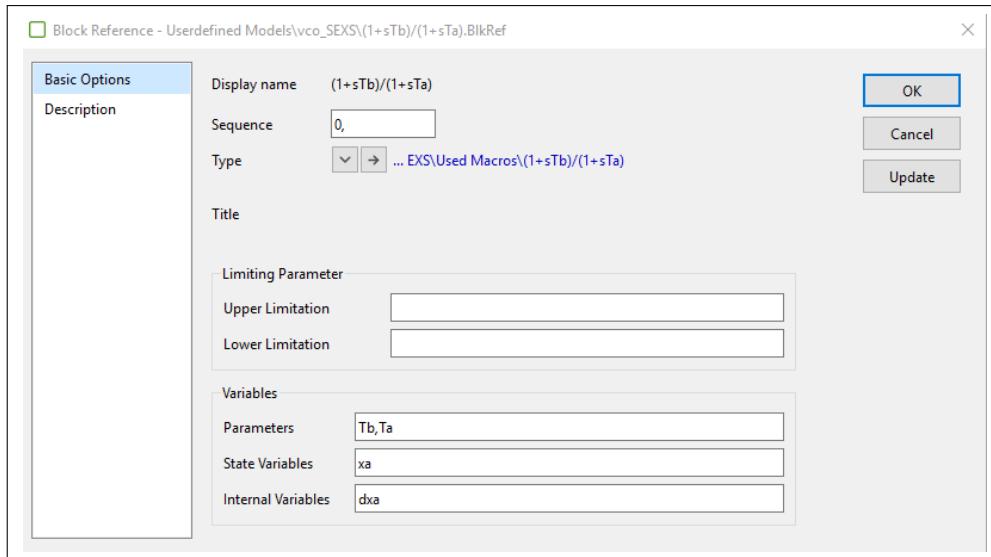


Figure 30.6.3: Block Reference dialog

### Adjusting Block References

*Block references* can be resized by clicking once on the block and then holding and dragging from one of the corners. The aspect ratio is maintained in this situation. When holding and dragging from one of the sides, the block is adjusted only in one direction.

#### 30.6.1.2 Customisation options for Block References

The **Block Reference** options are detailed as follows (refer to Figure 30.6.3):

- **Display name** - a read only field indicating either the *Display Name* of the referenced *DSL Macro* or the internally assigned object name of the newly added *Block Reference* in case the *Display Name* field of the referenced object (*BlkDef dsl*) is empty. The *Display name* can be used as an indication of the functionality of this *Block Reference*. If a *DSL Macro* is assigned to the block (i.e. the referenced *BlkDef* must have the flag *Macro* set), then the *Display name* (if not empty) is added to the description label of the *Block Reference* in the diagram. If the *Display name* is empty then the object name of the *DSL Macro* is shown instead in the diagram.
- **Sequence** - This parameter accepts any integer number. It defines the order in which the block's parameters are listed in a *DSL Model* that uses the main *DSL Model Type*. The *DSL Model* dialog displays a listing of all parameters within the referenced *DSL Model Type* in a tabular format. The order in this table, from top to down, is decided by this *Sequence* parameter: the parameters belonging to the *Block Reference* with the smallest *Sequence* value will appear on top of the list (first rows); the parameters belonging to the *Block Reference* with the second smallest *Sequence* value will follow the first ones in the list (subsequent rows), and so on.

- Note:**
- Multiple parameters belonging to the same *Block Reference* are added in their order of appearance in the field *Parameters* of the *Block Reference* edit dialog.
  - The *DSL Model* lists (and orders) parameters based on their functional scope. That is:
    - \* the first block of ordered parameters in the list belongs to those parameters defined in any *Block Reference* in the field *Parameters* of each *Block Reference* edit dialog.
    - \* the second block of ordered parameters in the list belongs to those parameters defined in the field *Additional Parameters* of the *Equations* page of the main *DSL Model Type* edit dialog.
    - \* the third block of ordered parameters in the list belongs to those parameters defined in any *Block Reference* in the field *Lower Limitation* of each *Block Reference* edit dialog.

- \* the fourth block of ordered parameters in the list belongs to those parameters defined in any *Block Reference* in the field *Upper Limitation* of each *Block Reference* edit dialog.
  - Inserting new *Block References* to the diagram will index upwards the *Sequence* value of each newly added *Block Reference* (the highest value of any existing *Block Reference* plus 1 is assigned to the last inserted block)
- 
- **Type** - Reference to a *DSL Model Type* (mandatory). When assigned, the *Block Reference* automatically reads the inputs and outputs of the *DSL Model Type* and applies them graphically by adding input and output nodes to the block. Whenever the *DSL Model Type* is changed by accessing this field, the *Block Reference* is automatically updated. If the changes to the *DSL Model Type* are externally applied, then the block diagram must be *Rebuild* using the **Rebuild** button.
  - **Title** - a read only field which displays the *Title* field of the referenced *DSL Model Type* ).
  - **Upper Limitation** - the *Upper Limitation* limiting parameters are automatically read from the referenced *DSL Model Type*. If none, then the field is left empty. The **Upper Limitation** field is initially set to a comma separated list of the originally declared parameter names in the referenced *DSL Model Type*. The parameter names can be directly changed in this field. Although names can be different, the order and number of the comma separated parameters is important and must correspond to the order and number of the declared parameters in the referenced *DSL Model Type*.
  - **Lower Limitation** - the *Lower Limitation* limiting parameters are automatically read from the referenced *DSL Model Type*. If none, then the field is left empty. The **Lower Limitation** field is initially set to a comma separated list of the originally declared parameter names in the referenced *DSL Model Type*. The parameter names can be directly changed in this field. Although names can be different, the order and number of the comma separated parameters is important and must correspond to the order and number of the declared parameters in the referenced *DSL Model Type*.
  - **Parameters** - the *Parameters* are automatically read from the referenced *DSL Model Type*. If none, then the field is left empty. The **Parameters** field is initially set to a comma separated list of the originally declared parameter names in the referenced *DSL Model Type*. The parameter names can be directly changed in this field. Although names can be different, the order and number of the comma separated parameters is important and must correspond to the order and number of the declared parameters in the referenced *DSL Model Type*.
  - **State Variables** - the *State Variables* are automatically read from the referenced *DSL Model Type*. If none, then the field is left empty. The **State Variables** field is initially set to a comma separated list of the originally declared state variables of the referenced *DSL Model Type*. The state variable names can be directly changed in this field. Although names can be different, the order and number of the comma separated state variables is important and must correspond to the order and number of the declared states in the referenced *DSL Model Type*.
  - **Internal Variables** - the *Internal Variables* are automatically read from the referenced *DSL Model Type*. If none, then the field is left empty. The **Internal Variables** field is initially set to a comma separated list of the originally declared internal variables of the referenced *DSL Model Type*. The internal variable names can be directly changed in this field. Although names can be different, the order and number of the comma separated internal variables is important and must correspond to the order and number of the declared internal variables in the referenced *DSL Model Type*.
  - **Update** button - the *Update* button is used to re-read and re-initialise the variable fields (upper/lower limitation, parameters, states and internal variables) according to the information in the referenced *DSL Model Type*. Upon clicking this button, all previous data of the variable fields is removed.

### 30.6.1.3 Adding Calculation Blocks

#### Summation Point

Any node can be used as an input or an output node. Only one output node is allowed. Not all nodes must be used. The sign of input nodes can be changed from the edit dialog of the summation point.

#### Multiplier

Any node can be used as an input or an output node. Only one output node is allowed. Not all nodes must be used. For example, the output of three input signals is:  $out = in_0 * in_1 * in_2$ .

#### Divisor

The left side node is used for the numerator input signal. Any subsequent node (counting clockwise) can be connected to an input. These input signals will be used as denominator terms. Only one output node is allowed. Not all nodes must be used.

#### Switch

Two input signals can be applied to this block, which will be connected to the output according to the value of the control signal (top node). If the control signal is 0.5 or less, the switch stays in the default state, whereas a signal greater than 0.5 will cause the switch to change state. The default state can be changed via the *Edit* dialog of the switch.

### 30.6.1.4 Connecting Signals

After drawing and defining the *Block references*, slots or other node elements, they can be connected with *Routed signal lines* or *Signal labels*. After selecting the  button from the *Drawing Tools*, a *Routed signal line* is drawn by clicking on the *Output node* of a block and afterwards clicking on the *Input node* of another block (an example is shown in Figure 30.6.4).

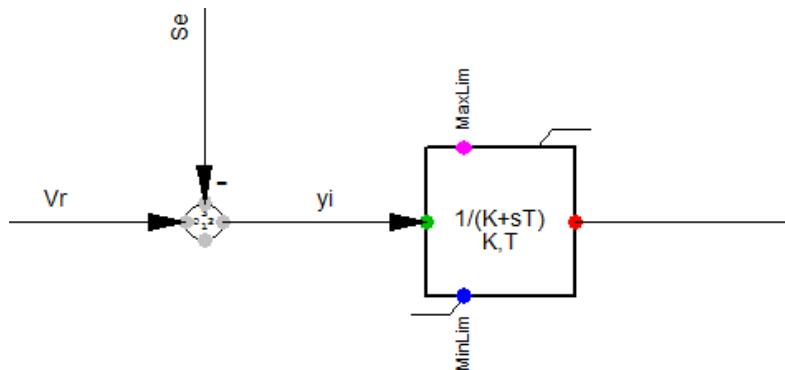


Figure 30.6.4: Routed Signal Lines of a *DSL Model Type* diagram

The routed signal lines can be edited in the corresponding dialog, which allows signal names to be changed.

### 30.6.1.5 Input and Output Signals of Block Diagrams

An input signal is defined by starting a *Routed signal line* from the left, top or bottom side of the *Boundary frame* of the diagram.

An output signal is defined by ending a *Routed signal line* on the right side of the *Boundary frame* of the diagram.

---

**Note:** The names of the input and output signals must be the same as the names of the input and output signals defined in the slot or block to which it is intended to use this block diagram.

---

### 30.6.1.6 Using *Signal labels*

Nodes (e.g. of *Block References*) can be interconnected using a combination of *Signal labels* and *Routed signal lines* as well. *Routed lines* can be “split” by *Signal labels* thus allowing inter-connection of nodes without drawing a continuous line between the source and destination *Node*. *Signal labels* can be regarded as “shortcuts” and come in handy especially in cases of highly cluttered diagrams of *DSL Model Types*.

To add *Signal labels* to a diagram of a *DSL Model Type*, do the following:

- Activate the *Goto* drawing mode by clicking the *Goto* (□) icon in the *Drawing Tools*. *Goto* labels are used to be linked with an *Output node* of a source component (e.g. a *Block Reference*). The signal of the *Goto* label will be later on linked with the *From* label and finally conveyed to the *Input node* of a destination component (e.g. a *Block Reference*).
- Deploy the *Goto* label in an empty area next to the component's *Output node* which is of interest.
- Activate the *Signal* drawing mode by clicking the *Signal* (→) icon in the *Drawing Tools*. Connect the *Output node* of the component with the *Input node* of the *Goto* label (□). Exit the *Signal* drawing mode by right-clicking in an empty area.
- Give a relevant name to the recently created *Routed signal line* (the signal name is further referred to “*SignalName*”).
- Activate the *From* drawing mode by clicking the *From* (□) icon in the *Drawing Tools*. *From* labels are used to link an externally named signal (e.g. of a *Goto* label for example) with an *Input node* of a destination component (e.g. a *Block Reference*).
- Deploy the *From* label in an empty area next to the component's *Input node* (destination signal) of interest.
- Activate the *Signal* drawing mode by clicking the *Signal* (→) icon in the *Drawing Tools*. Connect the *Output node* of the *From* label (□) with the *Input node* of the destination component (e.g. a *Block Reference*). Exit the *Signal* drawing mode by right-clicking in an empty area.
- Double click the newly created *From* label (□). In its Edit dialog, set the text field *Signal from* to be “*SignalName*”.
- A link is created between the *Output node* of signal source component (e.g. a *Block Reference*) and the *Input node* of a signal destination component (e.g. a *Block Reference*) (refer to Figure 30.6.5 for a simple example).

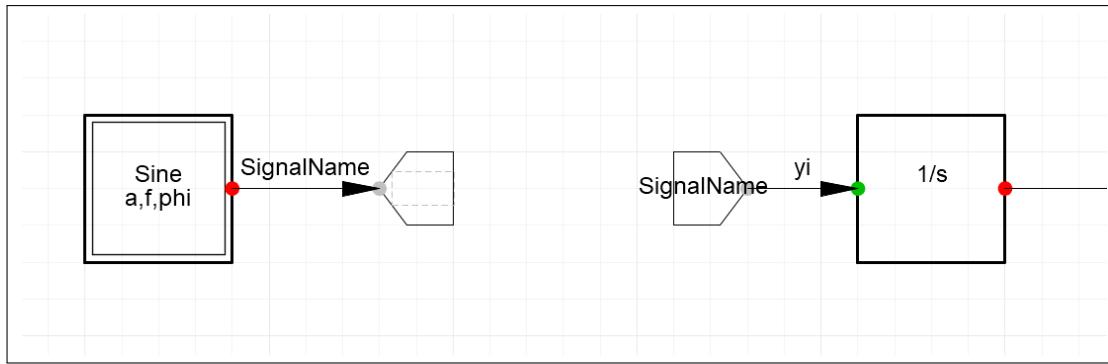


Figure 30.6.5: Using *Signal labels* to inter-connect *Output* and *Input nodes*

### 30.6.1.7 Equations page of the *DSL Model Type*

The drawn block diagram of a *DSL Model Type* typically describes the dynamic behaviour of the model. There are cases in which additional instructions must be added to the model in order to fully characterise its behaviour e.g. definition of initial conditions (using `inc()` function), definition of various variable units and associated descriptions (using `vardef()` function). For this purpose, the *Equations* page can be shown by left or double-clicking the enclosing rectangular boundary or by double clicking in an empty area of the diagram.

## 30.7 DSL: Coded *DSL Model Types* (non-graphically defined)

Although less common, a *DSL Model Type* (`BlkDef dsl`) can be programmed entirely using DSL code within the *Equations* page of the Edit Dialog. It can fully represent a dynamic system and include the following functionality:

- Parameter descriptions: name and unit
- Allowed parameter ranges
- Initial conditions and functions which are used to calculate initial values.
- Algebraic and differential equations defining the dynamic model during simulation.

The usability of this alternative is restricted to models not exceeding several dozens of lines of code. A more complex model is advised to be integrated using block diagrams, as model readability is an issue with larger code snippets. An example of a time-continuous integrator with time constant  $T_i$  is shown further below:

```
1. vardef(Ti) = 's';'Time constant' ! unit and description of parameter Ti
2. limfix(Ti) = (0,) ! a message is printed to output window if Ti<=0
3. inc(x) = yo ! initialisation statement: state variable x=yo
4. x. = yi/Ti ! equation 1 (applied during simulation): define the state
variable derivative equation
5. yo = x ! equation 2 (applied during simulation): assign output yo = x
```

## 30.8 Modelica: Integrating Modelica Models into a Simulation

### 30.8.1 Modelica Models and Modelica Model Types

Dynamic models, as discussed in Section 30.1, can be of various types. Invariably, the implementation of *User-defined* dynamic models is realised using one of the following two options:

- using DSL specific objects: *DSL Model* (*ElmDsl dsl*) and *DSL Model Type* (*BlkDef dsl*), or
- using Modelica specific objects: *Modelica Model* (*ElmMdl mdl*) and *Modelica Model Type* (*TypMdl mdl*)

If discussing the second option, the implementation of a *User-defined* dynamic model follows the modelling paradigms specified by the [Modelica language](#), as published by the [Modelica Association](#). The functionality supported in *PowerFactory* is described in Section 30.9.

The integration of a Modelica dynamic model conceptually follows the same *Type/Element* principle found throughout *PowerFactory*, and graphically depicted in Figure 30.8.1. It is apparent that the *Modelica Model* (*ElmMdl, mdl*) represents one instance in the power system of a user-defined *Modelica Model Type*. A *Modelica Model* is integrated within a slot of a *Composite Model* (a high-level control system) in order to be interconnected with other control relevant network components, with whom the *Modelica Model* will exchange input and output signals, as defined in the *Composite Model Frame*.

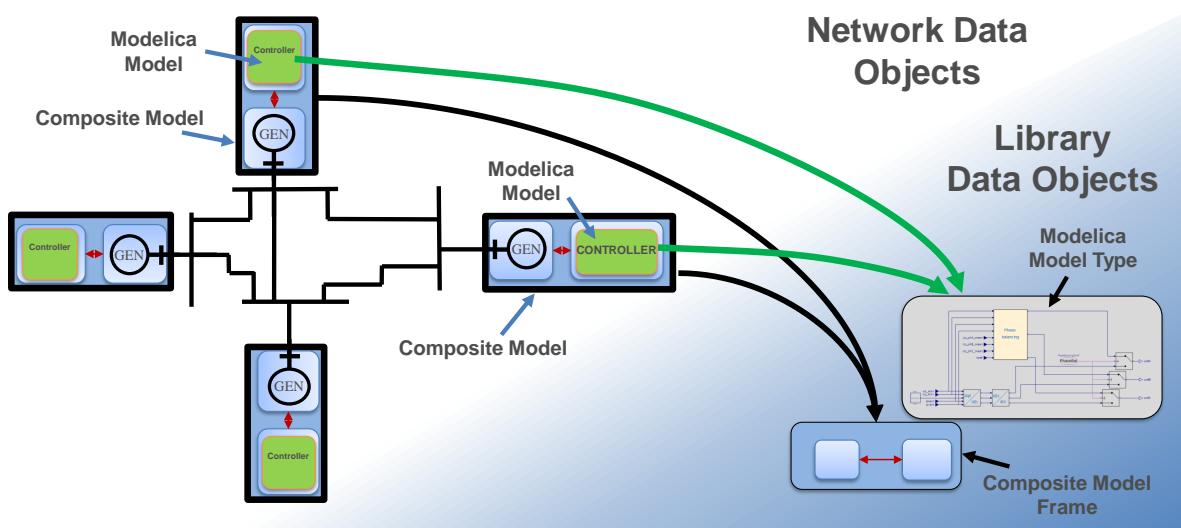
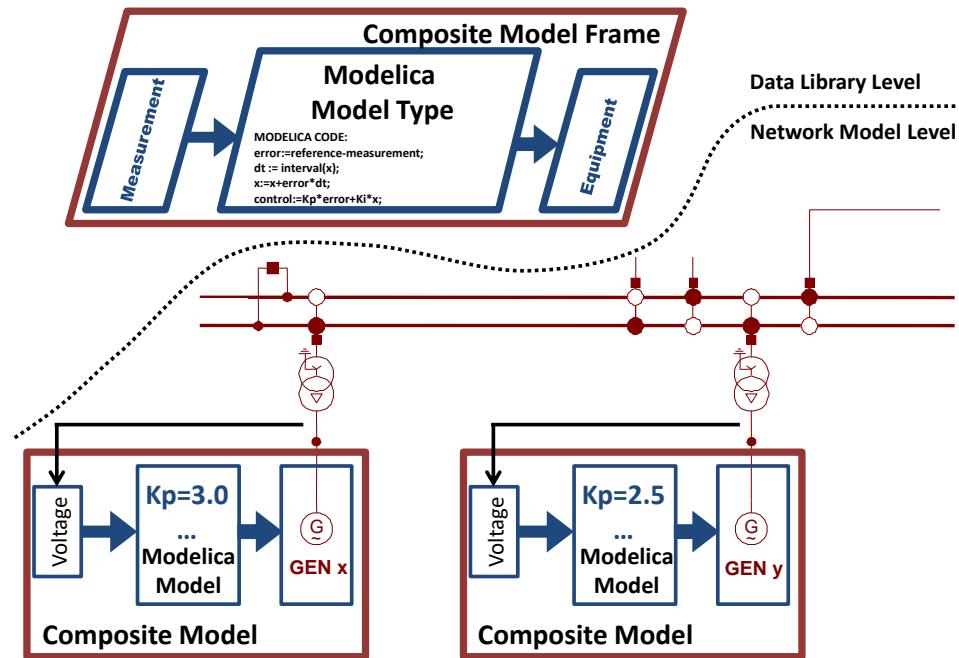


Figure 30.8.1: Integration of *Modelica Models* into a Simulation

A *Modelica Model* (*mdl*) instantiates a *Modelica Model Type* (*mdl*) with a specific set of parameter values. The initial parameter values are either zero or the default ones, if existing. Default parameter values are those pre-configured in the *Modelica Model Type* (refer to option *Show default/start values of parameters/variables in “Variable Declarations”*). The benefit of using a *Modelica Model* that stores the actual model parameter values is illustrated in Figure 30.8.2. It is apparent that two individual *Modelica Models* can be defined and individually parameterised while referencing the same controller structure (the same *Modelica Model Type*) and controlling different generators.

Figure 30.8.2: Individual model parameterisation using *Modelica Models*

Furthermore, *Modelica Models* have the following characteristics:

- Any number of *Modelica Models* can be defined in a power system model.
- A *Modelica Model* can reference only one *Modelica Model Type*.
- A *Modelica Model Type* can be referenced multiple times, by any number of *Modelica Models*.
- A single *Modelica Model* can be assigned to one *Slot* of a given Composite Model.
- Although seldom the case, one single *Modelica Model* can be assigned to *Slots* belonging to multiple Composite Models. In such a case, the *Model developer* needs to make sure that the input signals that are connected to the *Modelica Model* are unique e.g. it is not allowed to connect the same input signal twice, once with a component of one Composite Model, then with another one of a different Composite Model.

Parameters can be either scalar or multi-dimensional, as follows:

- Scalar parameters e.g. gains, time constants, set-points. Note, in order that a parameter is explicitly declared as a scalar, the *Size* field must be empty.
- Array parameters: one- and two-dimensional array parameters. In order that a parameter is declared as an array, the *Size* field must contain a positive integer.

If the referenced *Modelica Model Type* (*TypMdl* *mdl*) defines one or more array parameters, then these arrays can be customised within the *Modelica Model*, using the tab “*Arrays*” of the Edit Dialog of the *Modelica Model* (*ElmMdl* *mdl*). Array parameters are defined using references to *IntMat* objects. The *IntMat* objects will contain the actual arrays. A parameter array is a parameter variable whose *Size* field is declared (it is assigned a value). Note, parameter arrays of size 1 can be defined by declaring a parameter with *Size* equal 1. In order that a parameter is explicitly declared as a scalar, the *Size* field must be empty. The array characteristic must be defined within an external object of type *IntMat*. The *IntMat* object should be stored inside the *Modelica Model* (although not mandatory)

### 30.8.2 Creating a new *Modelica Model*

To create a new *Modelica Model* (*ElmMdl* ), the following actions are required:

- using the Data Manager:
  - Make sure that the Data Manager active directory is inside the folder “Network Model”
  - Click the *New Object* ( ) icon and select *Modelica Model*.
  - Give the model a relevant name.
  - Assign to the “Type” field a corresponding *Modelica Model Type* (*TypMdl*)

### 30.8.3 Saving and plotting model variables

It is many times useful to record *Modelica Model* variables for various purposes e.g. plotting variable against time. To record one or several *Modelica Model* variables, they need to be explicitly defined as *monitored variables* belonging to a *Results* object (*ElmRes*). To display the current values of model variables, the *Network Model Manager* can be used. The process of monitoring results in a dynamic simulation is described in detail in Section 29.5.1. The process of plotting the monitored variables is detailed in Section 29.8.

### 30.8.4 Support of *PowerFactory* functionalities

Modelica models, in addition to the Modelica Language specification, support the following *PowerFactory* specific functionalities.

#### 30.8.4.1 Initialisation

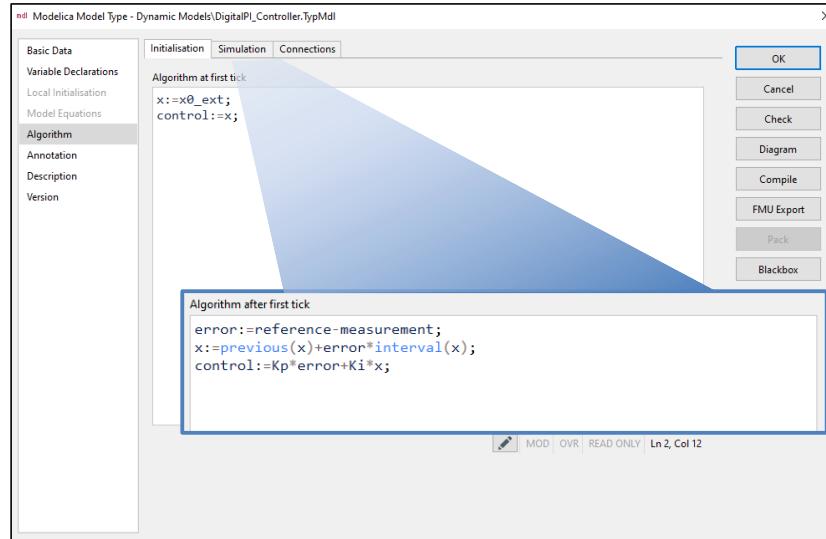
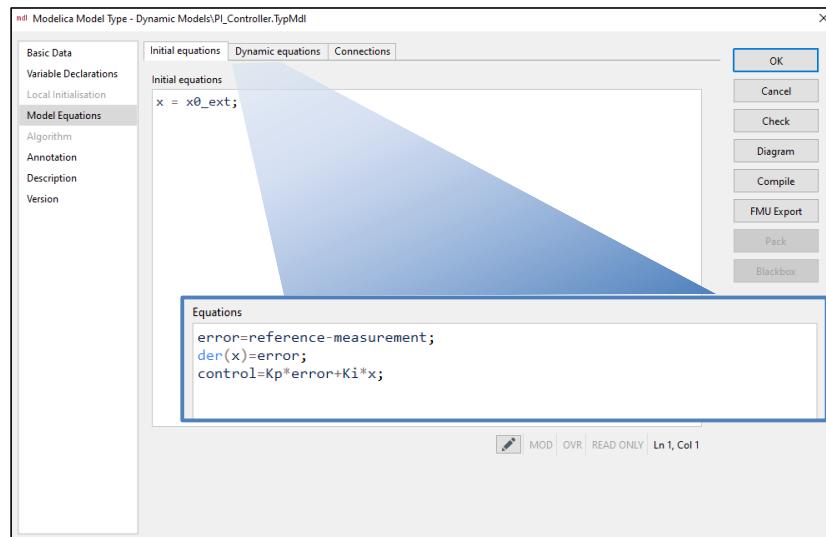
The Modelica Model Types (*TypMdl*) can be initialised to a steady-state operating point, which can be determined by a load flow calculation (refer to Section 30.10.2.9)

#### 30.8.4.2 Parameter events (*EvtParam*)

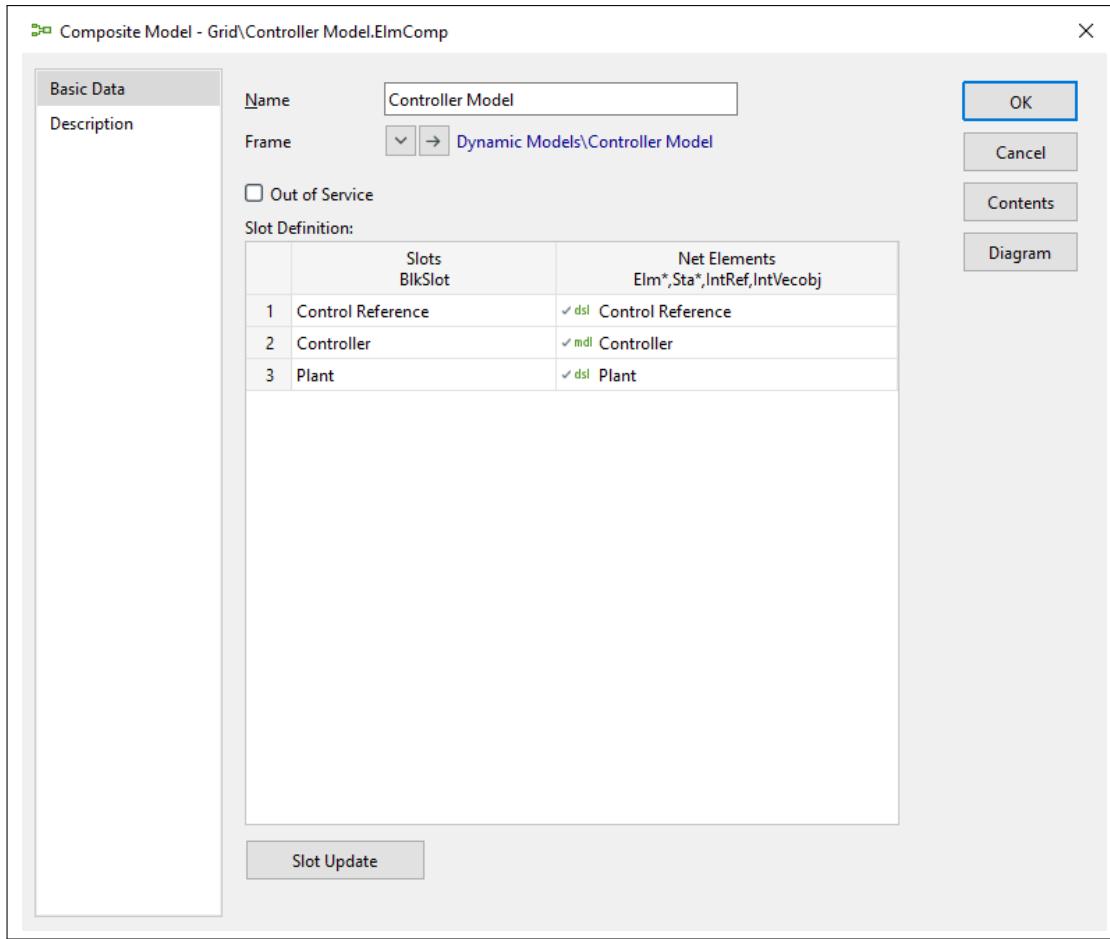
Parameter events (*EvtParam*) can be applied on a scalar *Input* variable of Real, Integer and Boolean types. This event changes the input value at the event’s execution time, provided that the input variable is not connected and has a declared start value.

### 30.8.5 Example of a *Modelica Model* implementation

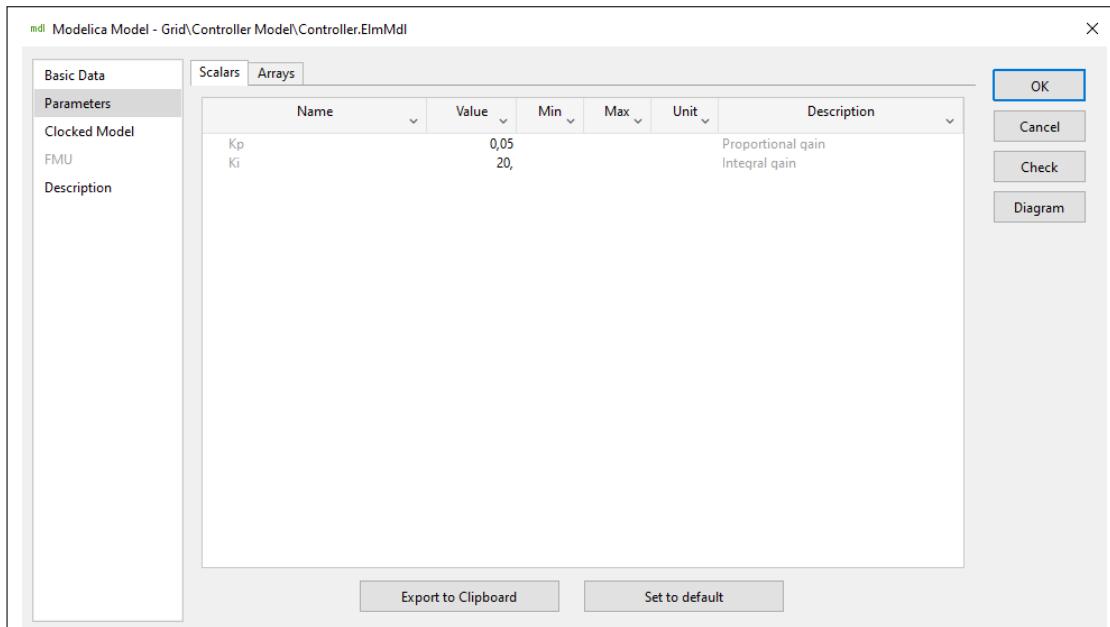
Examples of *Modelica Model Types* are shown in Figure 30.8.3 and Figure 30.8.4, where the former Figure implements a sampled discrete proportional-integral controller and the later Figure implements a time-continuous proportional-integral controller using Modelica code. Alternatively, Modelica models can be conveniently developed using the graphical modelling environment described in Section 30.10.

Figure 30.8.3: *Modelica Model Type* of a general purpose discrete PI regulatorFigure 30.8.4: *Modelica Model Type* of a general purpose time-continuous PI regulator

A *Modelica Model* referencing the previously exemplified type thus “instantiates” the model type in the power system. As a result, the *Modelica Model* can be assigned to a slot of a relevant *Composite Model* (e.g. see Figure 30.8.5) and thus be connected to an element to be controlled.

Figure 30.8.5: Integration of the *Modelica Model* into a *Composite Model*

Furthermore, users can customise these parameters by assigning values and applying them to the controller by configuring the *Modelica Model*, as shown in Figure 30.8.6.

Figure 30.8.6: *Modelica Model* dialog with parameterisation of a PI regulator

## 30.9 Modelica: A Non-proprietary, Object-oriented, Equation-based Language

### 30.9.1 Modelica: Overview of an Open and Comprehensive Systems Modelling Language

The [Modelica language](#) is a state-of-the art systems modelling language. Its characteristics are that it is: non-proprietary, object-oriented, equation-based. Using Modelica, it is possible to represent complex systems covering a broad range of physical and systems control domains. A long list of [open-source and commercially available tools](#) provide built-in support for Modelica. Using the current framework of *PowerFactory*, it is possible to design, deploy and use sampled discrete-time (so called “*synchronous systems*”) and continuous-time (in addition to DSL) dynamic simulation models using Modelica language. The implementation supports both RMS- and EMT-type simulation domains.

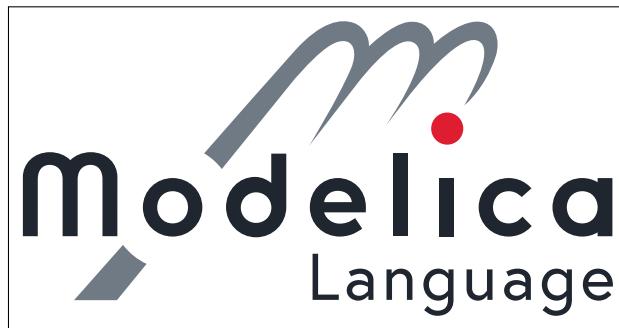


Figure 30.9.1: Modelica Language - a comprehensive systems modelling language

### 30.9.2 Supported *Modelica* functionality

With reference to the [Modelica language specification](#) (version 3.5), the functionality described in the following sections of the Modelica specification is currently partially supported (mainly oriented for representing *Synchronous Systems* - time discrete models):

- Introduction (Chapter 1)
- Lexical structure (Chapter 2)
- Operators and expressions (Chapter 3) (except specific parts of Sections 3.6.3, 3.7.1, 3.7.4, 3.7.5)
- Equations (except specific parts of Sections 8.3.2, 8.3.5, 8.3.7, 8.3.8)
- Arrays (Chapter 10) (except specific parts of Sections 10.1, 10.3.5, 10.4.1, 10.4.2, 10.5, 10.5.2)
- Statements and Algorithm Sections (Chapter 11) (except specific parts of Sections 11.2.1.1, 11.2.2.1, 11.2.2.3, 11.2.4, 11.2.5, 11.2.7)
- Synchronous Language Elements (Chapter 16) (partially supported using the built-in dialogs)
- Annotations (Chapter 18) (partially supported for graphical objects)

### 30.9.3 Terms and definitions

Please refer to the [Modelica Language specification](#) for a complete description of the terms and definitions used within Modelica. Within this section, only the most relevant terms are reviewed. For several *PowerFactory*-specific terms (not existing in the Modelica Language), a relevant description is provided for the sake of clarity.

**Block:** (Modelica-specific) representing a specialised Modelica *class*: same as the *model* class, with the restriction that each connector component of a *Block* must have prefixes input and/or output for all connector variables.

---

**Note:** The purpose is to model input/output blocks of block diagrams. Due to the restrictions on input and output prefixes, connections between blocks are only possible according to block diagram semantics.

---

**Component:** (Modelica-specific) A component is an instance of a Modelica class object.

**Modelica Model Type:** (*PowerFactory*-specific) representing the *Modelica Model Type* (*TypMdl*) *PowerFactory* object. It is used to define a Modelica *Block* in *PowerFactory* (refer to Section 30.8.1).

**Modelling Method Clocked:** (*PowerFactory*-specific) representing a clocked *Modelica Model Type* (*TypMdl*) *PowerFactory* object. It is used to define sampled discrete-time dynamic simulation models (so called “synchronous systems”) using Modelica statements/algorithms in *PowerFactory*.

**Modelling Method Hybrid (pilot version):** (*PowerFactory*-specific) representing a hybrid *Modelica Model Type* (*TypMdl*) *PowerFactory* object. It is used to define mixed continuous-time and discrete-time dynamic simulation models using Modelica equations in *PowerFactory*.

**Modelica Model:** (*PowerFactory*-specific) representing the *Modelica Model* (*ElmMdl*) *PowerFactory* object. Modelica models facilitate the interfacing between the Modelica modelling environment and the *PowerFactory* power system simulation model. The object stores Modelica *Component* data into *PowerFactory* and represents one instance of a *Modelica Model Type* object. It is mainly used for integration into High-level Control System representations (refer to Section 30.1) and interfacing it with other *PowerFactory* models via the *Composite Model* (refer to Section 30.8.1).

**Type:** (*PowerFactory*-specific) A *Type* is a *Modelica Model Type* for which one or several *Type Instances* may exist. Use *Types* to create a library of building blocks for block diagrams.

**Type Instance:** (*PowerFactory*-specific) The instance of a *Type*. Use *Type Instances* to add *Types* into Modelica block diagrams. *Type Instances* are automatically created inside a model whenever users are adding library blocks into a block diagram.

**Subsystem (class):** (*PowerFactory*-specific) A *Subsystem (class)* is a *Modelica Model Type*, which is restricted to have only one instance (the corresponding instance must be unique). Use a *Subsystem* to group components and add hierarchical structure.

**Subsystem (instance):** (*PowerFactory*-specific) The instance of a *Subsystem (class)*. Use a *Subsystem* to group components and add hierarchical structure to the model. Since there can only exist one instance of *Subsystem (class)*, the *Subsystem (instance)* is referred to throughout the text as a **Subsystem**, for the purpose of simplicity. *Subsystems* are always stored inside other *Modelica Model Type* objects and follow a strict hierarchical structure based on their location. Consequently, a *Subsystem* is a child object of another *Modelica Model Type* (the latter is known as the parent component of the *Subsystem*).

**Top-level diagram:** (*PowerFactory*-specific) Models defined using hierarchically structured block diagrams may contain several diagrams, each linked with specific diagrams via child-parent relationships. A top-level diagram represents in this context the highest level block diagram, which contains all subordinated diagrams of an hierarchical block diagram design.

**Top-level model:** (*PowerFactory*-specific) The associated *Modelica Model Type* (*TypMdl*) whose diagram is a *top-level diagram* is referred to as a *Top-level Modelica Model Type* or as a *Top-level model*. Inputs and outputs of a *Top-level model* are meant to interconnect it with other *PowerFactory* elements via *High-level Control System Representations* (as described in Section 30.2).

<b>PowerFactory term</b>	<b>Equivalent Modelica term</b>	<b>Usage</b>
<i>Modelica Model Type (TypMdl)</i>	Modelica <i>Block</i>	Store Modelica <i>Block</i> data in <i>PowerFactory</i>
<i>Modelica Model (ElmMdl)</i>	Not Available	Store Modelica Top-level model data in <i>PowerFactory</i> and allow interfacing with other <i>PowerFactory</i> models
<i>Type</i>	Modelica <i>Block</i>	Use <i>Types</i> to create a library of building blocks for block diagrams
<i>Type Instance</i>	Modelica <i>Component</i>	Use <i>Type Instances</i> to add <i>Types</i> into Modelica block diagrams
<i>Subsystem (class)</i>	Modelica <i>Block</i>	Use a <i>Subsystem</i> to group components and add hierarchical structure
<i>Subsystem (instance)</i>	Modelica <i>Component</i>	Use a <i>Subsystem</i> to group components and add hierarchical structure

Table 30.9.1: Mapping of *PowerFactory* and *Modelica Language* concepts

## 30.10 Modelica: Creating Modelica Models using Block Diagrams

In *PowerFactory* Modelica is used for the representation of control systems using block diagrams. With Modelica block diagrams, it is possible to represent any type of clocked digital or time-continuous models of controllers. These can range from slow-reacting wind power plant controllers (e.g. sample rate of 10 Hz) up to fast controllers for real-world power electronic converters (e.g. sample rates between 1-10 kHz) and can be integrated into large-scale power system models for analysis within an RMS- or EMT-simulation.

A block diagram of a *Modelica Model Type (TypMdl mdl)* is a graphical representation of a control system. The block diagram defines one or more output signals as a function of one or more input signals.

Block diagrams consist of the following elements:

- Instances of **Blocks**: *Components* that represent structural or functional elements within the model hierarchy and have inputs and/or outputs.
- **Connectors**: *components* used to connect signals with inputs and outputs of *Blocks*
- **Labels**: *components* used to create a connection between *Blocks*, which is not shown in the *Model Diagram*.
- **Signals**: used to interconnect inputs and outputs of *Blocks*, *Connectors* and *Labels*.
- **Annotations**: used to provide simulation non-relevant details that better explain or characterise the model's internal realisation.

A **Block instance** can be of the following form:

- **Type instance**: *Type instances* represent instances of *Modelica Model Types*. The referenced *Type* can be re-used within any number and across all hierarchical levels. It references a specific *Modelica Model Type* which contains all sub-ordinated model *components*. The referenced *Modelica Model Type* can be a basic Modelica Model (defined using Modelica Language) or may represent a sub-ordinated block diagram with a number of model *components*.
- **Subsystem**: a specialised *Type*, of which there exists only one instance (it is unique). Subsystems are intended to group model *components* in a single element within the model hierarchy. Subsystems are located inside *Modelica Model Types*. It references a specific *Modelica Model Type* which contains all sub-ordinated model *components*. A *Subsystem* is unique and cannot be referenced in multiple locations; copies of a *Subsystem* are independent from each other.

Consequently, building blocks within a Modelica block diagram (**Block** instances) can either be **Type Instances** or **Subsystems**.

### 30.10.1 Graphical environment for Modelica Models

The creation of graphical Modelica models is based on an intuitive and straightforward process. The main components of the graphical development environment of Modelica models are highlighted in Figure 30.10.1.

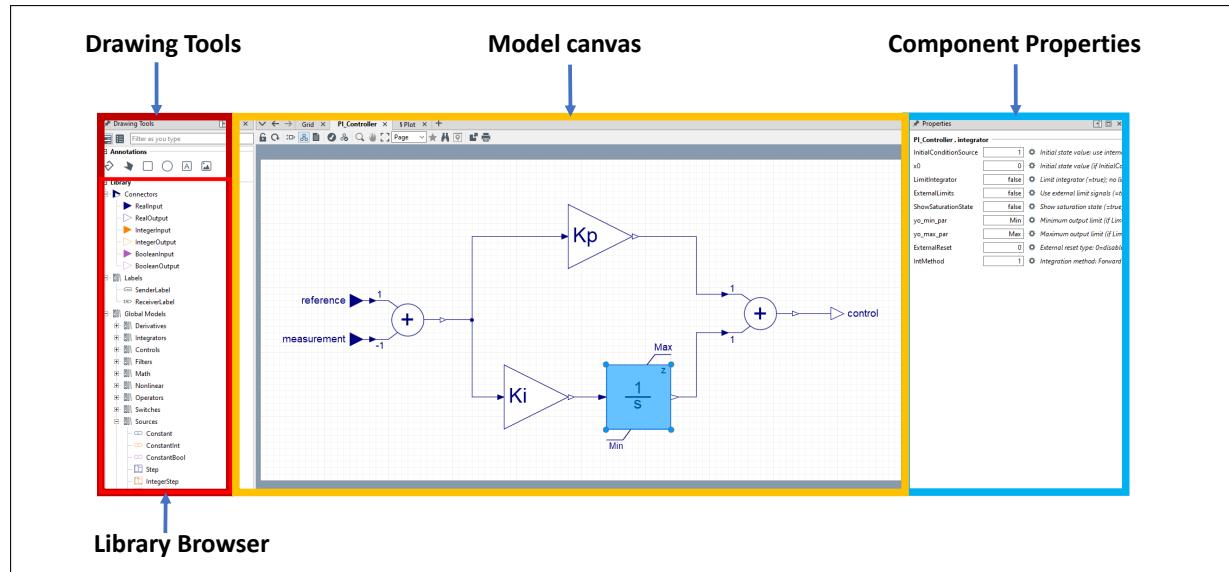


Figure 30.10.1: Overview of the Modelica models graphical environment

The development process involves selecting building blocks from the *Library Browser* and drag-and-dropping the required blocks into the model using the *Diagram view*. Configuration of the deployed building blocks (called *components*) can be directly performed via the *Properties Browser* (refer to Section 30.10.2.8).

The following toolbar buttons are available within the *Model canvas*:

- Graphical Freeze Mode : freeze/unfreeze the edit mode of the canvas (refer to Section 10.3.1);
- Rebuild : The graphic is rebuilt based on the currently stored graphic data of the model and any other dependent components (refer to Section 10.3.2);
- Icon View : change the view mode to *Icon view* in order to edit the icon of the model;
- Diagram View : change the view mode to *Diagram view* in order to edit the *Model Diagram*;
- Show Model Dialog : click this button in order to show the Edit dialog of the corresponding *Modelica Model Type*;
- Check Model : click this button to check the model for errors. *PowerFactory* will check for and report any found errors to the *Output Window*, as well as highlight in the *Model diagram* the invalid components;
- Highlight Invalid Components / : click this button to highlight invalid components within the model diagram, if any. Invalid components are highlighted with red. For detailed information on the particular error of each invalid component, click the *Check Model* button. If the *Highlight Invalid Components* mode is active () , then click the button again to deactivate this mode;
- Zoom In : activate the *Zoom in* mode in order to zoom into a certain region of the displayed view (refer to Section 10.3.3.1);
- Pan Tool : activate the *Pan Tool* in order to navigate within the displayed view. Alternatively, the *Pan Tool* is automatically active while clicking and holding the center mouse button anywhere within the *Model canvas* (refer to Section 10.3.3.4);
- Zoom All : click the *Zoom all* button in order to show the entire view (refer to Section 10.3.3.2);
- Zoom-level : adjusts the zoom level based on a number of presets, or a user defined zoom level (given in percentage) (refer to Section 10.3.3.3);

- View Bookmarks : shows the existing diagram bookmarks (refer to Section 10.3.3.5);
- Search in Diagram : searches within diagram for user defined keywords (refer to Section 10.3.3.6);
- Show Navigation Pane : toggles on/off the *Navigation pane* (refer to Section 10.3.3.7);
- Export Diagram : Exports the current view to a number of different supported formats. A dialog window is shown for further configuration options (refer to Section 10.3.12.2);
- Print : Prints the current view. A dialog window is shown for further configuration options (refer to Section 10.3.12.1);

### 30.10.1.1 Drawing Tools

The *Library Browser* provides access to global and project library building blocks. The global library elements are sourced from the database folder: *DlgSILENT Library* → *Dynamic Models* → *Modelica*. The project library elements are sourced from the database folder: <project path> → *Library* → *Dynamic Models*. A category of *Connectors* is provided where input and output signals of the model can be inserted. A category of *Labels* is provided where sender and receiver labels are available, used for simplified routing of signals.

*Library Browser* items can be inserted into the block diagram of a Modelica Model in two ways:

- Activate the *Diagram view* of the *Model canvas* and make sure the diagram is unfrozen;
- Method 1:
  - Drag and drop an item into the block diagram;
- Method 2:
  - Select an item from the library (left-click on item);
  - Deploy the item into the block diagram (left-click on target area of diagram).

Furthermore, the *Drawing Tools* window provides an *Annotations* browser where graphical annotations can be added to either the model's block diagram (using the *Diagram view*) or the model's icon (using the *Icon view*).

### 30.10.1.2 Model canvas

The *Model canvas* supports two views:

- *Diagram view*
- *Icon view*

Using the *Diagram view*, all model components can be inserted into a model based on the available elements of the *Drawing Tools*:

- *Annotation graphics*: Various graphical annotation objects can be used for adding further simulation non-relevant information to the model (e.g. text descriptions, explanatory graphics, etc)
- *Library browser types*: The actual building blocks of the block library can be inserted from the *Library browser*.

Each model can have an icon of its own, which is shown whenever the model is being reused as a component within a parent model (as a *Type Instance* or a *Subsystem*). The icon can be freely drawn using the *Icon view*.

An example of a simple block diagram, comprising a simple PI controller, is shown in Figure 30.10.2.

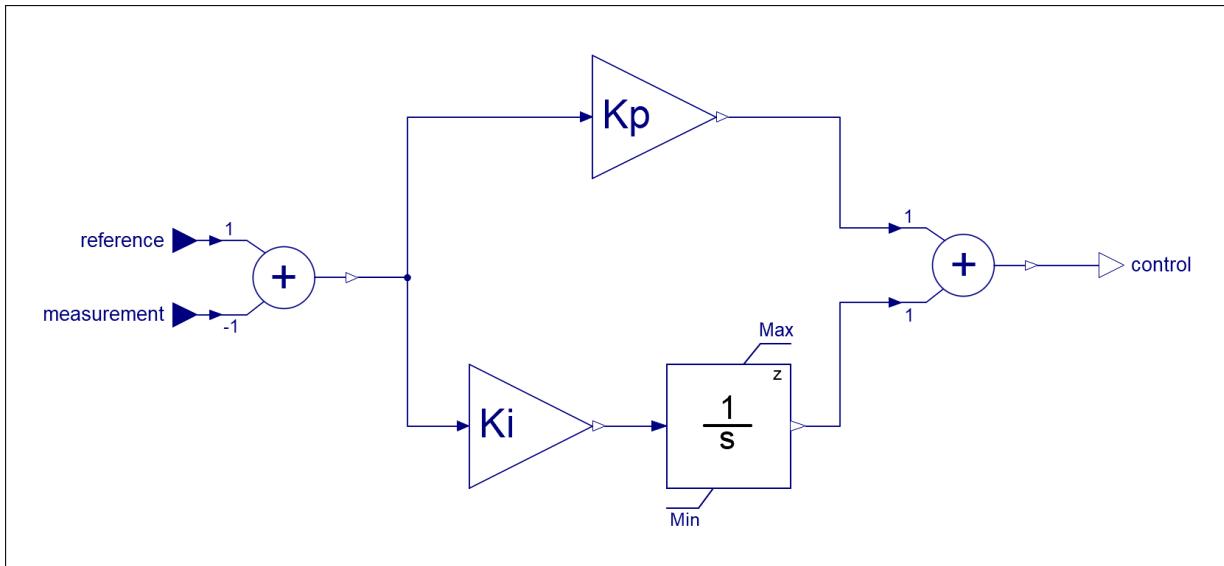


Figure 30.10.2: Example of a diagram of a simple PI controller

### 30.10.1.3 Properties window

Configuration of the deployed building blocks (called *components*) can be directly performed via the *Properties* window (refer to Section 30.10.2.8).

## 30.10.2 Developing Modelica Models

Development of Modelica Models within *PowerFactory* is largely based on the use of the graphical modelling environment (described in Section 30.10.1) which enables users to develop simulation models using hierarchical block diagram designs. This section guides users on how to create and configure such models within *PowerFactory*.

### 30.10.2.1 Inserting a new *Modelica Model Type* diagram

A graphically defined Modelica model can be developed using the block diagram of a *Modelica Model Type*. To create a new *Modelica Model Type* and its corresponding diagram, do the following:

- Activate a *PowerFactory* project where the model shall be created.
- Create a new graphically defined *Modelica Model Type* (refer to Section 10.2.4 for general information on handling new *PowerFactory* graphic windows). One way of creating a new *Modelica Model Type* is provided below:
  - From the *Page Tab* (Section 10.2.5), create a new graphics by clicking the **+** icon;
  - Select *Dynamic model* → *Modelica model type*;
  - Select *Clocked* or *Hybrid (pilot version)* in the drop down list;
  - Click **Execute**;
  - The *Edit dialog* of a newly created *Modelica Model Type* (**mdl**) is shown. Give a relevant object name and then click **OK**;

**Note:** The name of the *Modelica Model Type* object must comply with the Modelica requirements for naming class objects, i.e. compliant with the identifier requirements of Section 2.3.1 of the *Modelica language specification* (version 3.5) e.g. no white spaces, name starts with a letter or underscore (\_)

- A new *Modelica Model Type* is created and an empty block diagram is made available.
- By default, the *Diagram View*  is active. The diagram is by default unfrozen () ;
- The Modelica model can now be developed by inserting *Types* from the *Library browser* or by creating new *Subsystems*, adding *Connectors* and interconnecting them with *Signals*.

### 30.10.2.2 Inserting a library *Type* into the block diagram

To add a *Type instance* to the block diagram:

- Open the *Diagram view*  of the model;
- Unfreeze the diagram ;
- Use the *Library browser* to navigate inside the Global or the Project model library and identify the *Type* to be inserted. Alternatively, use the “Filter as you type” entry field in order to search for specific *Types*;
- Once identified, drag and drop the *Type* into the diagram. The inserted component is called a *Type instance*;
- Move and resize the *Type instance* as needed;
- Connect the inputs and outputs of the *Type instance* with inputs or outputs of other model components, as required;
- Configure the *Type instance* using the *Properties* window;
- To verify your design, click on the *Check Model*  button. Make sure to correct any errors.

**Note:** When the inserted *Type* itself is defined using a block diagram, then double-clicking on a corresponding *Type instance* opens a read-only *Component view* in the same window. Conversely, if the *Type* does not have a block diagram, then double-clicking on a corresponding *Type instance* opens a component dialog of the corresponding instance.

---

### 30.10.2.3 Creating a new project library *Type*

If the existing library *Types* are insufficient for the specific model implementation, a new project library *Type* can be created in one of two ways:

- Define from scratch a completely new *Modelica Model Type* as detailed in Section 30.11;
- Create a copy of an existing library *Type* and apply modifications to it. For this purpose, do the following:
  - Open the Data Manager and navigate to the location of the existing *Modelica Model Type*;
  - Copy and paste this object into a location within the *Project Library* (<project path> → *Library*→ *Dynamic Models*);
  - Apply modifications to the *Type* either graphically (if diagram exists) and/or within the relevant sections of the Edit Dialog of the *Modelica Model Type* object (as detailed in Section 30.11.1).

### 30.10.2.4 Connecting Signals between components

A signal can be drawn between two input/output nodes as follows:

- Position the mouse on the source node (can be an input or an output)

- Hold left click mouse button and drag towards the destination node
- Once the mouse position moves away from the source node, a dashed line is shown as temporary guide line for the chosen signal path. Left mouse click can be released, while still allowing the signal to be routed;
- Click along the required signal path as soon as the signal path changes direction; The procedure can be repeated as often as needed;
- Position the mouse on top of the destination node and left click once more. The signal is now connected between the two nodes.
- To verify your design, click on the *Check Model*  button. Make sure to correct any errors.

---

**Note:** Signals are colour coded: Signals use blue for Real data types, orange for Integer data types and violet for Boolean data types.

---

### 30.10.2.5 Model Connectors

The external inputs and outputs of the model can be created graphically using *Connectors*. They are available from within the *Library Browser*.

To add a model input or output:

- Identify the input/output data type: Real, Integer or Boolean;
- Choose an appropriate connector from the *Connectors* category of the *Library browser*;
- Drag and drop it into the diagram;
- Connectors may be moved anywhere within the model diagram in order to optimally organise the model appearance;
- To verify your design, click on the *Check Model*  button. Make sure to correct any errors.

---

**Note:** Connectors are colour coded: Real data type uses blue, Integer data type uses orange, Boolean data type uses violet.

---

The connectors support scalar or array input and output signals with one of the supported data types.

---

**Note:** Pay special attention to the *Variability* property of input/output signals of top-level models. This is set in the *Variable Declarations page* → *Inputs/Outputs tabs* of the Edit dialog of the top-level model. Set *Variability* of an input signal to “continuous” if the input connects to a signal belonging to a DSL model or a power system element. Set *Variability* of an input signal to “discrete” if the input connects to a discrete signal of another model e.g. Modelica model or other built-in element having discrete signals. When a model is not a top-level model, then the *Variability* of input/output signals may be selected as “inherited”. In these situations, *PowerFactory* tries to automatically identify the *Variability* attribute of input/output signals.

---

### Configuration options of model connectors

The properties of model connectors are accessed by double clicking on the connector and are listed as follows:

- Component name: Editable string field, must contain a Modelica compliant string for a component name. The name of the component, must be unique within the same hierarchical level of the main model. The component name is equivalent to the name of the input or output that the connector represents.
- Model type: not used.
- Size(s): Editable value field, accepts a positive integer number or an integer parameter whose instantiated value is a positive integer. This field refers to the size of the input/output variable that it represents. The same field is also accessible via the edit dialog of the *Modelica Model Type* in which the connector is placed, under the corresponding *Variable Declarations page → Inputs/Outputs tabs → Size(s)* field of the input/output signal that this connector represents.

The index field can contain a (comma-separated) list of indices with length up to the number of dimensions of the connector. Each index can either be an unsigned integer between one and the size of that dimension or the special `:` index indicating all values along this dimension should be kept. Any dimension without an index (because the list is shorter than the number of dimensions of the connector) are kept completely, as though index `:` was specified.

#### 30.10.2.6 Connections

Connections are created by connecting connector components, label components and inputs/outputs of any other Modelica components.

##### Configuring connections using the *Properties* window

Click on a connection in order to show the available options in the *Properties* window. The following connection properties can be customised via the *Properties* window.

- First index: Only shown if the first signal of the connector is a vector. Customises the first signal *A* in the underlying *connect(A,B)* statement. Leave empty to connect the whole first vector signal. To select one index from the first vector signal, set a positive integer *i* between 1 and the size *n* of the first vector signal *u[n]*. The connect statement will identify the vector index *u[i]* for connection. If the second signal *y* of the connector is a scalar, then the equivalent Modelica statement is *connect(u[i],y)*. Otherwise, the *Second index* field is shown, refer to below item. The below table provides an overview and guidance on various possible signal connections.
- Second index: Only shown if the second signal of the connector is a vector. Customises the second signal *B* in the underlying *connect(A,B)* statement. Leave empty to connect the whole second vector signal. To select one index from the second vector signal, set a positive integer *j* between 1 and the size *m* of the second vector signal *y[m]*. If the first signal *u* is a scalar, then the equivalent Modelica statement is *connect(u,y[m])*. Otherwise, the *First index* field is shown, refer to previous item. Consequently, the connector will connect the signal *u[i]* with *y[j]*, equivalent to *connect(u[i],y[j])*. The below table provides an overview and guidance on various possible signal connections.

Purpose	First index	Second index
Connect scalar to scalar	<not shown>	<not shown>
Connect vector to vector	<leave empty>	<leave empty>
Connect scalar to index of vector	<not shown>	<i>j</i>
Connect index of vector to scalar	<i>i</i>	<not shown>
Connect index of vector to index of vector	<i>i</i>	<i>j</i>

Table 30.10.1: Overview and guidance on signal connections using the *First index* and *Second index*

- Line style: Customises the line style of the connection.
- Line colour: Customises the line colour of the connection.
- Line width: Customises the line width of the connection.
- Smooth: Configures the smoothness property of the connection.

### 30.10.2.7 Model labels

A *Label* pair (*SenderId* and a *ReceiverLabel*) is used to create a connection between two *Modelica* components, connection which is not shown in the *Model Diagram*. This functionality is particularly useful in *Model Diagrams* which are cluttered either by containing too many signals, or requiring connections that span over the entire diagram. *Labels* are available in the *Library* under the “Labels” *Modelica Package* (accessible via the *Library Browser*). Two *Labels* are provided within this package: *SenderId* and *ReceiverLabel*.

To create a connection between two *Modelica Components* (e.g. a “Source component” and “Destination component”) using *Labels*, do the following:

- Add to the *Model Diagram* a *SenderId*. Open the *Modelica Component Dialog* of the label (e.g. double-click on it) and set a relevant *Component name* of the label, which easily identifies the signal being transferred (note that the *Component name* must comply with *Modelica* specifications). The component name is automatically added to a list of available *SenderId*s that can be selected by any *Receiver label* existing in the diagram;
- Move the label conveniently close to the “Source component” (e.g. a type instance) and connect the output of the component with the input of the *SenderId* (refer to SubSection 30.10.2.4 for information on how to connect signals between components);
- If the input of the *SenderId* is a vector (i.e. one-dimensional array), then set accordingly the size of the vector in the *Size* field of the *Modelica Component Dialog* of the label. Otherwise, leave the *Size* field empty. Note, that a one-dimensional array with one element (a vector of size 1) requires to set a value “1” in the *Size* field.
- Add to the *Model Diagram* a *ReceiverLabel*.
- Move the label conveniently close to the “Destination component” (e.g. a type instance) and connect the output of the label with the input of the component;
- Click the label once (in order to show the *Properties* window) and select from the drop-down list (named “y”) the linked sender label whose signal is propagated. The drop-down list contains a listing of all *Receiver labels* that are currently existing in the *Model Diagram*.
- To verify your design, click on the *Check Model*  button. Make sure to correct any errors.

### 30.10.2.8 Configuring model components using the *Properties* window

A model component (e.g. *Type Instances*, *Subsystems*) can be configured using the *Properties* window, as follows:

- Using the *Diagram view* of a top-level model, navigate to and select (left click) the model component;
- The *Properties* window is automatically updated with the properties of the selected model component;
- Depending on the specific component characteristics, the component may have various parameters, each with different usage.

- By default, parameter name, start/final value, description and further configuration options are displayed for each entry.

Figure 30.10.3 shows an example of the *Properties* of an integrator *Type Instance*. Component parameters are displayed in this window. If available, a short description of each parameter is provided. The start (default) value can be set directly for each parameter by entering a value in the corresponding input field. A valid value corresponds to (A) a constant value of the same data type as the parameter (e.g. set 1.001 for a Real parameter, set *true* for a Boolean parameter) or (B) a literal parameter of the parent model (the *Modelica Model Type* whose diagram contains the component). For example, w.r.t. Figure 30.10.3, if setting the parameter *yo\_max\_par* of the integrator component to be equal to *Max*, it implies that the parent model (*PI\_Controller*) defines a parameter *Max* which can be assigned to parameter values of various components.

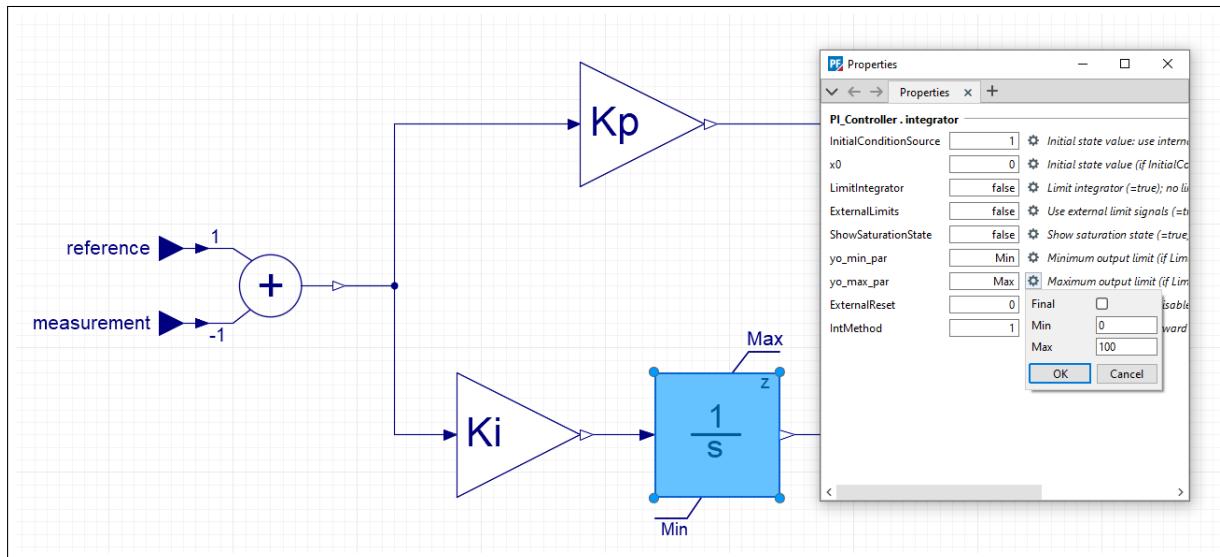


Figure 30.10.3: Properties window of an integrator

The *Properties* window displays the following model component items:

- Parameters
    - **Parameter name:** Displays the name of the parameter.
    - **Value entry field:** Accepts expressions of parameter variability. Sets the default value of the parameter. Examples: “1.32”, “ $\sqrt{A} + \sin(0.5)$ ”, “ $B - C$ ”, where  $A$ ,  $B$  and  $C$  are model parameters.
    - Configuration wheel: Provides access to additional parameter configuration options, as follows:
      - \* **Final:** Checkbox (on/off). If the parameter is intended to be hardcoded (the value never changes and it should not be changeable by model users), then check the *Final* checkbox. Such final parameters are removed from the list of editable parameters of the *Modelica Model (ElmMdl)*. Leave this checkbox unchecked if the parameter shall be shown in the list of editable parameters of the *Modelica Model (ElmMdl)* (refer to Section 30.10.3.2).
      - \* **Min:** Accepts expression of constant variability and supports for scalar parameters. For Real and Integer parameters the minimum value can be set. Minimum values are shown in the *Parameters* page of the *Modelica Model (ElmMdl)*. Examples: “-100.0”, “-230 \*  $\sqrt{2/3} - \sin(0.5)$ ”, “-3/2”.
      - \* **Max:** Accepts expression of constant variability and supports for scalar parameters. For Real and Integer parameters the maximum value can be set. Maximum values are shown in the *Parameters* page of the *Modelica Model (ElmMdl)*. Examples: “100.0”, “230 \*  $\sqrt{2/3} + \sin(0.5)$ ”, “3/2”.
    - Parameter description: Displays the description of the parameter.

### Configuration options of model components

Configuration options of model components are accessed from the model diagram. If the model component is a *Type Instance*, then double click on the component. Alternatively, right click on the component and select “Edit Component” from the drop down menu. If the model component is a *Subsystem*, then right click on the component and select “Edit Component” from the drop down menu. Following options are available:

- Component name: String field which must contain a Modelica compliant string for a component name. The name of the component must be unique within the same hierarchical level of the main model.
- Model type: A reference to the model type being used by the component.
- Size(s): Value field which accepts a positive integer number or an integer parameter whose instantiated value is a positive integer or an integer parameter. This field is only available for the component created by a type *Modelling* method *Clocked*. This field defines the size of this component. Leave empty if a single component is used. If an array of components of size  $n$  is intended to be represented, then set the size to  $n$ . Consequently, the component is multiplied (and represented) internally  $n$  times, creating a vector of components. All input and output variables of the vector of components increase dimension by 1 and size by  $n$  e.g. a scalar input becomes a vector size  $n$ , a vector output size  $m$  becomes a matrix array size  $[m,n]$ .

#### 30.10.2.9 Model Initialisation

In order to initialise the model according to a load flow solution, two initialisation options, “Local” and “External” (see column *Initialisation* in the corresponding sub-pages on page *Variable Declarations*) are set for *input* and *output* variables. These choices are explained as follows:

- External: The *input* or *output* variables are initialised by the connected variables/signals.
- Local: A textual initialisation algorithm can be defined for *inputs* on the *Local Initialisation* page. For *outputs*, the initialisation considers its graphical connections.

To initialise the model, follow these steps:

- Identify which *input* and *output* variables are known from the connected external models. Set these variables to “External”.
- For a clocked Modelica model type, an internal variable of a model component could use an input to initialise. For a hybrid Modelica model type, a state variable of a model component could use an input to initialise. To connect such inputs, add input connector components to the main model and connect. Set the initialisation *Local* or *External* as required.
- Set the remaining *input* and *output* variables to “Local”.
- Write the initialisation algorithm on the *Local Initialisation* page (refer to Section 30.12.2.4). Note that the order of the statements of this algorithm will be executed sequentially.

---

**Note:** The initialisation defined for *Modelica Model Types* on the *Local Initialisation* page must always be a Modelica Algorithm.

---

As an example, consider the in Figure 30.10.4 depicted simple excitation system, which has been graphically implemented using Modelica (*Modelling* method *Clocked*). The input signal  $u$  and output signal  $u_{errs}$  should connect to a machine element’s terminal voltage output and excitation voltage input, respectively. Therefore, the *Initialisation* of these two variables is set to “External”. The component of the first-order filter with gain  $K_e$  requires its internal variable to be initialised using an input. Initialisation of this input depends on the variable  $u_{errs}$ ; therefore, it is set to “Local”. The remaining two input

variables (*vbias* and *usetp*) are also set to “Local”, as their initial values are defined by formulas and not from outside of the model.

The initialisation of the inputs *initOut*, *vbias* and *usetp* depends on the externally initialised variables *u*, *uerrs* and on the parameter *Ke*. Next, the initialisation algorithm is defined under the “Local definition of input start values” on page *Local Initialisation* (see the lower right in Figure 30.10.4)

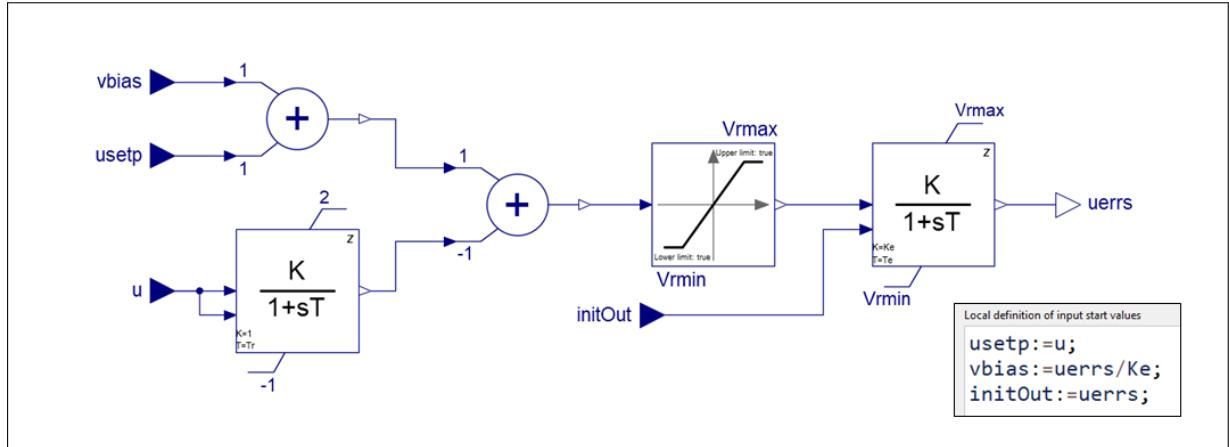


Figure 30.10.4: Initialisation of graphically defined Modelica Model Type

### 30.10.2.10 Creating Subsystems

For graphically defined Modelica models it is possible to define multiple hierarchical levels using the *Diagram view*. For this purpose, the existence of graphically defined *Subsystems* enable the grouping of components into combined blocks and the subsequent insertion of a sub-ordinate hierarchical level w.r.t. the parent component. *Subsystems* may be placed into other *Subsystems* as well, thus providing a means of organising the model using multiple levels.

To create a *Subsystem* in the diagram:

- Right-click on an empty area of the diagram and choose “Create subsystem”.
- Double-click on the *Subsystem* block to open the diagram of the *Subsystem* (*TypMdl*) in the same window;
- Develop the *Subsystem* by adding *Connectors* and library components.
- To verify your design, click on the *Check Model* button. Make sure to correct any errors.

To convert an existing library *Type* into a *Subsystem*:

- From the *Model Diagram*, right-click on the *Type Instance* to be converted.
- From the context menu, select “Convert Library Type Into Subsystem”. The library *Type* is copied within the model structure and the *Type Instance* is now linked to it. The block is now converted into and treated as a *Subsystem*.

To convert an existing *Subsystem* into a library *Type*:

- From the *Model Diagram*, right-click on the *Subsystem* to be converted.
- From the context menu, select “Extract Subsystem as Project Type”. The *Subsystem* is moved out of the model structure and into the project *Library*. The *Subsystem* is converted into a new library *Type* and treated accordingly. The *Type Instance* is now linked to the library *Type*.

**Note:** Copying and pasting an existing *Subsystem* creates a separate copy of the referenced *Modelica Model Type (TypMdl)*. This ensures that *Subsystem* copies are also treated as unique and separate components. Deleting a *Subsystem* also deletes the referenced *Modelica Model Type (TypMdl)*.

A simple example of a top-level model is presented in Figure 30.10.5. The PI controller model shown in Figure 30.10.2 is used here as a *Subsystem*.

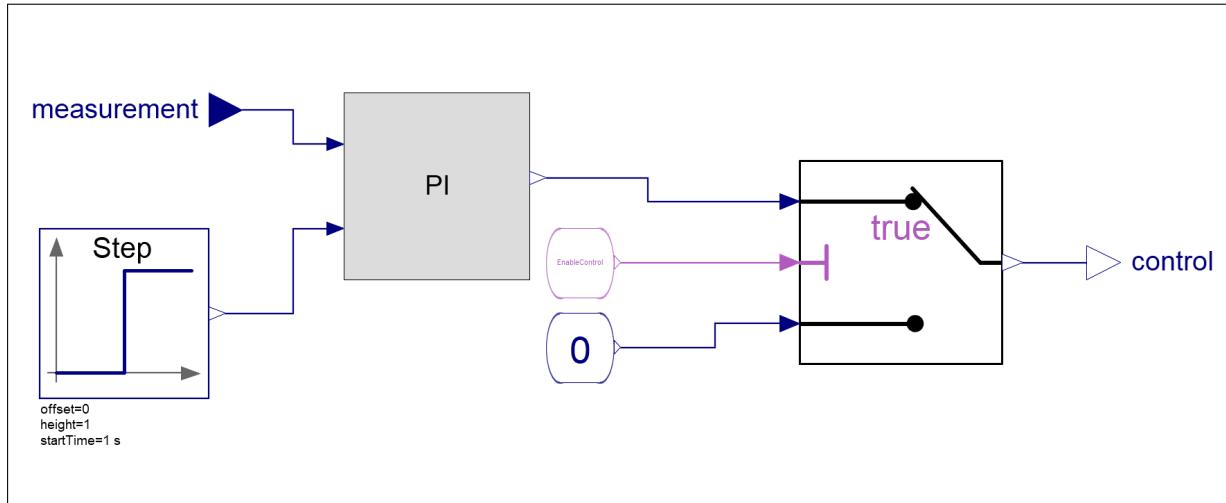


Figure 30.10.5: Example of a simple top-level model including a subsystem

### 30.10.2.11 Navigating through the model hierarchy

Hierarchical models defined by using block diagrams contain several block diagrams, each linked between each other via a child-parent relationship. In this context, a *Top-level diagram* represents the highest level block diagram, which contains all underlying model *Type Instances* or *Subsystems*. The associated *Modelica Model Type (TypMdl)* whose diagram is a top-level diagram is referred to a top-level *Modelica Model Type* (also known as *Top-level model*). A *Top-level model* is never a *Subsystem*. To navigate through the model hierarchy of a Modelica Model, do the following:

- Open the *Diagram view* of the top-level *Modelica Model Type* (this is the highest level model).
- Double click on a specific *Type Instance* or *Subsystem* in order to show their contents. Clicking the former one will show the component (not modifiable) in yellow and the latter one will show the *Subsystem* (modifiable.)
- Continue double-clicking on further contained *Subsystems* (if available) in order to move down into the model hierarchy.
- Click on the icon *Step up to parent component* in order to move up the model hierarchy.

**Note:** The *Step up to parent component* icon is shown as an arrow pointing upwards ↑ on the top right corner of the *Diagram view*, along with the current hierarchical position.

### 30.10.3 Integrating Modelica Models into a *PowerFactory* simulation

Development of Modelica Models within *PowerFactory* is largely based on the use of the graphical modelling environment (described in Section 30.10.1), which enables users to develop simulation models using hierarchical block diagram designs. This section guides users on how to use Modelica models within a *PowerFactory* simulation.

### 30.10.3.1 Integration of Modelica Models within high-level control system designs

The interfacing of a Modelica model follows the typical procedure of integrating a user defined controller within a high-level control system representation, as detailed in Section 30.2 and Section 30.8.1. *Composite Model Frames* and *Composite Models* fully support the interfacing of Modelica models with the *PowerFactory* simulation environment. Consequently, Modelica models can take full advantage of important *PowerFactory* features such as:

- Seamless integration with both RMS- and EMT-simulation functions;
- Integration within *Complete Power Equipment simulation models* (as described in Section 30.1.4).
- Integration with simplified, average and detailed models of power system equipment components (e.g. refer to Section 30.15).
- Model initialisation capabilities based on steady state load flow solution;
- Support of array signals and multiplexing/demultiplexing of arrays within complex system configurations (refer to Section 30.2.7);

### 30.10.3.2 Creation of Modelica Models based on graphically defined model types

**Top-level models** can be instantiated within *PowerFactory* as *Modelica Models*. A *Modelica Model* (*ElmMdl*) element (*PowerFactory* object) represents the instance of a *Top-level model*. It includes all necessary instance specific configuration options: parameterisation and interfacing with other *PowerFactory* models or power system components.

A Modelica dynamic model can be created by following these steps:

- Create a new *Modelica Model* (*ElmMdl*  ) and set the model type (refer to Section 30.8.2)
- Parameterise the model (refer to information further below)
- Integrate it into a high-level control system (i.e. create a corresponding *Composite Model* (*ElmComp*  ) and assign the *Modelica Model* (*ElmMdl*  ) to a relevant slot (see Section 30.2.6)
- Execute the simulation and test the model behaviour

The parameterisation of a graphically defined Modelica Model follows the typical procedure, as existing for *coded Modelica Models*. The only difference may exist in the organisation of parameters into hierarchical levels (shown using a tree-structure). The enforced hierarchy is automatically generated based on the structure of the *Top-level model*.

One hierarchical parameter set for the *Top-level model* displayed in Figure 30.10.5 is shown in Figure 30.10.6. Parameters which are unnecessarily shown in this parameter list must be set to *Final* (refer to Section 30.10.2.8 for more information) in order to be removed.

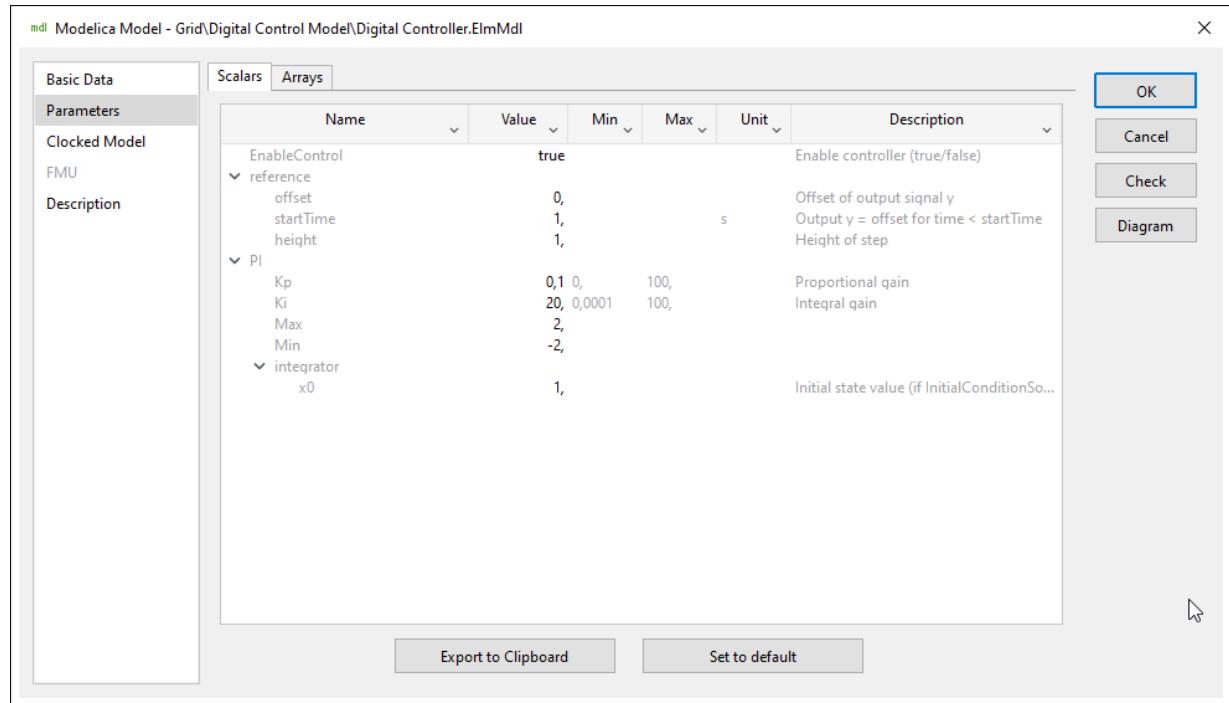


Figure 30.10.6: Example of a parameter set of a Modelica model

### 30.10.3.3 Executing simulations using Modelica Models

*PowerFactory* requires a *C/C++ Compiler* in order to simulate Modelica models. Please refer to Section 30.1.5 for detailed information. Refer to the Chapter 29 for a detailed description of the dynamic simulation within *PowerFactory*. As an overview, the process of performing a dynamic simulation involving Modelica Models is standard:

1. Calculation of initial values, including a load flow calculation (refer to Section 29.3)
2. Definition of result variables (refer to Section 29.5)
3. Definition of simulation events (refer to Section 29.6)
4. Execution of simulation (refer to Section 29.7)
5. Creation of simulation plots (refer to Section 29.8)

### 30.10.3.4 The *Element view* of Modelica Models

*Modelica Models (ElmMdl)* support an *Element view* mode, provided that the referenced Modelica Model Type (*TypMdl*) is graphically defined. This view is especially useful during the stages of model verification and validation, when internal attributes of a model must be quickly cross-checked at a certain simulation time without creating a simulation plot.

Whenever the *Modelica Model* is integrated in a simulation and the simulation is executed up to a certain simulation time (e.g. at initialisation), the *Element view* can be shown as follows:

- Navigate using the Data Manager to the Modelica Model (*ElmMdl*);
- Right-click and choose *Diagrams* → *Show Diagram*;
- A new diagram (read-only) is shown, containing the top-level model diagram. For clear distinction between the *Element View* and other view modes, the background of such diagrams is always shown in a light green color.

- Users can inspect the individual components and navigate through the model hierarchy (if components with diagrams are included). Selecting a component within the diagram will display the currently used parameter values (if any), as well as the values for the input and output signals.

An example of the *Element View* is shown in Figure 30.10.7.

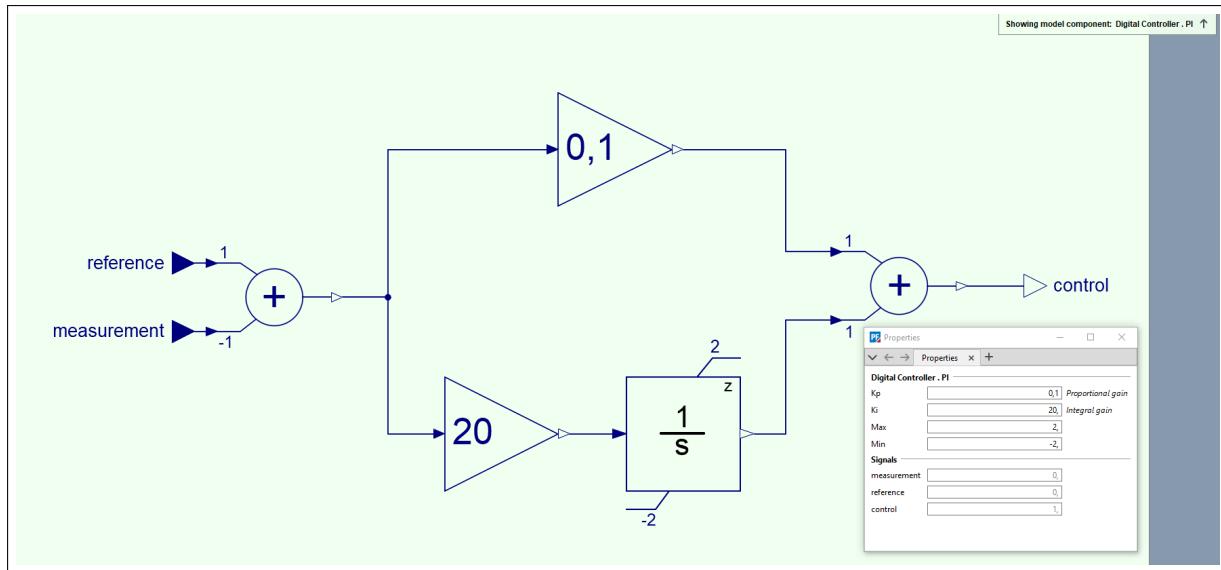


Figure 30.10.7: Example of the *Element View* of a Modelica model

## 30.11 Modelica: Creating Modelica Models using Code

A *Modelica* dynamic model can be created by following these steps:

- Create a new clocked or hybrid *Modelica Model Type* (*TypMdl*  ) (refer to Section 30.12.1).
- Define the dynamic model: set model variables. For a clocked model write the model's Algorithm in *Algorithm* → *Initialisation* and *Algorithm* → *Simulation* sections and for a hybrid model write the model's equations in *Model Equations* → *Initial equations* and *Model Equations* → *Dynamic equations* sections (refer to Section 30.11.1). Verify the model's correctness.
- Create a new *Modelica Model* (*ElmMdl*  ) and set the model type (refer to Section 30.8.2).
- Parameterise the model (refer to Section 30.11.3).
- Integrate it into a high-level control system (i.e. create a corresponding *Composite Model* (*Elm-Comp*  ) and assign the *Modelica Model* (*ElmMdl*  ) to a relevant slot (see Section 30.2.6).
- Execute the simulation and test the model behaviour.

### 30.11.1 Clocked Dynamic model definition

A clocked *Modelica* model can be defined using the *Modelica* language. Upon creation of the clocked *Modelica Model Type* (refer to Section 30.12.1), the structure of the *Modelica* dynamic model is fixed by the customisation options available in the object.

#### 30.11.1.1 Variable declarations

Model variables must be declared within the corresponding fields within the *Variable Declarations* page of the *Modelica Model Type* dialog. For example, a simple digital PI controller could have the following declared variables:

Variable Name	Base Type	Variability	Size(s)	Unit	Description
reference	Real	continuous		p.u.	Reference signal (real, continuous-time)
measurement	Real	continuous		p.u.	Measured signal (real, continuous-time)
x0_ext	Real	continuous			External initial state value

Table 30.11.1: Input Variables of the PI Controller

Variable Name	Base Type	Variability	Size(s)	Unit	Description
control	Real	discrete		p.u.	Control command (real, discrete-time)

Table 30.11.2: Output Variable of the PI Controller

Variable Name	Base Type	Size(s)	Unit	Description
Kp	Real			Proportional gain
Ki	Real		1/s	Integral gain

Table 30.11.3: Parameters of the PI Controller

Variable Name	Base Type	Size(s)	Unit	Description
error	Real			Control error
x	Real			Discrete state variable

Table 30.11.4: Internal Variables of the PI Controller

### 30.11.1.2 *Initialisation* section

The *Modelica Model Type* provides the section *Initialisation* under the page *Algorithm*. The *Initialisation* section can be used to provide model statements that will be executed at the simulation initialisation.

For example, for the simple aforementioned PI controller, a model initialisation could contain the following statements:

```
error:=reference-measurement;  
x:=x0_ext;  
control:=x;
```

---

**Note:** The *Initialisation* section does not specify the sample rate of the discrete model. This is done in the *Modelica Model*.

---

Standard *Modelica* “if-statement” and “for-statement” can be included in the model code. These statements can be nested as well (in the case of *for* only using a single iterator per statement). The “if-statement”, when written inside the *Initialisation* section, can be unbalanced (the true branch needs not have the same number of assignments as the false branch, the false branch is optional).

---

**Note:** In the *Initialisation* section, the following observations are relevant:

- Assignment statements may be used in this section. Assignments are executed sequentially.
  - A *Modelica Model* supports forward initialisation. Other model variables (internal variables and outputs) can be initialised (via assignments) based on the knowledge of the input variables.
  - If an output variable is used in the *Initialisation* section (e.g. an internal variable is assigned), then note that the start value of an output variable is defined in the *Start* field of the *Outputs* tab of the *Variable Declarations* page of the *Modelica Model Type* (refer to Section 30.12.2.3). If no *Start* value is assigned then the start value of the output is automatically set to zero.
  - If the model has any inputs, then each input is initialised from the external model and it is connected to or in the model itself (refer to Section 30.10.2.9).
  - There should not be any assignments for inputs and parameters.
- 

### 30.11.1.3 *Simulation* section

The *Modelica Model Type* provides the section *Simulation* under the page *Algorithm*. The *Simulation* section can be used to provide model statements that will be executed at each sampling period during the simulation. The statements are executed sequentially.

For example, for the simple aforementioned PI controller, the following statements could be contained in the *Simulation* section:

```
error:=reference-measurement;  
x:=previous(x)+error*interval(x);  
control:=Kp*error+Ki*x;
```

---

**Note:** The *Simulation* section does not specify the sample rate of the discrete model. This is done in the *Modelica Model*. If the sample period needs to be used in the model (e.g. for the implementation of an explicit integration algorithm), then the *Modelica* operator “interval()” can be used to retrieve the time period between two consecutive function calls.

---

---

Standard *Modelica* “if-statement” and “for-statement” can be included in the model code. These statements can be nested as well (in the case of *for* only using a single iterator per statement). The “if-statement”, when written inside the *Simulation* section, can be unbalanced (the true branch needs not have the same number of assignments as the false branch, the false branch is optional).

---

**Note:** In the *Simulation* section, the following observations are relevant:

- Assignment statements may be in this section. Assignments are executed sequentially.
  - Each internal variable and output must have at least one assignment statement.
  - There should not be any assignments for inputs or parameters.
- 

If there is an unconnected input and there is a start value, the input is initialised to the start value. However, if there is an unconnected input without a local initialisation and no start value, an initialisation error will be triggered.

#### 30.11.1.4 Model Initialisation

Follow the procedures outlined in Section 30.10.2.9 to define the initialisation according to the load flow solution on the *Local Initialisation* page.

#### 30.11.2 Hybrid Dynamic model definition

A hybrid *Modelica* model can be defined using the *Modelica* language. Upon creation of the hybrid *Modelica Model Type* (refer to Section 30.12.1), the structure of the *Modelica* dynamic model is fixed by the customisation options available in the object.

##### 30.11.2.1 Variable declarations

Model variables must be declared within the corresponding fields on the *Variable Declarations* page of the *Modelica Model Type* dialog. For example, the model can have the same input and parameter variables as in Table 30.11.1 and 30.11.3, respectively, to implement the time-continuous PI controller similar to the clocked PI controller. The output variable has a continuous variability (refer to Table 30.11.5) and can have state and internal variables.

Variable Name	Base Type	Variability	Size(s)	Unit	Description
control	Real	continuous		p.u.	Control command (real, continuous-time)

Table 30.11.5: Output Variable of the PI Controller

Variable Name	Base Type	Size(s)	Unit	Description
x	Real			State variable

Table 30.11.6: State variable of the PI Controller

Variable Name	Base Type	Size(s)	Unit	Description
error	Real			Control error

Table 30.11.7: Internal Variable of the PI Controller

### 30.11.2.2 *Initial equations* section

The *Modelica Model Type* provides the section *Initial equations* under the page *Model Equations*. The *Initialisation* section can be used to provide initialisation or initial equations of model state variables.

For example, for the simple aforementioned PI controller, a model initialisation could contain the following equations:

```
x=x0_ext;
```

Standard *Modelica* “if-equations” can be included in the model code. These equations can be nested as well. When written inside the *Initial equations* section, the “if-equations” should be balanced (the true branch needs to have the same number of equations as the false branch).

---

**Note:** In the *Initialisation* section, the following observations are relevant:

- The state variables can be initialised by algebraic explicit equations (e.g.,  $state\_var = value$ ). Equations with Modelica `der()` operator  $der(var) = 0$  or  $der(var) = val$  are not available yet.
- Sequential Initial equations are not necessary.
- Parameters can be initialised in this section.
- There should not be any initial equations for continuous inputs, internal variables and outputs.
- Any discrete input, output and internal variables (e.g., variables used in when-clauses, Integer and Boolean variables) can be initialised using Modelica `pre()` operator (e.g.,  $pre(var) = value$ ).
- A *Modelica Model* supports forward initialisation. Other model variables (internal variables and outputs) are initialised (via equations) based on the equations defined in the *Dynamic equations*.
- If the model has any inputs then each input is initialised from the external model it is connected to or in the model itself (refer to Section 30.10.2.9).

---

### 30.11.2.3 *Dynamic equations* section

The *Modelica Model Type* provides the section *Dynamic equations* on the page *Model Equations*. The *Dynamic equations* section can be used to provide model equations that will be executed during the simulation. The equations are ordered automatically by *PowerFactory*.

For example, for the simple aforementioned PI controller, the following equations (can be defined in any order) could be contained in the *Dynamic equations* section:

```
error=reference-measurement;  
der(x)=error;  
control=Kp*error+Ki*x;
```

Standard *Modelica* “if-equations” and “when-equations” can be included in the model code. These equations can be nested as well. The “if-equations”, when written inside the *Dynamic equations* section, should be balanced (the true branch needs to have the same number of equations as the false branch).

---

**Note:** In the *Dynamic equations* section, the following observations are relevant:

- Sequential equations are not necessary, i.e., the order of equations is irrelevant.
- The variables should be located on the left in the respective equations used to solve this specific variable.

- Differential equations for state variables are expressed with the Modelica der() operator.
- Multiple equations to solve a variable outside nested “if-equations” and “when-equations” is not allowed.
- There should not be any equations for inputs or parameters.
- The Modelica operator initial() can be used.
- The global built-in variable *time* can be used without declaration.

If there is an unconnected input and there is a start value, the input is initialised to the start value. However, if there is an unconnected input without a local initialisation and no start value, an initialisation error will be triggered.

#### 30.11.2.4 Model Initialisation

Follow the procedures outlined in Section [30.10.2.9](#) to define the initialisation according to the load flow solution on the *Local Initialisation* page.

#### 30.11.3 Dynamic model parameterisation

After instantiating the model (i.e. create a new *Modelica Model*), it can be parameterised within the available *Modelica Model* dialog.

An example of the parameterisation of the previously discussed PI controller is shown in Figure [30.11.1](#).

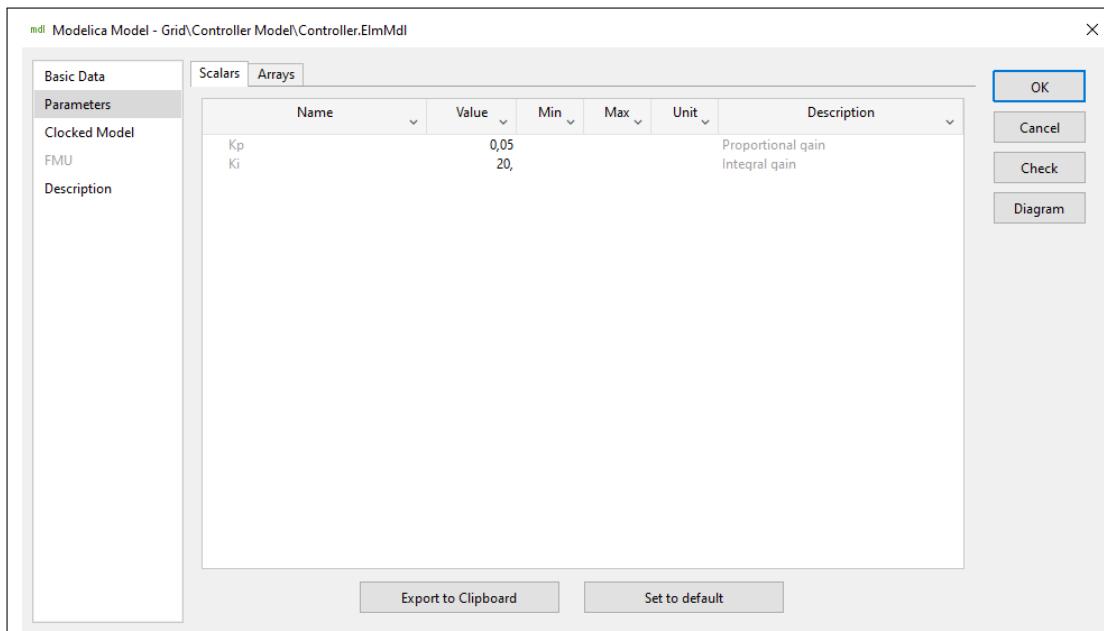


Figure 30.11.1: *Modelica Model* dialog with parameterisation of a PI regulator

## 30.12 Modelica: Overview of the *Modelica Model Type*

The Modelica Model Type supports both textual and graphical model implementations for algorithmic models.

### 30.12.1 Creating a new *Modelica Model Type*

To create a new *Modelica Model Type* (*TypMdl* *mdl*):

- from the main menu:
  - Click *Insert* → *Dynamic Model* → *Modelica Model Type* → *Clocked...* or *Hybrid(pilot version)*...
- from the Data Manager using the **New Object** button:
  - Focus the Data Manager on the library subfolder *Library* → *Dynamic Models*
  - Click *New Object*
  - Select *Modelica Model Type*
- from the *Dynamic Models* folder using the right-click context menu:
  - right-click on the *Dynamic Models* folder on the left side of the Data Manager and select *New* → *Modelica Model Type* → *Clocked...* or *Hybrid(pilot version)*...

### 30.12.2 The Edit Dialog of the *Modelica Model Type*

The edit dialog of the *Modelica Model Type* contains a large set of information about:

- Basic Data - General: Model name, category and model type;
- Basic Data - Advanced: Modelling Method, Initialisation options, FMU details;
- Variable Declarations;
- Local Initialisation;
- Model Equations: text-based model equations can be defined here;
- Algorithm: text-based model algorithm can be defined here;
- Annotation: a graphical representation of the model is provided here;
- Description: text based description can be added here;
- Version: further information regarding versioning.

#### 30.12.2.1 Basic Data page - General

**Name:** The object's name.

**Category:** A user defined model category. A list of standard categories is available. A custom category can be set as well. If using one of the available standard categories then both the *Modelica Model Type* and any referencing *Modelica Model* will be separately categorised within the *Network Model Manager*, according to their designated category. Furthermore, the icon appearance of both the *Modelica Model Type* and any referencing *Modelica Model* is appropriately changed (e.g. a model having the “Automatic Voltage Regulator (avr)” category will change the icon of a *Modelica Model* from *mdl* to *avr*).

**Compiled model:** Check this flag in order to reference a specific compiled model. Otherwise, leave this checkbox unchecked. If a *Compiled model* is selected, then the *Modelica Model Type* can be linked to an externally interfaced model based on either the *FMI Standard* (refer to Section 30.14.1), on the *IEC Interface* (refer to Section 30.14.2) or on the *PowerFactory*-specific interface for Modelica models (such models are obtained by the direct compilation of Modelica Model Type using the **Compile** button). For *FMI Standard* models, an “.fmu” file shall be linked in the additional file path field. For the *IEC Interface* models, a “.dll” file shall be linked in the additional file path field. For the *PowerFactory*-specific interface for Modelica models, a “.pfmu” file shall be linked in the additional file path field.

---

**Note:** The file assigned in the file path of the *Compiled model* can be automatically included in the project export/import process by means of the *PowerFactory* specific \*.pfmx file format, as documented in Section 9.1.5. This allows the external file to be packed into the \*.pfmx project file with all the necessary data and shared as a single package.

---

### 30.12.2.2 Basic Data page - Advanced

*Modelling:* The dropdown menu to choose in between *Clocked* or *Hybrid (pilot version)*.

*Initialisation:* Optionally, set here a *Configuration Script* for flexible model initialisation. *Configuration Scripts* are described in greater detail in Section 30.12.2.11.

*FMU:* an information pane is shown that provides logger settings of the loaded Functional Mock-Up Unit (FMU). This pane is active only if *Compiled model* is ticked and a valid FMU path is provided.

### 30.12.2.3 Variable Declarations page

The *Modelica Model Type* supports the following variables:

- Inputs
- Outputs
- State variables
- Parameters
- Internal Variables
- Enable input

These variables are described in more detail below.

*Show additional columns:* The following additional options are available. Selecting these options on the corresponding page also selects the same option on all other pages, where the option is available.

- *Minimum/maximum values:* Check this box to display the variable modifier fields “Min” and “Max”.
- *Default/start values:* Check this box to display the variable modifier fields “Default/Start”.

**Inputs:** A tabular entry list of the model’s input variables is available. The input variable name, base type, variability, initialisation, size, unit, description can be entered.

*Input variables* have the following properties:

- Name - the variable’s name, according to Modelica variable naming requirements.
- Base Type - The base type can be: Real, Integer or Boolean.
- Variability - Choices are: continuous, discrete or inherited.
- Initialisation - Choices are: External, Local.
- Size(s) - Set a size for a corresponding variable:
  - Inputs can be scalars or one-dimensional arrays (vectors).
  - Two- and multi-dimensional input arrays are not allowed.
  - Leave field empty if a scalar is declared.
  - One-dimensional input arrays (vectors) are declared by setting in this field a strictly positive integer number or an integer parameter e.g. “3”, “size\_in”, where *size\_in* is an integer parameter whose instantiated value must be strictly positive.

- The format is according to Modelica array specification e.g. “1”, “3”, “size\_in”, “[1]”, “[10]”, “[size\_in]”.
- Note, by entering “1” in the *Size* field, *PowerFactory* interprets it as being a vector of size 1.
- Start - a value that is initially (at simulation start) assigned to the variable.
- Min - a minimum value allowed for the variable.
- Max - a maximum value allowed for the variable.
- Unit - a string that defines the unit of this variable.
- Description - a string that defines the description of this variable.
- Condition - a string that defines a user defined condition applicable to this input. The *Condition* must be a Boolean scalar expression of parameter variability e.g. “enable\_input” or “enable\_input == true”, where *enable\_input* is a Boolean parameter; “ $A > B$ ”, where *A* and *B* are Real parameters. If the condition evaluates to true, then the input is enabled; if the condition evaluates to false, then the input is disabled. The enabled/disabled behaviour is applied graphically as well, via the corresponding connector, which is shown/hidden depending on this *Condition*.

**Outputs:** A tabular entry list of the model’s output variables is available. The output variable’s name, base type, variability, initialisation, size, unit, description can be entered.

*Output variables* have the following properties:

- Name - the variable’s name, according to Modelica variable naming requirements.
- Base Type - The base type can be: Real, Integer or Boolean.
- Variability - Continuous, discrete or inherited. The option continuous is available for *Modelling* method *Hybrid (pilot version)*.
- Initialisation - External or Local.
- Size(s) - Sets the size for a corresponding variable.
  - Outputs can be scalars or one-dimensional arrays (vectors).
  - Two- and multi-dimensional output arrays are not allowed.
  - Leave field empty if a scalar is declared.
  - One-dimensional output arrays (vectors) are declared by setting in this field a strictly positive integer number or an integer parameter e.g. “3”, “size\_out”, where *size\_out* is an integer parameter whose instantiated value must be strictly positive.
  - The format is according to Modelica array specification e.g. “1”, “3”, “size\_out”, “[1]”, “[10]”, “[size\_out]”.
  - Note, by entering “1” in the *Size* field, *PowerFactory* interprets it as being a vector of size 1.
- Start - a value that is initially (at simulation start) assigned to the variable.
- Min - a minimum value allowed for the variable.
- Max - a maximum value allowed for the variable.
- Unit - a string that defines the unit of this variable.
- Description - a string that defines the description of this variable.
- Condition - a string that defines a user defined condition applicable to this input. The *Condition* must be a Boolean scalar expression of parameter variability e.g. “enable\_output” or “enable\_output == true”, where *enable\_output* is a Boolean parameter; “ $A > B$ ”, where *A* and *B* are Real parameters. If the condition evaluates to true, then the output is enabled; if the condition evaluates to false, then the output is disabled. The enabled/disabled behaviour is applied graphically as well, via the corresponding connector, which is shown/hidden depending on this *Condition*.

*Show additional columns:* The following additional options are available. Selecting these options on this page also selects the same option on all other pages (e.g., *Inputs*).

- *Minimum/maximum values:* activate this checkbox in order to show the variable modifier fields “Min” and “Max”.
- *Default/start values:* activate this checkbox in order to show the variable modifier fields “Default-/Start”.

**State variables:** A tabular entry list of the model's state variables is available. This entry is available the *Modelling* method *Hybrid (pilot version)*. The state variable's name, size, unit, description can be entered.

*State variables* have the following properties:

- Name - the variable's name, according to Modelica variable naming requirements.
- Base Type - The base type can be: Real.
- Size(s) - Sets the size for the corresponding variable.
  - State variables can be scalars or one-dimensional arrays (vectors).
  - Two- and multi-dimensional output arrays are not allowed.
  - Leave field empty if a scalar is declared.
  - One-dimensional output arrays (vectors) are declared by setting in this field a strictly positive integer number or an integer parameter e.g. “3”, “size\_state”, where *size\_state* is an integer parameter whose instantiated value must be strictly positive.
  - The format is according to Modelica array specification e.g. “[1]”, “[10]”, “[size\_out]”.
  - Note, by entering “1” in the *Size* field, *PowerFactory* interprets it as being a vector of size 1.
- Start - a value that is initially (at simulation start) assigned to the variable (to display this field, use the corresponding checkbox in the “Show additional columns” pane).
- Min - a minimum value allowed for the variable.
- Max - a maximum value allowed for the variable.
- Unit - a string that defines the unit of this variable.
- Description - a string that defines the description of this variable.

**Parameters:** A tabular entry list of the model's parameters is available. The parameter name, base type, size, minimum, maximum, unit and description can be entered.

*Parameters* have the following properties:

- Name - the parameter's name, according to Modelica variable naming requirements.
- Base Type - The base type can be: Real, Integer or Boolean.
- Size(s) - Set a size for this variable. Accepts an integer value or an integer parameter.
  - Parameters can be scalars, one- or two-dimensional arrays.
  - Multi-dimensional (dimension greater than 2) parameter arrays are not allowed.
  - Leave field empty if a scalar is declared.
  - One-dimensional parameter arrays (vectors) are declared by setting in this field a strictly positive integer number or an integer parameter e.g. “3”, “size\_par”, where *size\_par* is an integer parameter whose instantiated value must be strictly positive.
  - Two-dimensional parameter arrays (matrices) are declared by using in this field two strictly positive integer numbers or integer parameters e.g. “[3,3]”, “[1,size\_par]”, “[size\_par\_row,size\_par\_col]” where *size\_par*, *size\_par\_row*, *size\_par\_col* are integer parameters whose instantiated values must be strictly positive.

- The format is according to Modelica array specification e.g. “1”, “3”, “size\_par”, “[1]”, “[10]”, “[size\_par]”, “[3,3]”, “[1,size\_par]”.
- Note, by entering “1” in the *Size* field, *PowerFactory* interprets it as being a vector of size 1. This results in having this parameter interpreted as a vector and not as a scalar. Furthermore, the parameter will be configurable using a referenced *IntMat* object in the *Arrays* tab of the *Modelica Model* Edit dialog and not in the *Scalar parameters* tab.
- Default - The default value of the parameter. This value can be overwritten in the *Modelica Model*. The format to define the default of array parameters is according to Modelica array specification notation, i.e., the { } notation. For example, the default of a Real parameter with size [3] is {1.1,2.1,3.1} and the default of an Integer parameter with size [2,3] is {{1, 2, 3}, {5, 6, 7}}.
- Min - a minimum value allowed for the parameter.
- Max - a maximum value allowed for the parameter .
- Unit - a string that defines the unit of this parameter.
- Description - a string that defines the description of this parameter.

**Internal variables:** A tabular input list of the model’s internal variables is available. The internal variable name, base type, size, unit and description can be entered. This list is only relevant for models created using code (refer to Section [30.11](#)).

*Internal variables* have the following properties:

- Name - the variable’s name, according to Modelica variable naming requirements.
- Base Type - The base type can be: Real, Integer or Boolean.
- Size(s) - Set a size for this variable.
  - Internal variables can be scalars, one- or two-dimensional arrays.
  - Multi-dimensional (dimension greater than 2) internal variable arrays are not allowed.
  - Leave field empty if a scalar is declared.
  - One-dimensional internal variable arrays (vectors) are declared by setting in this field a strictly positive integer number or an integer parameter e.g. “3”, “size\_var”, where *size\_var* is an integer parameter whose instantiated value must be strictly positive.
  - Two-dimensional internal variable arrays (matrices) are declared by using in this field two strictly positive integer numbers or integer parameters e.g. “[3,3]”, “[1,size\_var]”, “[size\_var\_row,size\_var\_col]” where *size\_var*, *size\_var\_row*, *size\_var\_col* are integer parameters whose instantiated values must be strictly positive.
  - The format is according to Modelica array specification e.g. “1”, “3”, “size\_var”, “[1]”, “[10]”, “[size\_var]”, “[3,3]”, “[1,size\_var]”.
  - Note, by entering “1” in the *Size* field, *PowerFactory* interprets it as being a vector of size 1.
- Min - a minimum value allowed for the variable.
- Max - a maximum value allowed for the variable.
- Unit - a text string that defines the unit of this variable.
- Description - a text string that defines the description of this variable.

**Models:** A tabular entry list of the model’s dependencies with other *Modelica Model Types*. This list is automatically configured when defining *Modelica Models* using block diagrams.

**Labels:** A tabular entry list of the model’s internal labels, used for simplified signal routing. This list is automatically configured and updated when defining *Modelica Models* using block diagrams.

**Enable:** This subpage is to add a boolean enable input. A boolean enable input allows the execution of the block conditionally during simulation while the input signal is true. This entry is available for *Modelling* method *Clocked*.

*Add Boolean enable input:* Select this option to add a boolean enable input. It is a scalar. Its name, variability, description and start can be entered.

- Name - the inputs's name, according to Modelica variable naming requirements.
- Variability - Choices are: discrete or inherited.
- Description - a text string that defines the description of this input.
- Start - a value that is initially (at simulation start) assigned to the input.

#### 30.12.2.4 Local Initialisation page

A text field is provided under “Local definition of input start values” to define the local initialisation algorithm. The input variables with initialisation choice: Local on page *Variable Declarations* can be initialised by writing *Modelica* compliant algorithmic code based on *inputs* and *outputs* with initialisation choice: “External” and *parameters*. At least one of the *inputs* must have the initialisation choice: “Local” to make this page active.

#### 30.12.2.5 Model Equations page

This page is available for *Modelling* method *Hybrid (pilot version)*.

*Initial equations:* A text field for the initial equations is provided. *Modelica* compliant code can be programmed here to initialise the model. This text field is only relevant for the models (*Modelling* method *Hybrid (pilot version)*) created using code (refer to Section 30.11).

*Dynamic equations:* A text field for the equations to be executed during simulation. *Modelica* compliant code can be programmed here to define the model behaviour during simulation. This text field is only relevant for the models (*Modelling* method *Hybrid (pilot version)*) created using code (refer to Section 30.11).

*Connections:* If the model is graphically defined, then the graphical connections between the various model components are specified here.

#### 30.12.2.6 Algorithm page

This page is available for *Modelling* method *Clocked*.

*Initialisation:* A text field for the initialisation statements is provided. *Modelica* compliant code can be programmed here to initialise the model. This text field is only relevant for the models created using code (refer to Section 30.11).

*Simulation:* A text field for the code to be executed during simulation. *Modelica* compliant code can be programmed here to define the model behaviour during simulation. This text field is only relevant for the models created using code (refer to Section 30.11).

*Connections:* If the model is graphically defined, then the graphical connections between the various model components are specified here.

#### 30.12.2.7 Annotation page

*Annotation:* A text field containing the *Modelica* Annotation code, relevant for various parts of the graphical modelling environment.

### 30.12.2.8 Description page

A general purpose *Description* page is available for each *Modelica Model Type*.

### 30.12.2.9 Version page

A *Version* page is available for each *Modelica Model Type*.

*Version*: A *Version* text field

*Author*: An *Author* text field

*Copyright*: A *Copyright* text field

*Checksum*: An internally calculated model checksum.

*Change log*: A multi-line text field used to log model changes.

### 30.12.2.10 Dialog buttons

**Check**: This button is used to check the model validity, including syntax checks of the model code.

**Diagram**: This button is used to show the *Modelica Model Type* diagram (if existing).

**Compile**: This button is used to compile the model into a *PowerFactory* compliant binary file. The generated binary uses a custom format and is exported as a “.pfmu” file. The compiled model file can only be used within *PowerFactory* and it is typically employed for improving the simulation performance or for sharing the model between *PowerFactory* users.

**FMU Export**: This button is used to export the model into a Functional Mock-Up Unit (FMU), compliant with FMI version 2.0 and 3.0 (refer to Section 30.14.1.3 for more details).

**Pack**: If this model uses any references to external *Types* then use the **Pack** command to copy the external *Types* into a dedicated folder located inside the object. It automatically reassigns the existing references to these internal *Types*. The function allows simplified export and sharing of the Modelica models across project boundaries or outside the *PowerFactory* database.

---

**Note**: The *Pack* command dialog provides the option *Include objects from the DlgSILENT Library*. If this option is enabled, all referenced *DlgSILENT Library* objects will be packed, and their references will be updated accordingly to ensure consistency.

---

**Blackbox**: This button compiles the model into a *PowerFactory* compliant binary file by hiding the internal variables and parameters whose default values depend on other parameters, i.e., calculated parameters. The generated binary file uses a custom format and is exported as a “.pfmu” file. This functionality is available for the *Modelling* method *Clocked*. The button allows to export and share the Modelica models across project boundaries or outside the *PowerFactory* database by hiding internal variables and calculated parameters. Users who receive the black box model will not be able to observe the hidden variables in the type, nor will they be able to plot these variables after simulation.

### 30.12.2.11 Configuration scripts for initialisation

A DPL *Configuration Script* can be added to any **Top-level** Modelica model. When used, it performs a run-time update of certain Modelica model parameters. The script is automatically executed by the

*Calculation of Initial Conditions (ComInc)* command after the successful execution of the load flow calculation and prior to the actual model initialisation.

---

**Note:** Only Real, Integer and Boolean scalar parameters of a Modelica model can be modified by a Configuration Script. For example, given the model parameters displayed in Figure 30.10.6, a configuration script could modify all the *Scalars*.

---

The functionality of the *Configuration Script* is two-fold:

- it provides a flexible way of configuring Modelica model parameters;
- it allows Modelica models to issue customised output window messages at initialisation i.e. via DPL output window methods such as *Info()*.

The Configuration Script is linked within a Modelica model as shown in Figure 30.12.1. It can be set up by following these steps:

- Creating a DPL script within the *Modelica Model Type (TypMdl)* object. In the script itself, users may define various *Result* variables to overwrite Modelica model parameters.
- To define result variables which are not *Top-level* (e.g., PI.Kp in Figure 30.10.6) define those using the dot notation (e.g., PI.Kp) in the column *Dynamic model parameter* on *Results*. Set a name without the dot (e.g., PI\_Kp) in the column *Name*. Use the defined name, i.e., PI\_Kp, in the script code from the *Name*. For *Top-level* parameters, use the same name as the Modelica model in the column *Name* and leave the entry on *Dynamic model parameter* empty.
- Those results whose parameter names match Modelica model parameters will be considered for overwriting the values already assigned to the *Modelica Model (ElmMdl)* object. That is, during the calculation of initial conditions, the pre-existent values of the Modelica model parameters will be overwritten by the *Configuration Script*.
- Optional: the DPL script can be programmed to contain both *Input parameters* and External objects.
- Referencing the DPL script to the *Modelica Model Type (TypMdl)* object, using the corresponding field “Configuration Script” within *Basic Options* → *Advanced* tab of the corresponding Edit dialog.
- Creating a new *Modelica Model (ElmMdl)* within the *Network Model* folder of the project (e.g. typically within one of the “Grid” folders). This action will automatically create a corresponding DPL script that references the previously defined *Configuration Script*.
- Optional: It is possible to assign instance-dependent attributes to the *Configuration Script*. Each *Modelica Model* contains a DPL script that references the remote script of the *Modelica Model Type*. Those *Input parameters* and *External objects* which were previously defined within the remote script (refer to the optional step above), will be available for editing for each Modelica model that shares the same *Modelica Model Type*. This provides increased flexibility, allowing each model to be differently parameterised. For example, it is possible to link different external objects to each *Configuration Script*.

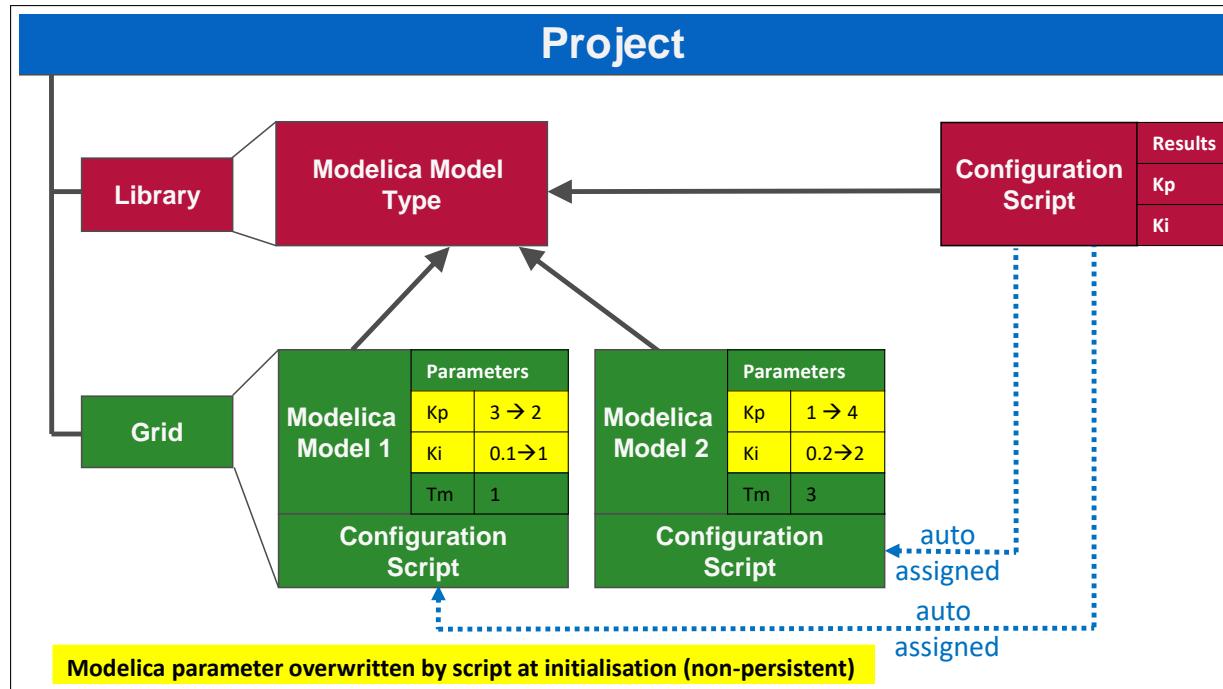


Figure 30.12.1: Configuration Script functionality and inter-operation

**Notes:**

- The scripting language used for the Configuration Script is *DlgSILENT Programming Language* (DPL). Refer to Chapter 23 for a comprehensive description of DPL. Further documentation is available within the *DPL Reference*, which provides a full description of available functions. The *DPL Reference* can be accessed in the *PowerFactory* program from the main menu at *Help* → *Scripting References* → *DPL*. Functions that can be used within the *Configuration Script* are marked appropriately in the *DPL Reference* documentation with a star (\*) symbol (e.g. *GetFrom-StudyCase\**). There are functions (without a star (\*) symbol) that are not available; e.g. the execution of a load flow (*ComLdf*) command is not allowed.
- To modify a *Real* parameter of a Modelica model, the DPL result variable must be set to *double*;
- To modify an *Integer* parameter of a Modelica model, the DPL result variable must be set to *int*;
- To modify a *Boolean* parameter of a Modelica model, the DPL result variable must be set to *int*; an automatic type casting between integer (DPL) and Boolean (Modelica) occurs.
- Parametric changes made by the *Configuration Script* are non-persistent. When the simulation is reset (e.g. by clicking the button *Reset Calculation*), the values of the modified Modelica model parameters will be set back to their initial values - meaning that the *Configuration Script* does not permanently modify project data.
- The values of modified Modelica model parameters can be displayed using the Data Manager. After the successful execution of the *Calculation of Initial Conditions*, do the following:
  - Using the *Data Manager* or the *Network Model Manager*, navigate to and identify the *Modelica Model* that contains the modified parameter;
  - Left-click the Modelica model and activate the *Detail Mode* from the toolbar;
  - Left-click on the *Flexible Data* tab and then on the *Variable Selection* button from the toolbar. The *Variable Selection* window will be shown;
  - Use the filter *Group* to get the *Results* and the filter *Calculation* to get *Simulation RMS* or *Simulation EMT* (depending on the case).
  - Select the modified parameter(s) in order to display them;
  - Click *OK* and observe the parameter values being used (overwritten by the *Configuration Script*).

- In Figure 30.12.1 above, an example is provided for the operation of a script. In this case, the *Configuration Script* declares two result variables  $K_p$  and  $K_i$ . The Block definition contains three parameters:  $K_p$ ,  $K_i$  and  $T_m$ . This means that the script will overwrite only two model parameters:  $K_p$  and  $K_i$ . In the figure, exemplary values are provided: for “Modelica Model 1” parameter  $K_p$  has an initial value of 3 and the script will overwrite it with the value 2. Similarly, parameter  $K_i$  has an initial value of 0.1 and the script will overwrite it with the value 1. The overwritten values of the parameters will be used for all subsequent calculations (e.g. initial conditions and simulation). The script allows assignment of instance-dependent attributes. It implies that, depending on the script’s code, each *Modelica Model* may be configured with individual values for the two parameters. Consequently, the parameters  $K_p$  and  $K_i$  of “Modelica Model 2” will have been assigned different values. Note that at the end of the simulation all modified parameters are automatically set back to their initial values.

## 30.13 Modelica: Overview of the *Modelica Model* (ElmMdl)

The *Modelica Model* (*ElmMdl* object) can be configured using the corresponding edit dialog. The relevant configuration options are detailed within this section.

### **Dialog Buttons**

- OK*: Close dialog and save changes.
- Cancel*: Close dialog and do not save changes.
- Check*: Checks if the model is correctly configured. If there are any issues, a detailed error message is written to the output window.
- Diagram*: This button is used to show the *Modelica Model* diagram.

### **Basic Data page**

Following configuration options are available:

- Name*: Set the name of this object.
- Type*: Reference to the Modelica Model Type (*TypMdl*) being instantiated; the Modelica Model Type may represent a dynamic model implemented using native Modelica Language, or an externally-interfaced model (refer to Section 30.14 for more information on external models).
- Configuration script*: Reference to the Configuration Script for initialisation that is used for this model (if any).
- Out of Service*: Tick this box to set the model to “Out of Service”.
- A-stable integration algorithm (only for RMS Simulation)*: tick this option to use the A-stable integration algorithm. This option is available for referenced Modelica Model Type (*TypMdl*) has *Modelling method Hybrid (pilot version)*.

### **Parameters page**

The model parameters can be defined in the *Parameters* page. Scalar parameters are defined in the *Scalars* tab. If the model contains sub-blocks, then a hierarchical parameter structure can be presented, depending on the model’s internal realisation.

### **Additional Dialog Buttons**

- Export to Clipboard*: Copies the entire parameter list to the clipboard (applicable only to scalar parameters)
- Set to default*: Overwrites existing scalar parameters with the default parameter values

Array parameters are defined in the tab *Arrays*. PF allows one- and two-dimensional array parameters. The array parameters must be defined using *IntMat* objects assigned in each corresponding field of an array.

#### *Vector parameters (one-dimensional arrays)*

A vector  $x[n]$  (of size  $n$ ) is treated by default as a column vector, representing a matrix with  $n$  rows and 1 column. For instantiating the vector parameter  $x[n]$ , then an *IntMat* object with  $n$  rows and 1 column must be created and assigned to the corresponding field.

A row vector  $x[1,n]$  (of size  $n$ ) is treated as a matrix with 1 row and  $n$  columns. For instantiating a row vector parameter  $x[1,n]$ , then an *IntMat* object with 1 row and  $n$  columns must be created and assigned to the corresponding field.

#### *Matrix parameters (two-dimensional arrays)*

A matrix  $x[n,m]$  ( $n$  rows and  $m$  columns, of size  $n * m$ ) can be defined by creating an *IntMat* object with  $n$  rows and  $m$  columns and subsequently assigning it to the corresponding field.

For Real arrays, the entry fields represent the actual values of the variable. For Integer arrays, the entry fields are mapped using the math function *round()*. For Boolean arrays, an entry value of 0 is mapped to *false*, any other value is mapped to *true*.

#### **Clocked Model page**

This page is only active for reference to the Modelica Model Type (*TypMdl*), which has the *Modelling* method *Clocked*.

#### *Sampling options* pane

The *Modelica Model* code within the *Simulation* section is called during simulation with a rate/period defined by these parameters:

- *Sampling period*: set the clock of the model by means of a Sampling period parameter
- *Sampling rate*: set the clock of the model by means of a Sampling rate parameter
- *According to integration step size*: The clock of the model is automatically chosen at runtime. The sampling period is equal to the chosen simulation step size, as defined in the *Calculation of initial conditions* → *Step Size* page. This option is only applicable if the automatic step size adaptation (flag *Calculation of initial conditions* → *Basic Options*→ *Automatic step size adaptation*) is unticked.
- *Additional delay of outputs by one sampling period*: Set this option to add a sampling period delay to the model outputs.

---

**Note:** If an FMU model based on FMI for Co-simulation is configured, then this option allows the inclusion/removal of a sampling period delay applied to the model outputs.

---

**Note:** The *According to integration step size* option is highly useful for models dedicated to post-processing tasks (e.g. identification of minima and maxima of a large set of variables at each simulation time step, RMS voltage computation according to various algorithms for later use in a technical report, etc.) or for replicating the behaviour of legacy simulation environments based purely on procedural models. However, whenever such models do influence the power system simulation trajectory (e.g. models controlling power equipment), the simulation results obtained with such models are inevitably simulation time step dependent. Therefore, special attention should be paid when using such models in simulations.

---

#### **FMU page**

If a Functional Mock-up Interface (FMI) based model (Functional Mock-up Unit - FMU) is configured in the Modelica Model Type (refer to Section 30.14.1), then this page is active.

*FMU Type* - details which FMI standard is supported by the FMU. If the FMU supports multiple choices, then the user may switch between them.

- *Model Exchange* - FMU is based on the “FMI for Model Exchange” standard
- *Co-Simulation* - FMU is based on the “FMI for Co-Simulation” standard

#### *Sampling options* pane

This option is only available for *FMI for Co-Simulation* based FMUs. The *FMU* simulation code is called during simulation with a rate/period defined by these parameters:

- *Sampling period*: set the clock of the model by means of a Sampling period parameter
- *Sampling rate*: set the clock of the model by means of a Sampling rate parameter
- *According to integration step size*: - The clock of the model is automatically chosen at runtime. The sampling period is equal to the chosen simulation step size, as defined in the *Calculation of initial conditions* → *Step Size* page. This option is only applicable if the automatic step size adaptation (flag *Calculation of initial conditions* → *Basic Options*→ *Automatic step size adaptation*) is unticked.

---

**Note:** The *According to integration step size* option is highly useful for models dedicated to post-processing tasks (e.g. identification of minima and maxima of a large set of variables at each simulation time step, RMS voltage computation according to various algorithms for later use in a technical report, etc.) or for replicating the behaviour of legacy simulation environments based purely on procedural models. However, whenever such models do influence the power system simulation trajectory (e.g. models controlling power equipment), the simulation results obtained with such models are inevitably simulation time step dependent. Therefore, special attention should be paid when using such models in simulations.

---

#### *Description page*

A standard object description page is provided in order to configure general object settings.

## 30.14 Interfaces for Dynamic Models

### 30.14.1 Functional Mock-Up Interface (version 2.0)

The [Functional Mock-up Interface](#) (FMI) is a free standard which facilitates the exchange of simulation models among a large number of simulation programs. Currently, more than [190 tools](#) actively support FMI. The benefits of a widely used and well established standard are readily available in *PowerFactory* by allowing the import and export of *Functional Mock-up Units* (FMUs) .



Figure 30.14.1: Functional Mock-Up Interface (FMI) standard, [Modelica Association](#)

### 30.14.1.1 Importing FMUs into *PowerFactory*

An overview of the proposed *FMU Import* tool-chain is shown in Figure 30.14.2, in which one or multiple Functional Mock-Up Units (FMUs) can be integrated within a power system simulation executed in *PowerFactory*.

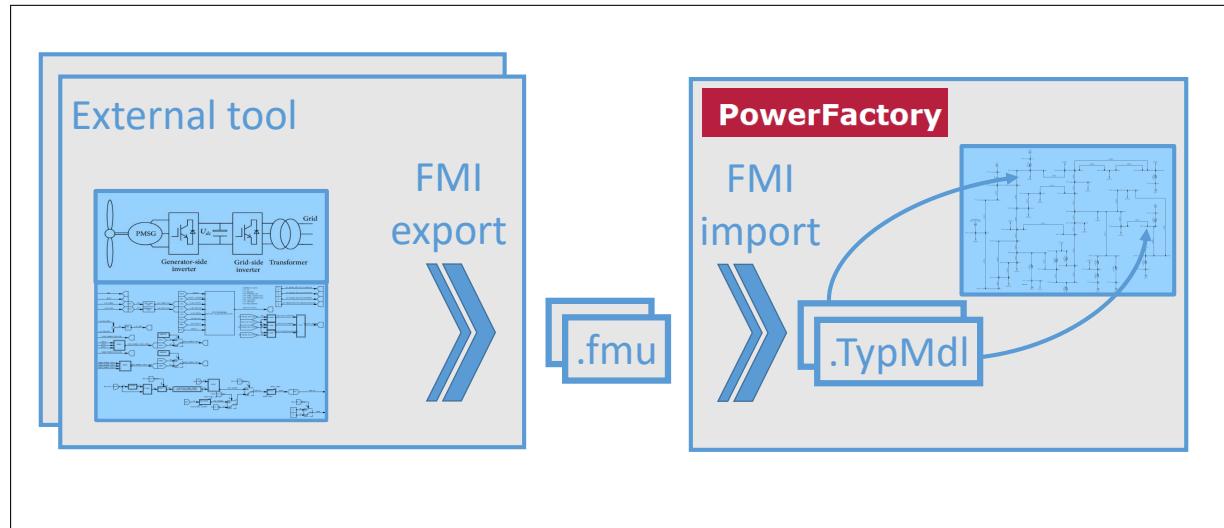


Figure 30.14.2: Overview of integrating *Functional Mock-Up Units* (FMUs) into *PowerFactory*

For example, the state-of-the-art control systems dynamic modelling environment *Matlab® Simulink®* allows the export of a simulation model as a [tool coupling FMU](#) or as a [standalone FMU](#) compliant with FMI standard (e.g. FMI version 2.0, FMI for co-simulation). These FMUs can be imported into *PowerFactory* and used within a large scale RMS Simulation or a detailed EMT Simulation. In particular, the integration of detailed simulation models (e.g. black-box, firmware code based) now becomes a straightforward process of exporting an FMU from one tool, commonly used by model developers of equipment, and importing it into *PowerFactory*, commonly used for executing power systems simulations.

*PowerFactory* supports the following FMI Import components:

- FMI version 2.0 for Model Exchange (FMU model import)
- FMI version 2.0 for Co-simulation (FMU model import)
- FMI version 3.0 for Model Exchange (FMU model import)
- FMI version 3.0 for Co-simulation (FMU model import)

The integration of the FMUs is realised within *PowerFactory* by employing the object classes for *Modelica* compliant dynamic models i.e. *Modelica Model Type* (*TypMdl* `mdl`) and *Modelica Model* (*ElmMdl* `mdl`). A fully automated process of importing the FMU is supported, thus allowing the power systems engineer to focus on the analysis itself and to not waste time on the cumbersome integration of external models into a power system model.

Not all FMI standard capabilities are supported by the FMI import function in *PowerFactory*. Therefore, *PowerFactory* imposes specific requirements on FMUs compliant with FMI versions 2 and 3. These requirements are identical for both versions and are detailed below.

- **Input and Output Signals:**
  - An FMU input signal can be either a scalar or a one-dimensional array.
  - An FMU output signal can be either a scalar or a one-dimensional array.
- **Parameters:**

- An FMU parameter can be a scalar, a one-dimensional, or a two-dimensional array.
- **Internal Variables:**
  - An FMU internal variable can be of any FMI-supported type. However, only scalar, one-dimensional, or two-dimensional arrays can be monitored in a *Results* object and subsequently plotted.
  - Additionally, only scalar, one-dimensional, or two-dimensional arrays can have their current values displayed (e.g., using the *Network Model Manager*).

#### Interface scope and limitations:

- FMI models can be used in both RMS and EMT simulations.
- FMI models based on *FMI for Model Exchange* are considered within the Modal/Eigenvalue Analysis.
- FMI models based on *FMI for Co-simulation* are not considered within the Modal/Eigenvalue Analysis.

#### 30.14.1.2 How to integrate an FMU into *PowerFactory*

Perform the following steps:

- Create a new *Modelica Model Type* (*TypMdl mdl*) (refer to Section 30.12.1)
- On the *Basic Options* page set the *Simulation model type* to “Compiled model”
- Assign the file path to the FMU (file path must include the FMU file name and *.fmu* file extension). The FMU must support either a 64-bit or a 32-bit system architecture. It is recommended to use 64-bit FMUs whenever possible.

---

**Note:** *PowerFactory* is supplied based on a 64-bit architecture; hence 64-bit compiled DLL files of all interfaced external models are typically required. *PowerFactory* is capable of loading both 64-bit and 32-bit DLL files but, for simulation performance reasons it is recommended to use 64-bit compiled DLL files.

---

- Switch to the *Advanced* tab of the *Basic Options* page and check the displayed *FMU* information.
- Check the *Variable Declarations* page. If successfully loaded then the model variables should be listed (read-only) in this page.
- Ensure adequate initialisation procedure:
  - By default, all **inputs** of the FMU are assumed to be externally initialised i.e. the upstream model to which an input is connected to will provide an initial value. This is defined on the *Variable Declarations* page where, by default, all inputs have the *Initialisation* property set to “External”.
  - By default, all **outputs** of the FMU are assumed to be locally initialised i.e. the initial value of the FMU output is calculated locally and made available as the initial value to any downstream model(s) whose inputs are connected to the respective FMU output. This is defined under *Variable Declarations* page, where all outputs have the *Initialisation* property set to “Local”.
  - The initialisation procedure follows the rules and guidelines documented in Section 30.10.2.9.
  - It is possible to locally initialise one or more inputs of the FMU by setting the corresponding *Initialisation* property of a specific input to *Local*. This allows a backwards initialisation procedure of “unknown” inputs and providing initial start values to any connected signals of upstream models.
  - If at least one FMU input has a *Local* initialisation property, then the *Local Initialisation* page is activated and can be edited. Corresponding initialisation statements can be programmed for any of the locally initialised inputs, as described in Section 30.12.2.4.

**Note:** The script stored in the *Local Initialisation* page is not part of the FMI standard specification nor is it automatically packed in the *.fmu* file when performing an FMU Export.

- It is possible to configure one or multiple outputs to be initialised externally by setting the corresponding *Initialisation* property of a specific output to *External*. The initial value of an externally initialised output is provided by the corresponding input of the downstream model to which the FMU output is connected to. In this situation, the *Local Initialisation* script will allow the use of externally initialised output variables, such that locally initialised inputs will be dependent on the initial values of these outputs.
- Click **OK**.
- Design/Create a *Composite Model Frame* (*BlkDef* ) that contains a corresponding *Slot* for the FMU (see Section 30.2). Make sure that the FMU input and output signals are inter-connected with other model components via the inputs/outputs of various *Slots* e.g. measurement devices or generator elements.
- Create a *Modelica Model* (*ElmMdl* ) and reference to it the *Modelica Model Type* of the FMU (see Section 30.8.2).
- Use the Edit dialog of the *Modelica Model* to access the parameter list of the FMU and configure the model accordingly. Note, the parameterisation is shown using a tree-like view if parameters are declared using parameter structures.
- Create a corresponding *Composite Model* (*ElmComp* ) and assign the FMU to a relevant slot (see Section 30.2.6)
- Run the model in a power systems simulation using either *RMS Simulation*, *EMT Simulation* or a single/multiple domain co-simulation, depending on the FMU model design and capability.

To perform various management actions on the associated compiled files (i.e. FMU) of the *Functional Mockup Interface*, please refer to Section 30.14.4.

---

**Note:** Matlab® and Simulink® are registered trademarks of The MathWorks, Inc.  
Please see [mathworks.com/trademarks](https://www.mathworks.com/trademarks).

---

### 30.14.1.3 Exporting FMUs from *PowerFactory* (ComFmuexport)

An overview of the proposed *FMU Export* tool-chain is shown in Figure 30.14.3, in which a *Modelica Model Type* (*TypMdl* ) can be exported from *PowerFactory* and later on, imported in an *FMI Import* compliant third party simulation tool.

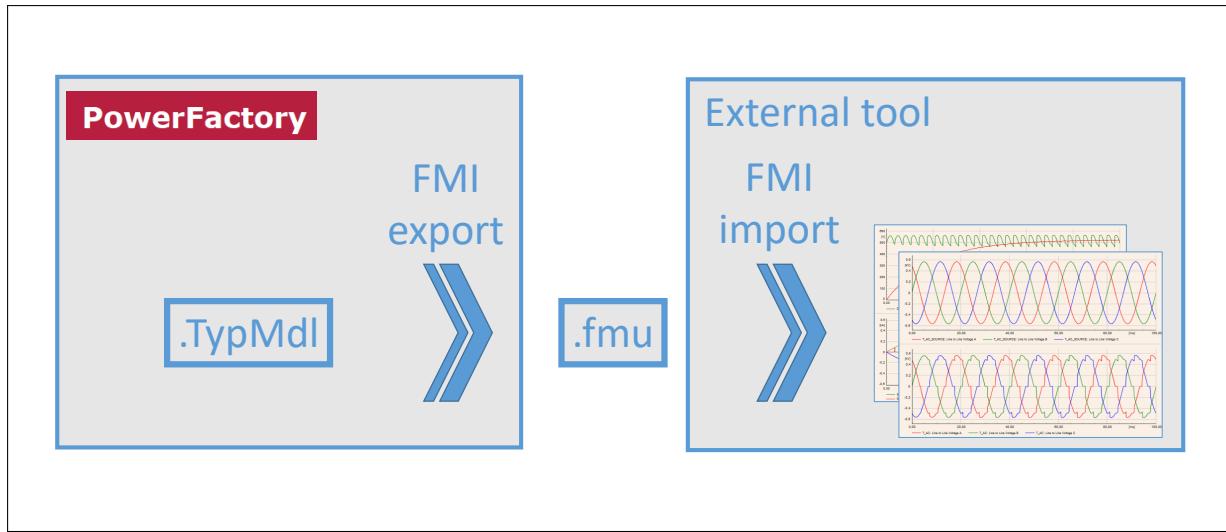


Figure 30.14.3: Overview of exporting a *Modelica Model Type* into an Functional Mock-up Unit (FMU)

The *FMU Export* function exports a *Modelica Model Type* as a Functional Mock-up Unit (FMU) based on the standardised *Functional Mock-up Interface (FMI)*, version 2.0. The *FMU Export* function is supported only by *PowerFactory* licences containing the licensing module “FMU Export”.

*PowerFactory* exports the *Modelica Model Type* using both FMI variants:

- *FMI for Model Exchange* and
- *FMI for Co-simulation*.

A standard compatible *.fmu* file is generated upon finalising this process which includes: model parameterisation, model binary files and model source code.

Exported models are compatible for importing in any of the *third-party tools* currently supporting the import function for either *FMI for Model Exchange* or *FMI for Co-simulation*.

#### 30.14.1.4 How to export a *Modelica Model Type* from *PowerFactory*

Perform the following steps:

- Open the dialog of the *Modelica Model Type* (*TypMdl*) to be exported
- Click on the button **FMU Export**.
- Customise the FMU Export via the shown FMU export dialog
- Click **OK**

The export is executed based on the following configuration options (found in the FMU Export dialog):

- **Model To Export:** A reference to the model to export. If the *FMU Export* is started by pressing the **FMU Export** button of a given *Modelica Model Type*, then this field is automatically referenced (and set to read-only).
- **Destination Folder:** A text field where the file path of the exported FMU can be entered.
- **FMI version:** Choose between FMI version 2 or 3.
- **Include source code files:** A checkbox that enables the export of the source code files along with the other FMU components.

- **Print compilation output:** A checkbox that enables the FMU Export to print compilation messages to the *PowerFactory* output window.
- **Variables export:** This option is shown only when exporting clocked Modelica models
  - Export all variables;
  - Export of interface variables only (inputs, outputs and parameters).
- **Sampling options:** The sampling options are shown only when exporting clocked Modelica models
  - **Sampling rate via sampling time parameter:** The discrete model is exported using a *Sampling time* parameter (name and value can be customised in the dialog)
  - **Sampling rate via sampling frequency parameter:** The discrete model is exported using a *Sampling frequency* parameter (name and value can be customised in the dialog)
  - **Sampling rate via hardcoded sampling time:** The discrete model is exported using a hardcoded *Sampling time* (value can be customised in the dialog)
  - **Sampling rate via hardcoded sampling frequency:** The discrete model is exported using a hardcoded *Sampling frequency* (value can be customised in the dialog)

Note, since the *FMU Export* is executed on the *Modelica Model Type*, any model parameterisation done in the *Modelica Model* is not included in the *FMU Export*. Default values of parameters are assigned, if existing, from the *Default* parameterisation fields (refer to option *Show default/start values of parameters/variables in “Variable Declarations”* described in Section 30.12.2.2).

For example, for the simple PI controller exemplified in Section 30.11.1, the *FMU Export* function may generate along with the standard C source and binary files, a corresponding *Modelica* model file (.mo file), as follows:

```
model DiscreteController
parameter Real Kp(unit "")=0.05"Proportional gain";
parameter Real Ki(unit "1/s")=20"Integral gain";
input Real reference(unit="p.u.",start=1)"Reference signal (real, continuous-time)";
Real reference_clocked(unit="p.u.",start=1)"Reference signal (real, continuous-time)";
input Real measurement(unit="p.u.",start=1)"Measured signal (real, continuous-time)";
Real measurement_clocked(unit="p.u.",start=1)"Measured signal (real, continuous-time)";
output Real control(unit="p.u.')"Control command (real, discrete-time)";
Real control_clocked(unit="p.u.')"Control command (real, discrete-time)";
Real error(unit="p.u.')"Control error (real, discrete-time)";
Real x(unit="p.u.");
Real dt(unit="s");
Real internalTime__;
parameter Real Ts =0.05;

equation
when Clock(Ts) then
internalTime__ = time;
reference_clocked=sample(reference);
measurement_clocked=sample(measurement);
end when;
control=hold(control_clocked);

algorithm
if (firstTick(internalTime__)) then
error:= reference_clocked - measurement_clocked;
dt := interval(x);
control_clocked :=1;
x:= control_clocked /Ki;
else
error:= reference_clocked - measurement_clocked;
```

```

dt := interval(x);
x:=previous(x)+error*dt;
control_clocked:=Kp*error+Ki*x;

end if;

end DiscreteController;

```

### 30.14.1.5 Using *Snapshots* with FMU models

*PowerFactory* supports serialisation of FMU states, by saving/loading the FMU state variables within/from a model snapshot (more information on *Snapshots* is provided in Section 29.3.7). An FMU shall be considered within a model *Snapshot* if the following FMU flags (within its model description XML) are set to *true*:

- canGetAndSetFMUstate
- canSerializeFMUstate

## 30.14.2 External C Interface acc. to IEC 61400-27

It is possible to directly interface external models compliant with IEC 61400-27-2 Edition 1 Annex G (and former IEC 61400-27-1 Edition 1 Annex F) specifications (referred to in this section as *IEC interface*). The interfaced models can be used for both RMS and EMT type simulations (provided they are appropriately designed and intended for the specific simulation type).

The IEC 61400-27-1 Edition 1 Annex F and IEC 61400-27-2 Edition 1 Annex G specify a common standardised binary simulation model interface which can be interpreted by any third party software. The advantage to this approach is that the same simulation model (i.e. binary code) can be used in multiple simulation environments.

The integration of *IEC interface* models is realised within *PowerFactory* by employing the object classes *Modelica Model Type* (*TypMdl* *mdl*) and *Modelica Model* (*ElmMdl* *mdl*). A fully automated process of loading an *IEC interface* model is supported, thus allowing the user to focus on the analysis itself and to not waste time on the cumbersome integration of external models into a power system model.

### 30.14.2.1 How to integrate an *IEC interface* model into *PowerFactory*

Carry out the following steps:

- Create a new *Modelica Model Type* (*TypMdl* *mdl*) (refer to Section 30.12.1).
- On the *Basic Options* page, check the “Compiled model” flag.
- Assign the file path to the DLL file (file path must include the file name and *.dll* file extension). Note the selection dialog filters for specific file types - make sure that an *IEC file* (*.dll*) is selected.

**Note:** *PowerFactory* is supplied based on a 64-bit architecture; hence 64-bit compiled DLL files of all interfaced external models are typically required. *PowerFactory* is capable of loading both 64-bit and 32-bit DLL files of compatible interfaces but, for simulation performance reasons it is recommended to use 64-bit versions whenever possible.

- 
- If successfully loaded, the model inputs, outputs and parameters are then listed (read-only) in the corresponding fields of the *Modelica Model Type*. Furthermore, a complete list of model information is provided in the *Description* page of the dialog, including relevant model parameters such as the sampling period of the model (parameter *FixedStepBaseSampleTime* of the “StaticExtSimEnvCapi” model information structure).

- Ensure adequate initialisation procedure:
  - By default, all **inputs** of the *IEC interface* model are assumed to be externally initialised i.e. the upstream model to which an input is connected to will provide an initial value. This is defined on the *Variable Declarations* page where, by default, all inputs have the *Initialisation* property set to “External”).
  - By default, all **outputs** of the *IEC interface* model are assumed to be locally initialised i.e. the initial value of the *IEC interface* model output is computed locally and made available as initial value to any downstream model(s) whose inputs are connected to the respective IEC model output. This is defined under *Variable Declarations* page, where all outputs have the *Initialisation* property set to “Local”).
  - The initialisation procedure follows the rules and guidelines documented in Section 30.10.2.9.
  - It is possible to locally initialise one or more inputs of the *IEC interface* model by setting the corresponding *Initialisation* property of a specific input to *Local*. This allows a backwards initialisation procedure of “unknown” inputs and providing initial start values to any connected signals of upstream models.
  - If at least one *IEC interface* model input has a *Local* initialisation property, then the *Local Initialisation* page is activated and can be edited. Corresponding initialisation statements can be programmed for any of the locally initialised inputs, as described in Section 30.12.2.4.

**Note:** The script stored in the *Local Initialisation* page is not part of the IEC 61400-27 standard specification.

- It is possible to configure one or multiple outputs to be initialised externally by setting the corresponding *Initialisation* property of a specific output to *External*. The initial value of an externally initialised output is provided by the corresponding input of the downstream model to which the *IEC interface* model output is connected to. In this situation, the *Local Initialisation* script will allow the use of externally initialised output variables, such that locally initialised inputs will be dependent on the initial values of these outputs.

- Click **OK**.
- Design/Create a *Composite Model Frame* (*BlkDef*  ) that contains a corresponding *Slot* for the *IEC interface* model (see Section 30.2). Make sure that the *IEC interface* model input and output signals are interconnected with other model components via the inputs/outputs of various *Slots* e.g. measurement devices or generator elements.
- Create a *Modelica Model* (*ElmMod*  ) and reference to the *Modelica Model Type* of the *IEC interface* model (see Section 30.8.2).
- Use the Edit dialog of the *Modelica Model* to access the parameter list of the *IEC interface* model and configure the model accordingly.
- Create a corresponding *Composite Model* (*ElmComp*  ) and assign the *Modelica Model* to a relevant slot (see Section 30.2.6).
- Run the model in a power systems simulation using either *RMS Simulation*, *EMT Simulation* or a single/multiple domain co-simulation, depending on the *IEC interface* model’s design and capability.

**Note:** The DSL Model Type (*BlkDef*) also supports *IEC interface* models, for the purpose of backward compatibility (legacy) with earlier *PowerFactory* versions. Nevertheless, it is recommended to migrate such configurations in order to use a *Modelica Model Type*. For information on the use and configuration of *IEC interface* models using the DSL Model Type, please refer to the *IEC interface* description provided in the *PowerFactory 2022 User Manual*.

---

### 30.14.2.2 Using dynamic models based on the *IEC interface*

The model can be integrated into a *Composite Model Frame* in the same manner as a typical *Modelica Model Type* (as described in Section [30.6.1](#)).

Under the *Description* page of the *Modelica Model Type*, the parameter *FixedStepBaseSampleTime* of the “*StaticExtSimEnvCapi*” model information structure is displayed (with unit in seconds). The simulation solver interacts by calling the functions “*Model\_Update*” and “*Model\_Outputs*” at integer multiples of the sampling period *FixedStepBaseSampleTime*, in a precise manner (i.e. simulation interrupt at precise time).

#### Interface variable names:

- The DLL *Model developer* is responsible with the correct definition of the interface. All variable names used in this interface must comply with the specifications of the *IEC Interface* (although not explicitly stated, the C variable naming rules should be adopted for all variables that are part of the *IEC Interface*).
- The integration of the *IEC Interface* model into *PowerFactory* is done by connecting this model using a *Composite Model Frame*. Scalar and vector (1-dimensional array) signals are supported for the model’s inputs and outputs. Refer to Section [30.2.7](#) for information on how signals (scalars or vectors) are interconnected within a user-defined dynamic model.

#### Interface scope and limitations:

- Simulation models based on the *IEC 61400-27* Interface can be used in both RMS and EMT simulations.
- Simulation models based on the *IEC 61400-27* Interface are not considered within the Modal/Eigenvalue Analysis.
- The calculation of initial conditions procedure in *PowerFactory* RMS and EMT simulation is only partly supported by the *IEC Interface* compliant models. That is, only forward initialisation is possible: outputs and internal model variables or states are initialised based on knowledge of input quantities only. Inputs cannot be initialised by an *IEC Interface* compliant model.

To perform various management actions on the associated compiled files (i.e. DLLs) of the *IEC Interface*, please refer to Section [30.14.4](#).

### 30.14.2.3 Using *Snapshots* with *IEC Interface* models

*PowerFactory* supports saving/loading of *IEC Interface* model states to/from a model snapshot (more information on *Snapshots* is provided in Section [29.3.7](#)). An *IEC Interface* model is automatically considered within a model *Snapshot* if interfaced using *Modelica* objects (*TypMd\ElmMdl*). The following *IEC Interface* model state variables are saved/loaded into/from a *Snapshot*:

- *InstanceExtSimEnvCapi* → *MiscStates*

- 
- Note:**
- Model states stored within *InstanceExtSimEnvCapi* → *ContinuousStates* and *InstanceExtSimEnvCapi* → *StateDerivatives* are not saved within a *PowerFactory* model *Snapshot* as these two fields are not used by *PowerFactory*.
  - *IEC Interface* models using DSL Model objects (*BlkDef\ElmDsl*) are not included in a model *Snapshot*.
-

### 30.14.3 DSL-C Interface

The *DSL-C Interface* provides the framework to create dynamic models developed entirely in the C programming language. For simplicity, these models will be hereafter referred to as “compiled DSL models”. It is possible to generate C-source code and compile *DSL Model Types* into binary files. The generated C source files need to be compiled (using an external ANSI-C compiler) into a Dynamic Link Library (DLL) in order to be used within a dynamic simulation.

---

**Note:** *PowerFactory* is supplied based on a 64-bit architecture; hence 64-bit compiled DLL files of all interfaced external models are typically required. *PowerFactory* is capable of loading both 64-bit and 32-bit DLL files of the DSL-C Interface but, for simulation performance reasons, it is recommended to use 64-bit versions whenever possible. In particular for the support of 32-bit architecture, only DLLs based on the DSL-C Interface version 220001 can be loaded.

---

The supported C compilers are listed on the *External Applications* page of the Configuration dialog (accessed via *Tools* → *Configuration*).

*PowerFactory* provides “compiled DSL models” for all DSL models available in the “Standard Models” global library database folder and those located in the “Standard Models” subfolder of the *PowerFactory* installation (e.g. ‘C:\Program Files\DIgSILENT\PowerFactory XXXX\Standard Models’).

#### 30.14.3.1 DSL-to-C Interface Converter

Any *DSL Model Type* may be compiled into a *DSL C Interface* model using the automatic DSL-to-C interface converter. As with any DSL model, it is recommended that the *Calculation of Initial Conditions* of a *DSL Model* representing the *DSL Model Type* is performed and no internal DSL warnings are displayed during the simulation. Creating a *DSL C Interface* model requires two steps:

1. Creation of C source files from the *DSL Model Type*; and
2. Compilation of source files into a DLL using an external compiler.

To create the C source files for an existing *DSL Model Type* (which must be a non-macro, highest-level *DSL Model Type*), the following steps are required:

- Open the *Advanced* tab of the *DSL Model Type* dialog as shown in Figure 30.14.4;
- As *Compilation options* fill in the *Author*, *Company*, *Copyright* and *Version* fields with relevant information;
- An initial check of the model is required and can be performed by clicking on **Check for compilation** button. Check the output window for the verification status;
- When a `Block is ok` message is displayed in the output window, the code generation process may be started by clicking on the **Compile** button;
- A folder selection dialog is opened with a newly-created model folder within the *PowerFactory* workspace location;
- Click on **OK**. If the process completes successfully, a status message will be reported in the output window (e.g. Model ‘`BlockName`’ created and stored as ‘`C:\Users\user\AppData\Local\DIgSILENT\PowerFactory xxxx\Workspace.nnnnnn\db\User Models\BlockName\MyModel.c`’).

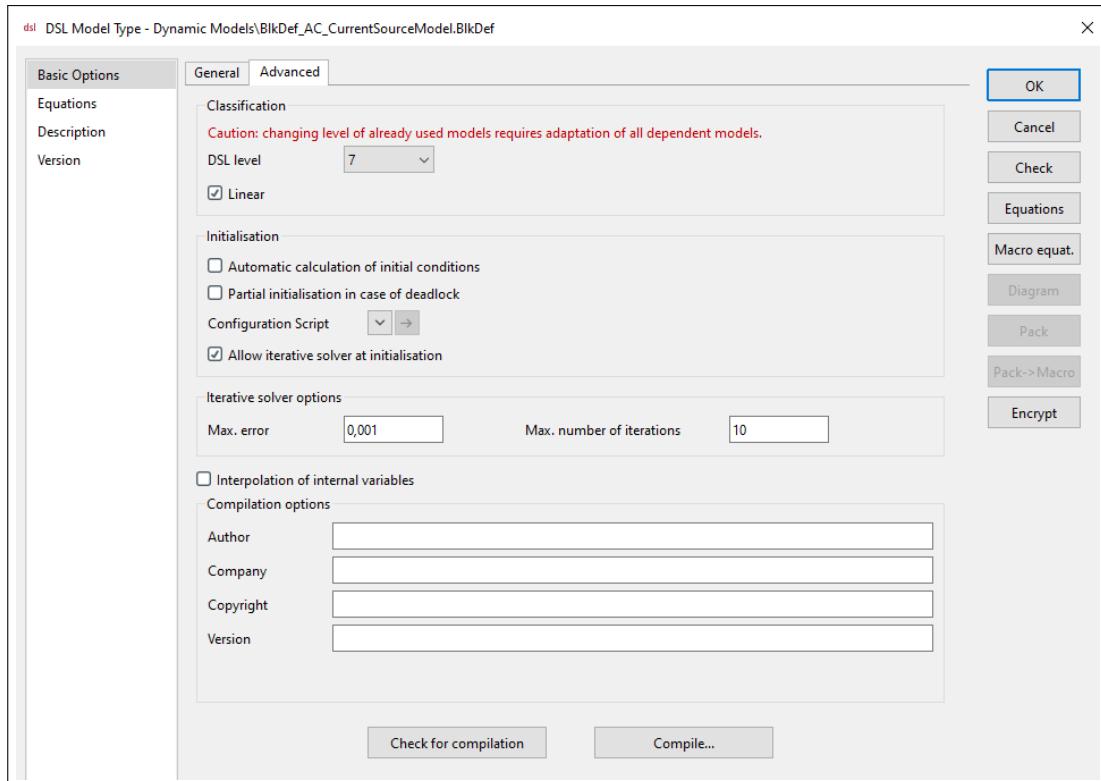


Figure 30.14.4: *DSL Model Type* dialog, *Basic Options* page, *Advanced* tab

The generated source files are listed below:

- dig userModel.c - dDLLMain declaration function;
- dig userModel.def - module-definition file containing one or more module statements that describe various attributes of the DLL;
- dig userModel.h - contains necessary function prototype declarations;
- dig userModelAdvanced.h - contains additional function prototypes;
- BlockName.vcxproj , where “BlockName” is the name of the *DSL Model Type* from which the source files have been created - Visual Studio 2012©project file;
- MathConstants.h - contains various mathematical constant definitions;
- ModellInterface.h - declaration of advanced functions;
- MyModel.c - C source file containing function definitions of the dynamic model.

The typical process of compiling the source files into a DLL should be followed according to the instructions of the external compiler. In Visual Studio 2012, the “BlockName.vcxproj” project file can be opened. The user needs to make sure that the 64-bit architecture is selected when building the DLL in order to comply with *PowerFactory* requirements.

After creating the DLL, the file can be readily used in *PowerFactory*. On the *General* tab of the *DSL Model Type* dialog, check the “Compiled model” radio button. A folder selection text box is shown. Introduce the file path (including the file name) in the text box or click on the “...” button to navigate and select the DLL file. For future uses of the compiled model, *PowerFactory* stores the previously compiled model directory path as default.

*PowerFactory* automatically loads the DLL and in the *General* tab of the *DSL Model Type* (shown in Figure 30.14.5) it displays relevant information. The list of used variables (output and input signals, state

variables, etc.) is shown in the Variables area (non-editable) and a summary of the DLL-file is shown in the “DLL info” area. The *Source* field states the name of the compiled model. The *PowerFactory* version, creation date, author, company and copyright owner are shown. The checksum field contains a 16-digit unique identifier of the original DSL model on which it has been based. If a DSL model is defined as well, then the checksum of the DSL block will also be shown in the “DSL-info” area (below the “DLL info” area). Based on the two available checksums, a direct comparison between the compiled and the DSL model can be performed to verify their equivalence.

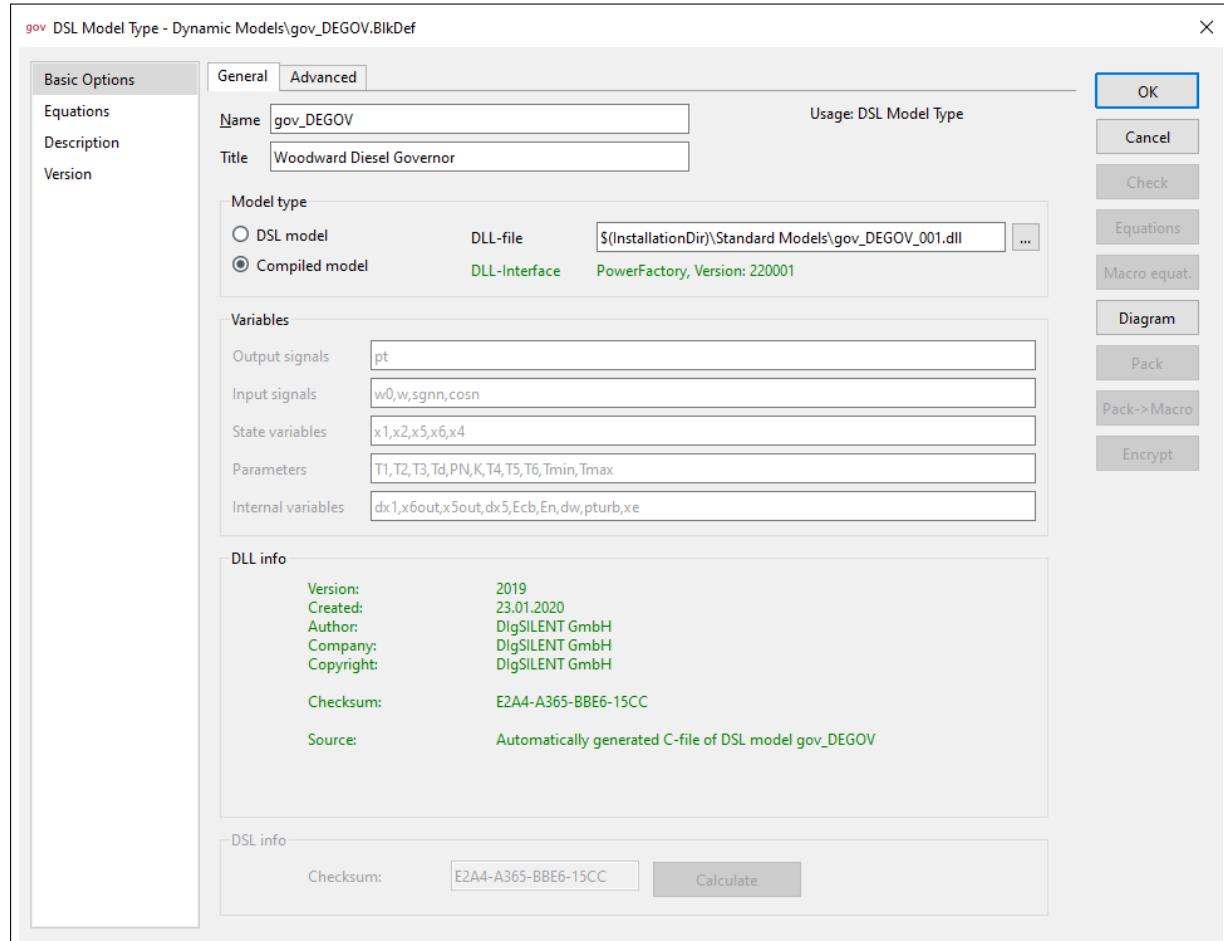


Figure 30.14.5: Compiled model options

The model can be integrated in a *Composite Model Frame* in the same manner as a typical *DSL Model Type* (as described in Section 30.6.1).

More information about the *DSL C Interface* including the description of the interface and examples, is available on the *External C Interface for dynamic models*, which is located in the “External C Interface” subfolder of the *PowerFactory* installation (e.g. ‘C:\Program Files\DlgSILENT\PowerFactory XXXX\External C Interface’).

To perform various management actions on the associated compiled files (i.e. DLLs) of the DSL-C Interface, please refer to Section 30.14.4.

#### 30.14.4 Management of compiled files of dynamic models

When a dynamic model is used in a simulation, corresponding compiled files are either automatically generated (in the case of *PowerFactory* native models i.e. DSL and Modelica) or accessed from the respective file paths (in the case of externally interfaced dynamic models). In all cases, the compiled files

are supplied in the form of Windows Dynamic Link Library (DLL) files. Upon executing the *Calculation of Initial Conditions* command, all relevant compiled files are loaded by *PowerFactory* in order to be used during the simulation.

In certain cases, interaction with the relevant compiled files is needed by the user. The *DLL Manager* command (*ComDllmanager*) facilitates a number of user actions, as described further below.

#### 30.14.4.1 How to create a new *DLL Manager* command (*ComDllmanager*)

To create a new *DLL Manager* command object, do the following:

- Open the Data Manager and navigate to a location which allows the creation of the *DLL Manager* command object (e.g. a study case of the currently active project);
- Click on the *New Object* button in the toolbar of the Data Manager;
- Search for the “*DLL Manager*” keywords and select the *DLL Manager* object type;
- Click **OK**.

#### 30.14.4.2 The *DLL Manager* command

The dialog of the *DLL Manager* command, allows the execution of various actions applicable to a number of DLL types. These types are listed below and can be selected within the *DLL Type* pane:

- DPL DLL: this category corresponds to DLL files used within the function “DPL Functions Extensions”, with documentation available on the menu *Help → Additional Packages*. DPL DLL files are not used within dynamic simulations.
- Exdyn DLL: this category corresponds to externally interfaced dynamic models built using the interfaced “digexdyn”. This interface is considered as legacy.
- Exfun DLL: this category corresponds to externally interfaced dynamic models built using the interfaced “digexfun”. This interface is considered as legacy.
- Dynamic Model DLL: this category corresponds to DLL files used by PowerFactory *native models* and by externally interfaced dynamic models using non-legacy interfaces.

To perform an action, select first a certain DLL type, as previously listed. The dialog of the *DLL Manager* command supports the following actions:

- Load: loads all identified DLL files which are relevant for the certain DLL type;
- Unload: unloads all previously loaded DLL files which are relevant for the certain DLL type;
- Delete automatically compiled: deletes all identified DLL files which are automatically compiled (i.e. *PowerFactory* native models).

The dialog of the *DLL Manager* command has the following buttons:

- Execute: executes the command according to the selected configuration;
- Close: closes this dialog and saves any applied modifications to the dialog's configuration;
- Cancel: closes this dialog and discards all modifications;
- Report: performs a general report on currently relevant DLL files.

### 30.14.5 MATLAB Interface

Interfacing Matlab® Simulink® models with *PowerFactory* can be done using the Functional Mock-Up Interface (FMI), described in Section [30.14.1](#).

---

**Note:** Matlab® and Simulink® are registered trademarks of The MathWorks, Inc.  
Please see [mathworks.com/trademarks](http://mathworks.com/trademarks).

---

## 30.15 Developing User-defined Power Electronics Models for EMT Simulation

*PowerFactory* supports various commonly used power electronics (PE) based devices within several built-in simulation models. For example:

- three phase line commutated converters are supported by the *Rectifier/Inverter* element (*ElmRec*) and the HVDC-LCC element (*ElmHvdclcc*);
- three phase two-level and MMC type voltage source converters are supported by the *PWM Converter* element (*ElmVsc*);
- three phase Static Var Systems can be built using the SVS element (*ElmSvs*).

There are many advantages to using built-in *PowerFactory* models for power electronics, ranging from the simplicity in configuration to the full support of all relevant calculation functions. Nevertheless, for power electronics equipment which is not directly supported by the built-in models or for which various detailed studies require access to internal model components, *PowerFactory* provides users the possibility to build customisable PE equipment. There are two main options when developing a user defined PE equipment for EMT simulations:

- Simulation model development **as a Submodel of a built-in element** ;
- Simulation model development **as an independent EMT model** (neither belonging to nor developed under any specific built-in element).

If the PE equipment is developed within a built-in element, then it can be connected to the external power system via predefined *connection ports* belonging to a *Submodel*. These ports can be either AC or DC and correspond to the output terminals of the built-in model. In case of three-phase AC terminals, the corresponding *connection ports* will be “per-phase” e.g. an ABC-N terminal will have 4 connection ports, one for each phase plus the neutral wire. This user defined PE equipment model will be operational during EMT Simulation only. The *connection ports* will enable an electrical connection between the *Submodel* and the external power system immediately after initialisation. More information is available in Section [30.15.1](#).

If the PE equipment is developed outside a built-in model, then the modelling approach is standard as with any other detailed model: individual components are added to the grid and linked such that the topology and configuration of the model is adequately represented. The connection between the DC components and the AC part of the model or the external power system can be done by using the *AC/DC Connector* element. All *PowerFactory* calculation functions will be operational in such a situation. Nevertheless, the *AC/DC Connector* element will link the AC and DC sides (and thus enable an electrical connection) only within EMT simulation, immediately after initialisation. More information is available in Section [30.15.2](#).

---

**Note: DC components are typically used for PE models** - in both modelling approaches, the power electronics dedicated components (diode, IGBT, thyristor, MMC valve, etc) are DC elements. Therefore, the PE models are intended to use “DC” components only (nevertheless, it is not

forbidden to add AC components, if special cases require it). The PE model can afterwards be connected to the AC power system either by directly using *AC/DC Connectors* in the single line diagram (i.e. when the PE model is developed outside a built-in element) or via the *connection ports* of a *Submodel* (when the PE model is defined within a built-in element). The DC elements of the PE model are inter-connected by using DC nodes (terminals). Make sure that the rated voltage of the internal DC nodes is appropriately set to a common value throughout the model.

---

### 30.15.1 Model development as a *Submodel* of a built-in element

The following built-in elements are currently supported:

- PWM Converter (*ElmVsc* and *ElmVscmono*)
- Static Var Systems (*ElmSvs*)
- DC-DC Converter (*ElmDcdc*)
- Static Generator (*ElmGenstat*)
- PV System Generator (*ElmPvsys*)

When developing a PE model within a built-in element make sure to assign a *Submodel* to one of the supported network elements. To do so, follow these steps:

- From the single line diagram, right-click on the built-in element;
- From the available options, select *Network Models* → *Submodel* → *New...* (if the element is not supported, then this option will not be available);
- In the *Submodel* selection dialog click on the “Default *Submodel* templates” folder.
- Select the *Submodel* template which corresponds to the element class of the built-in element e.g. for a *PWM Converter/2 DC connections* element (of class name *ElmVsc*) the template is named “*ElmVsc*”. Click **OK**;
- The Edit dialog of the *Submodel* is displayed. Click **OK**;
- The *Submodel* single line diagram is created based on the template. The connection ports are automatically added to the diagram (with their corresponding DC terminals). All DC terminals have the same default voltage set to 100 kV. Make sure to change this value accordingly when developing the model.

After the *Submodel* creation is complete, it is possible to add to the *Submodel* diagram all relevant DC components by using the *Drawing toolbar* (if the diagram is frozen, it needs to be un-frozen in order to add elements). Any element can be added to the diagram, nevertheless, all modelling rules and requirements existing in the standard model creation apply to this *Submodel* as well. A selection of these rules is provided below:

- Elements are linked between themselves using terminals;
  - DC elements connect to DC terminals only - (typically, a *Submodel* will contain only DC elements and terminals);
  - AC elements connect to AC terminals only;
  - Whenever load flow calculation is executed, the *Submodel* and its components will be included in the calculation (but in a disconnected state, separated from the rest of the power system). This means that the *Submodel* must comply with the load flow data model requirements as well.
-

**Note:** Typically, the *Submodel* will not contain any voltage sources and hence, the *Submodel* will be de-energised within load flow. Whenever voltage sources exist in the *Submodel*, then the load flow calculation will energise the *Submodel* and the load flow will attempt to compute a steady state operating point. Even in this situation, the *Submodel* is still considered as being disconnected from the rest of the power system. On the other hand, the built-in model is operational and will decide the power flows on its terminals. Therefore, the built-in element to which this *Submodel* is assigned will not be influenced by its internal representation and will continue to operate as defined within its *Load Flow* configuration page.

For example, a three-phase, three-level (3L) neutral point-clamped (NPC) inverter can be defined for the PWM Converter element (*ElmVsc*) by implementing the corresponding topology using a *Submodel*, as shown in Figure 30.15.1. The model is available [here](#). The AC and DC connection ports are used to link the internal components with the elements to which the built-in element terminals are connected in the power system (either AC or DC power system parts, depending on the built-in element and corresponding terminals).

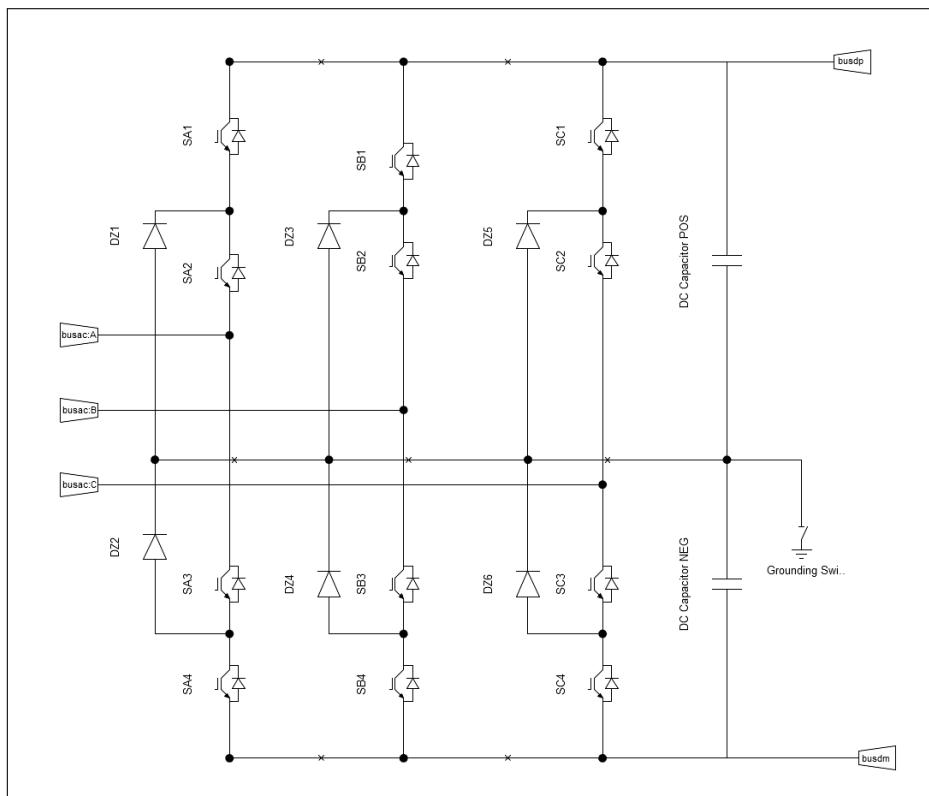


Figure 30.15.1: Three-level NPC converter example

The control system is designed such that each individual IGBT is controlled via its gate signal. Since the individual gate signal of one IGBT element is a scalar, the total number of gate signals is small i.e. 12. Consequently, the control system does not require any vectors for transferring the gate signals to the controlled elements, having therefore a typical DSL model structure as shown in Figure 30.15.2. Note that the *Converter* slot is only required when the control system is intended to support also the control of the built-in EMT Simulation model. This can be achieved, for example, whenever the *Submodel* element is set to “Out of service”. In this case, the built-in element will ignore the *Submodel* and make use of the dynamic controller’s output signals (i.e. the reference voltages) in order to internally generate the gate signals. Within the example provided the built-in element is configured as a two-level, detailed converter model (as an alternative choice).

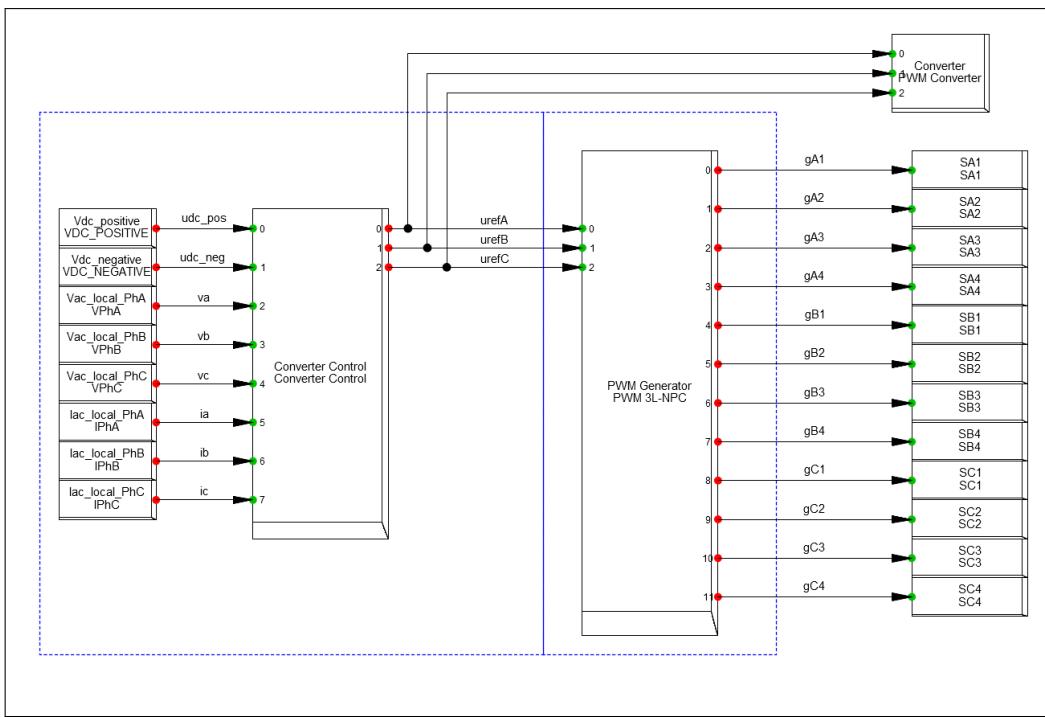


Figure 30.15.2: Three-level NPC converter - control structure

### 30.15.2 Model development as an independent EMT model

There are cases when there are no built-in elements that support the specific PE model configuration (e.g. AC drives with multi-phase front ends). In such cases, the alternative is to directly develop the simulation model within the single line diagram of a grid (*ElmNet*). When modelling as an independent EMT model, make sure to use the *AC/DC Connectors* for connecting the *DC components* with the AC part of the system. To do so, follow these steps:

- Choose a grid in which the PE model is intended to be developed (for models containing a large number of components, a separate grid could be created).
- Within the single line diagram add corresponding *AC/DC Connectors* (*ElmAcdconnector*) from the Drawing Toolbar. If the PE model is a three phase system, then three *AC/DC Connectors* are needed. The AC and DC sides of this element are correspondingly marked.
- Connect the *AC/DC Connectors* between an AC and DC terminals. It is assumed that the AC terminals correspond to the external power system part and the DC terminals to the PE model part. In the case of a multi-phase AC terminal, make sure to select the corresponding phase with the relevant *AC/DC Connector*.
- Further expand the PE model topology starting from the available DC terminals. Use the available DC components from the drawing toolbar (e.g. diodes, IGBTs, MMC Valves, DC current and voltage sources, DC resistances and inductances).
- Each DC component is inter-connected with another one via a DC terminal. Use an *internal node* bus type (i.e. dot terminal graphical representation) whenever useful.

**Note:** Since the PE model is developed directly within the single line diagram of a grid, the same rules apply regarding data consistency as with any other DC part of an electrical system. The only difference is that the *AC/DC Connector* acts as an open circuit during load-flow calculation (and any other calculation except the EMT Simulation).

For example, a 22 module, MMC based, STATCOM model can be represented in detail by developing the PE model directly within the single line diagram of a grid, as shown in Figure 30.15.3.

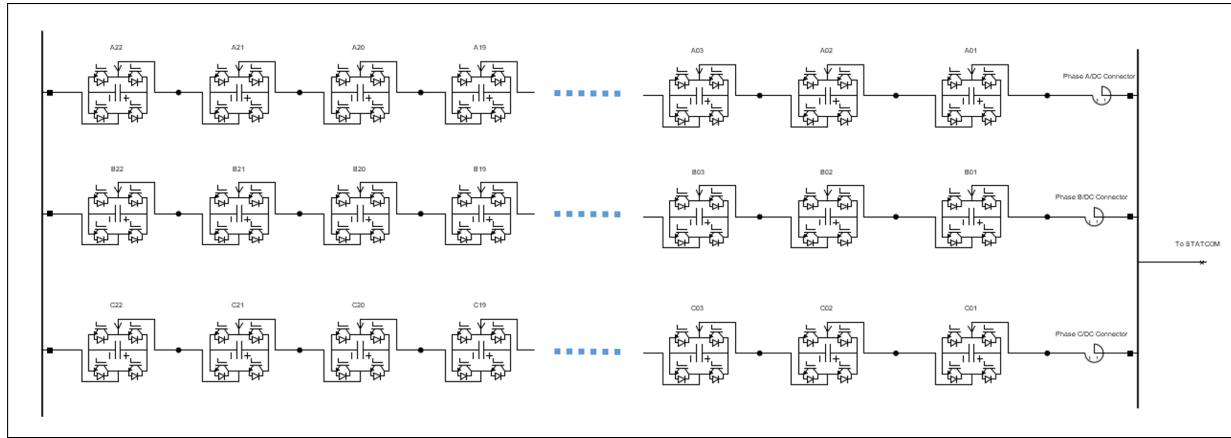


Figure 30.15.3: Detailed MMC STATCOM example - topology in wye

In this model, the most relevant DC components are:

- MMC Valve (*ElmMmcvalve*) and
- AC/DC Connectors, per-phase (*ElmAcdconnector*).

The MMC Valve model is configured as a full bridge, detailed model (in such a case, 4 IGBTs and a capacitor are represented internally). The model also supports the half-bridge configuration. For both configurations, the model can be set up either as a detailed (one single bridge), detailed equivalent circuit ( $n$  series connected bridges, individual IGBTs and module capacitors) or aggregated ( $n$  series connected bridges, single capacitor, controlled voltage source). For the detailed and detailed equivalent circuit variants, the IGBTs can be controlled individually. For the detailed equivalent circuit and aggregated variants, one MMC valve element can represent  $n$  series connected half- or full-bridges. More detailed information is available in the corresponding Technical Reference of this element.

The per-phase AC/DC Connectors (depicted as reactors on the right hand side of the diagram) can be configured to model a resistance and/or an inductance. The arm inductance is therefore included within the AC/DC Connectors.

The right-most terminal is an AC bus and connects the detailed STATCOM model with the upstream AC power system (optional STATCOM transformer, supply network, etc.) - refer to Figure 30.15.4 for an example.

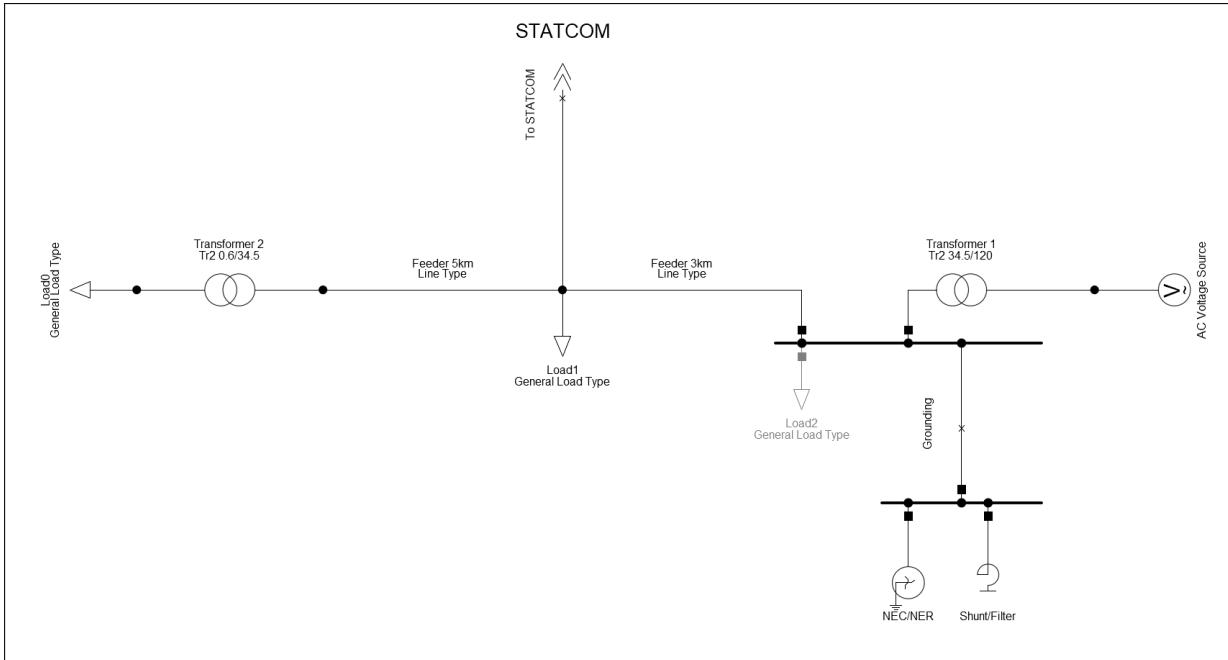
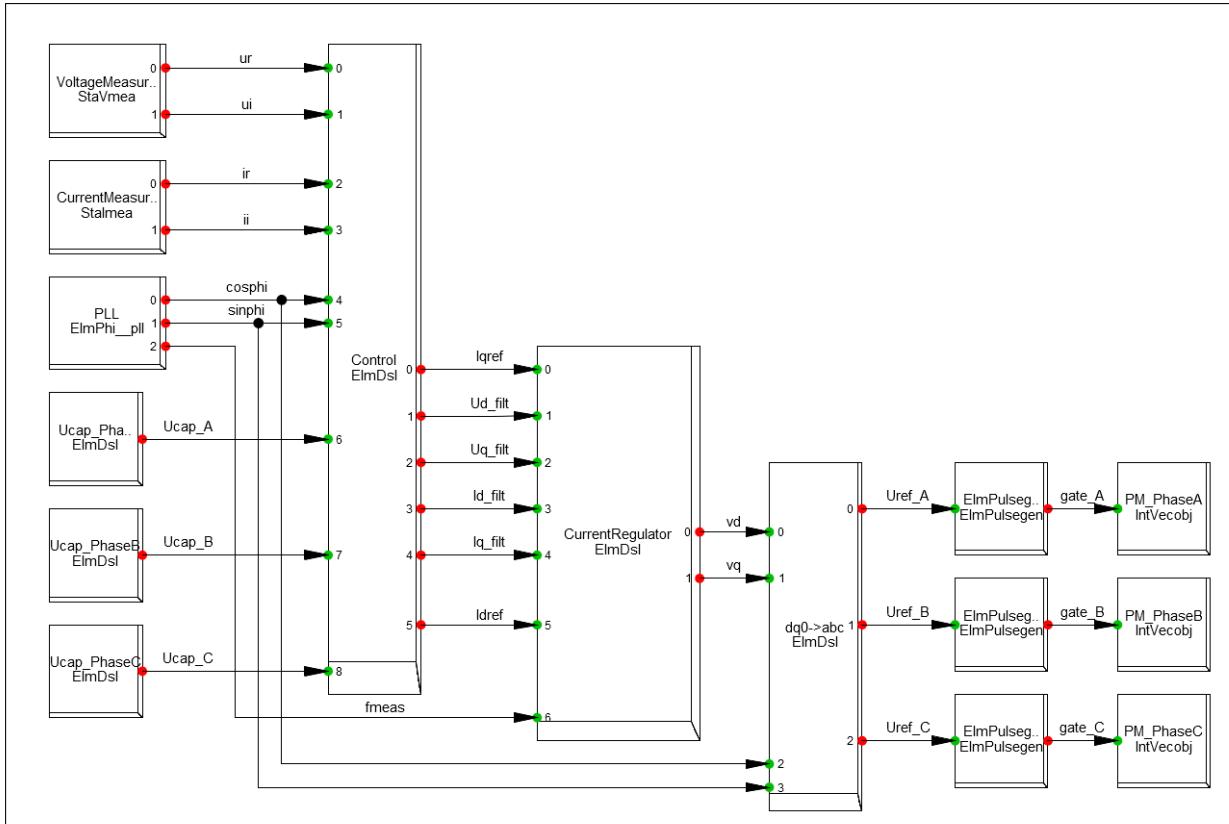


Figure 30.15.4: STATCOM's connected upstream network

The control of the STATCOM is more elaborate (refer to Figure 30.15.5) and requires the use of vector based signals within the corresponding *Composite Model* as well as a detailed PWM switching strategy for MMCs. Therefore, besides the typical DSL control system implementation, two important components are additionally employed:

- Pulse generator element (*ElmPulsegen*) and
- *Vector of Objects* (*IntVecobj*)

Figure 30.15.5: *Composite Model Frame of the STATCOM Controller*

The *Pulse generator* element (refer to the configuration in Figure 30.15.6) is used to generate an accurate set of gate signals for each MMC valve within a given arm of the STATCOM (in total three *Pulse generator* elements). The input of the model is the per-phase reference arm voltage that is to be generated at the entire arm's connection points. The output is a vector based signal (named *gate\_A* / *gate\_B* / *gate\_C*) which, in this case, has size  $4 \cdot 22 = 88$ . For more detailed information, please refer to the Technical Reference of the *Pulse generator* element (*ElmPulsegen*).

**Note:** Although the *Pulse generator* element demonstrates in this example the user-defined PE modelling techniques outside built-in elements, the same *Pulse generator* element could be used also for generating the gate signals for either (a) user-defined PE models represented inside built-in elements or (b) the PWM Converter built-in model (*ElmVsc*) if configured as a detailed, Type 4 model (refer to the Technical reference of the PWM Converter for more information).

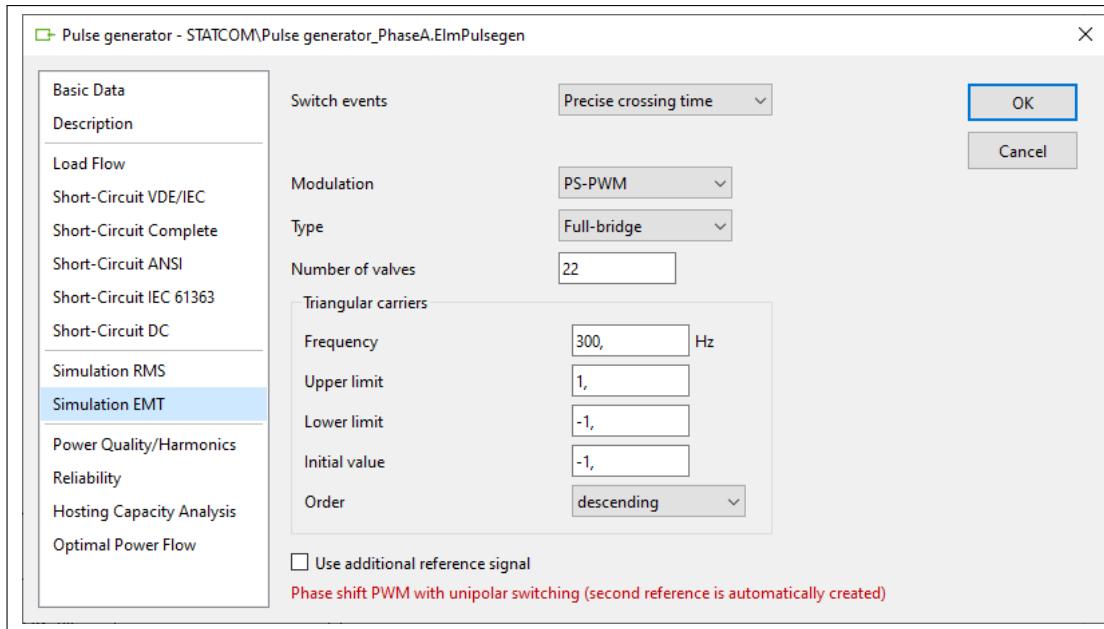


Figure 30.15.6: Pulse generator model (built-in element) for gate signal generation

The *Vector of Objects* element (*IntVecobj*) is used to distribute the generated gate signal vector to the individual MMC valves. A comprehensive description of the various possible configurations of connecting array signals in a high-level control system is provided in Section 30.2.7.

In the current configuration, the *Vector of Objects* element demultiplexes the input vector into either multiple scalars or sub-vectors, depending on the size of the source and destination signals. The source signal is the incoming signal connected to the corresponding slot of the *Vector of Objects* element within the Composite Model Frame. The destination signal is the input signal stated within the corresponding slot of the *Vector of Objects* element within the Composite Model Frame. This signal name must match one of the input signals of the controlled elements. The destination signal can be either a scalar or a vector.

In the case at hand, the following applies:

- The source signal is the vector signal *gate* of the *Pulse generator* element having a size  $4 \cdot 22 = 88$
- The destination signal is stated within the *Input Signals* text field of the slot assigned to the corresponding *Vector of Objects* element within the *Composite Model Frame*. The signal is named *gate* and it represents the input signal of the *MMC Valve* element, having a size 4 (the *MMC Valve* is configured as a Full Bridge);
- The destination elements are programmed within the configuration dialog of the *Vector of Objects* element (refer to Figure 30.15.7). Within this dialog there are 22 individual *MMC Valve* elements added.
- The demultiplexing logic is as follows:
  - the first destination element is the *MMC Valve* element named “A01”; the first 4 indexes of the source signal *gate* will be assigned (in the same order) to the destination signal *gate* (which is a vector signal of size 4) of the *MMC Valve* “A01”.
  - the second destination element is the *MMC Valve* element named “A02”; indexes 5 to 8 of the source signal *gate* will be assigned (in the same order) to the destination signal *gate* (which is a vector signal of size 4) of the *MMC Valve* “A02”.
  - the procedure continues until: (a) all destination elements have been parsed or (b) the last index of the source signal has been reached.

**Note:** The order within the configuration dialog of the *Vector of Objects* element is important! The demultiplexing logic parses all these objects in ascending order (i.e. first element in the table is processed first, second element is processed afterwards, and so on). The order can be changed using buttons **Move up** and **Move down**: select a row and then click on one of these buttons - the MMC Valve element will consequently be moved up/down.

---

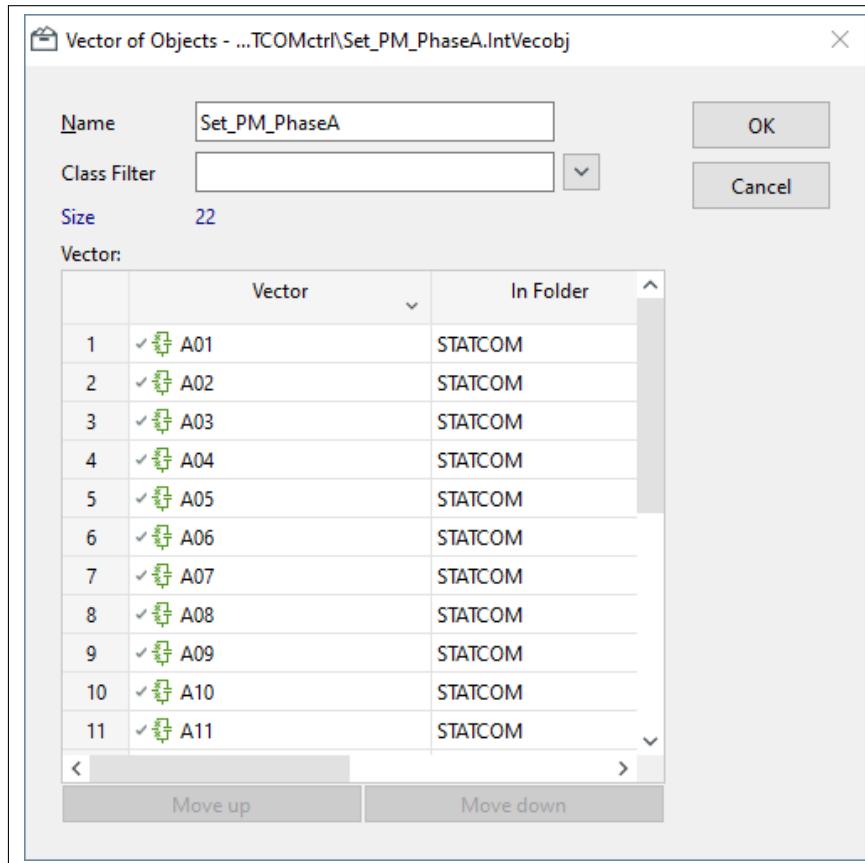


Figure 30.15.7: Distribution of the gate vector signal to the individual MMC valve elements

---

**Note:** Although the *Vector of Objects* element demonstrates in this example the user-defined PE modelling techniques outside built-in elements, the same element could be used also for distributing signals to either (a) user defined PE models represented inside built-in elements or (b) other applications, completely unrelated to EMT Simulation of power electronics (e.g. Power Plant Controllers dispatching individual wind turbines within an RMS Simulation).

---

## 30.16 DSL Reference

### 30.16.1 DSL Standard Functions

DSL Function	Short description	Example
<b>sin(x)</b>	sine	$\sin(1.2) = 0.93203$
<b>cos(x)</b>	cosine	$\cos(1.2) = 0.36236$
<b>tan(x)</b>	tangent	$\tan(1.2) = 2.57215$
<b>asin(x)</b>	arcsine	$\text{asin}(0.93203) = 1.2$
<b>acos(x)</b>	arccosine	$\text{acos}(0.36236) = 1.2$
<b>atan(x)</b>	arctangent, result within $(-\pi/2, +\pi/2)$	$\text{atan}(2.57215) = 1.2$
<b>atan2(y,x)</b>	principal value of arctangent of $y/x$ , i.e. $\text{atan}(y/x)$ shifted by $\pm\pi$ if $x < 0$ , result within $[-\pi, +\pi]$	$\text{atan2}(2.57215, 1) = 1.2$ $\text{atan2}(-2.57215, -1) = -1.9416$
<b>sinh(x)</b>	hyperbolic sine	$\sinh(1.5708) = 2.3013$
<b>cosh(x)</b>	hyperbolic cosine	$\cosh(1.5708) = 2.5092$
<b>tanh(x)</b>	hyperbolic tangent	$\tanh(0.7616) = 1.0000$
<b>exp(x)</b>	exponential value	$\exp(1.0) = 2.718281$
<b>ln(x)</b>	natural logarithm	$\ln(2.718281) = 1.0$
<b>log(x)</b>	$\log_{10}$	$\log(100) = 2$
<b>sqrt(x)</b>	square root	$\sqrt{9.5} = 3.0822$
<b>sqr(x)</b>	power of 2	$\text{sqr}(3.0822) = 9.5$
<b>pow(x,y)</b>	power of y	$\text{pow}(2.5, 3.4) = 22.5422$
<b>abs(x)</b>	absolute value	$\text{abs}(-2.34) = 2.34$
<b>min(x,y)</b>	smaller value	$\text{min}(6.4, 1.5) = 1.5$
<b>max(x,y)</b>	larger value	$\text{max}(6.4, 1.5) = 6.4$
<b>modulo(x,y)</b>	remainder of $x/y$	$\text{modulo}(15.6, 3.4) = 2$
<b>trunc(x)</b>	integer part	$\text{trunc}(-4.58823) = -4$
<b>frac(x)</b>	fractional part	$\text{frac}(-4.58823) = -0.58823$
<b>round(x)</b>	closest integer	$\text{round}(1.65) = 2$
<b>ceil(x)</b>	smallest larger integer	$\text{ceil}(1.15) = 2$
<b>floor(x)</b>	largest smaller integer	$\text{floor}(1.78) = 1$
<b>time()</b>	current simulation time	$\text{time}() = 0.1234$
<b>pi()</b>	$\pi = 3.141592\dots$	$\text{pi}() = 3.141592\dots$
<b>twopi()</b>	$2 \cdot \pi = 6.283185\dots$	$\text{twopi}() = 6.283185\dots$
<b>e()</b>	$2.718281\dots$	$\text{e}() = 2.718281\dots$

Table 30.16.1: DSL Standard Functions

**Remark:**

- Names of DSL standard functions are reserved. Note that DSL model variables may not have any reserved name.

### 30.16.2 DSL Special Functions

DSL Function	Short description
aflipflop	“Analog” flip-flop function
balanced	Function returning the balanced/unbalanced network representation
delay*	Delay function
event	Multi-purpose simulation event function
flipflop	Logical flip-flop function
gradlim_const*	Gradient limiter function
invlapprox	Inverse lapprox function
lapprox	Linear approximation based on a single dimensional array
lapproxext	Linear approximation based on a single dimensional array (function extension)
lapprox2	Linear approximation based on a two dimensional array
lastvalue*	Function returning the last valid value of a signal
lim_const	Nonlinear limiter function with constant limits
lim	Nonlinear limiter function
limfix	Function used to print a warning message if a constant parameter is outside limits
limits	Function used to print a warning message if a parameter is outside limits
limstate_const	State variable limiting function with constant limits
limstate	State variable limiting function
movingavg*	Moving average filter implementation
output	Function used for generating output window messages
outfix	Function used for generating output window messages only at initialisation
picdro_const	Logical pick-up/drop-off function with constant pick-up/drop-off arguments
picdro	Logical pick-up/drop-off function
picontrol_const	Function used in IEEE Std 421.5-2005 Std. PI controller implementations with constant limits
picontrol	Function used in IEEE Std 421.5-2005 Std. PI controller implementations with variable limits
reset	Function used to reset state variables and constant internal variables
rms	Function returning the RMS/EMT simulation type
sapprox	Spline approximation based on a single dimensional array
sapprox2	Spline approximation based on a two dimensional array
select_const	Conditional function with constant true/false expressions
select	Conditional function
selfix_const	Conditional function applied based on constant boolean expression with constant true/false expressions
selfix	Conditional function applied based on constant boolean expression
time	Returns the current simulation time
vardef	Function assigning a text description and unit to a specific variable
file	Obsolete function
fault	Obsolete function
mavg	Obsolete function

Table 30.16.2: DSL Special Functions

---

\*Buffer based DSL function

- 
- Note:**
- The behaviour/availability of certain DSL special functions may be affected by the DSL level, as summarised in Table [30.5.1](#).
  - Names of DSL special functions are reserved. Note that DSL model variables may not have any reserved name.
- 

## aflipflop

[[back](#)]

### ***aflipflop (x, boolset, boolreset)***

Analog flip-flop function (with internal state and memory of input value).

**Arguments:**

- *x* - Input value to be stored
- *boolset* - Set expression to be evaluated as true or false. True is evaluated if *boolset*  $\geq 0.5$ . False is evaluated if *boolset*  $< 0.5$ .
- *boolreset* - Reset expression to be evaluated as true or false. True is evaluated if *boolreset*  $\geq 0.5$ . False is evaluated if *boolreset*  $< 0.5$ .

**Return value:**

- If *internal state*=0 then return the current value of *x*
- If *internal state*=1 then the value of *x* from the instant when *internal state* changed from 0 to 1.

The ***internal state*** will change during simulation as described below:

- changes from 0 to 1, if *boolset* is true and *boolreset* is false (SET)
- changes from 1 to 0, if *boolset* is false and *boolreset* is true (RESET)
- remains unaltered in all other situations. (HOLD)

## balanced

### ***balanced()***

This function provides the balanced/unbalanced network representation mode used during a dynamic simulation.

**Arguments:** none.

**Return value:** *yo* calculated as below:

$$yo = \begin{cases} 1 & \text{if RMS balanced simulation is executed} \\ 0 & \text{if RMS unbalanced or EMT simulation is executed} \end{cases}$$

## delay

[[back](#)]

### ***delay (x, Tdelay)***

Delay function. This is a buffer based DSL function. Stores the value of *x* at the current simulation time (*Tnow*) and returns the value of *x* at time *Tnow* – *Tdelay* where *Tdelay* must evaluate to a time-independent constant and may therefore only consist of constants and parameter variables. If it is smaller than the integration step size, the latter is used. The expression *x* may contain other functions.

**Arguments:**

- $x$  - input to be delayed
- $T_{delay}$  - Amount (in seconds) by which the output of the function is delayed w.r.t. the input

**Return value:** Returns (at simulation time instant  $T_{now}$ ) the value of  $x$  at time  $T_{now} - T_{delay}$ .

**Example:**

```
y = delay(yi, 1.0)
```

Delays input signal  $yi$  by one second.

---

**Note:** Using a time delay which is lower than the simulation time step range adopted during a simulation will lead to a warning message printed to the output window.

---

**event**

[\[back\]](#)

**event(enable, trigger, ConfigurationString)**

This function can create or call any kind of simulation relevant event. The event function is enabled if the value of  $enable$  is greater than or equal 0.5. If the function is enabled, an event is triggered each time the value of  $trigger$  crosses zero on a rising edge (changes sign from - to +).

**Arguments:**

- $enable$  - Enabling condition, if evaluates to true then function is enabled, if evaluates to false then function is disabled; if function is enabled, the event can be executed, depending on the trigger signal. The value of  $enable$  should be set constant throughout the simulation.
- $trigger$  - The  $trigger$  signal, which will enable the execution of the event at the instants when the sign of the  $trigger$  argument changes from - to +.
- $ConfigurationString$  - A text string that configures the event.

**Return value:** None.

The following configuration options are available:

**Option 1:** New event based on predefined/preconfigured event with optional specification of *variable* and *value* to change and optional specification of *dtime* delay

```
event(enable, trigger, 'name=ThisParameterEvent dtime=delay value=val variable=var')  
or  
event(enable, trigger, 'name=ThisEvent dtime=delay var_name=val')
```

**Option 2:** New parameter event with specification of target slot, variable and value to change, and optional *dtime* delay

```
event(enable, trigger, 'target=ThisSlot name=ThisParameterEvent dtime=delay value=val  
variable=var')
```

**Option 3:** New user-defined event type with specification of *target* slot, *variable* and *value* to change, and optional *dtime* delay

```
event(enable, trigger, 'create=EvtParam target=ThisSlot name=ThisEvent  
dtime=delay value=val variable=var')
```

or

```
event(enable, trigger, 'create=ThisEvtType target=ThisSlot name=ThisEvent  
dtime=delay var_name=val')
```

The *ConfigurationString* determines the details of the event call and which of the three options (described above) will apply:

- *string* ThisEvtType (mandatory, only option 3)  
Type of event to be created. To specify the type, use e.g. “EvtParam” for a parameter event or “EvtSwitch” for a switch event, etc. (see Section [29.6](#) for a list of available event types).
- *string* ThisSlot (mandatory, only options 2 and 3)  
If “target=this” is defined, the event is applied to a signal of the present DSL model. If any other name is given, the DSL interpreter checks the composite model where the present DSL model (*DSL Model*) is used and searches for a slot with the given name. The event is then applied to the element assigned to that slot. The *DSL Model* which calls the event has to be stored inside the composite model.
- *string* ThisEvent (mandatory)  
Name of the event created (option 3) or the external event to be started (option 1/2). The external event must be stored locally in the DSL model. If “name=this” is set, a parameter event will be created and executed automatically with the DSL element itself as the target.
- *string* ThisParameterEvent (mandatory)  
Name of the parameter event created (option 3) or the external parameter event to be started (option 1/2). The external event must be stored locally in the DSL model. If “name=this” is set, a parameter event will be created and executed automatically with the DSL element itself as the target.
- *double* delay (optional)  
Delay time of the event after triggering (in seconds).
- *double* val (optional)  
Value of the set parameter (only when “name=this” is set or when a parameter event is created).
- *string* var (optional)  
Variable name of the DSL model generating the event (if Option 1) or of the target element (Option 2 and 3) (can be a parameter, state variable, internal variable or not connected input signal) whose value will be set to “val” (only when “name=this” is set or when a parameter event is created).
- *string* var\_name (optional)  
Name of the event’s variable that is to be changed (e.g. refer to Example 4 further below). The parameter will be assigned the value “val”.

#### Remarks:

- For DSL level 6 or higher, if a DSL state variable or a constant internal variable needs to be changed, it is recommended to use **reset** function instead.
- If the event()-definition according to options 2/3 is used, the create and target parameters must be the first parameters of the *ConfigurationString*.
- The event command has changed from DSL Level 3 to DSL Level 4. With DSL Level 3 and smaller, a maximum number (MaxTrig) of trigger events was entered, event(MaxTrig, ...) (with MaxTrig being set to 0, unlimited trigger events may be produced), instead of a boolean expression for the condition, event(Condition, ...).
- The *ConfigurationString* contains assignments (e.g. ‘target=ThisSlot’ or ‘value=val’). The first term in an assignment (e.g. ‘**target**=ThisSlot’ or ‘**value**=val’) represents a fixed identifier which does not change (remains as specified above). The only exception is the

use of the 'var\_name=val' assignment, where the 'var\_name' identifier will be user-defined (refer to example 4 below). The string of the first assignment term may not contain spaces.

- The second term in a *ConfigurationString* assignment (e.g. 'target=**ThisSlot**' or 'value=**val**') represents user-defined strings being assigned to the specific identifiers. The following reserved keyword exists and can be used as described above: 'this'. The string of the second term in the assignment may not contain spaces. For example, if the name of a slot contains spaces, the following assignment is incorrect: 'target=Transformer Slot'. The correct approach is to remove the spaces existing in the name of the *Composite Model Frame* slot such that the string will not contain spaces e.g. 'target=Transformer\_Slot'.
- The order of the assignments within the *ConfigurationString* is important for the correct operation of the function and must be as specified above (refer to Options 1 - 3).

**Example 1:** The following example generates a **new event based on predefined/preconfigured event** named "OpenBreaker", which must be stored within the *DSL Model* containing the code snippet. The "OpenBreaker" event must be already configured: target object and action, i.e. close or open. A new event is triggered when *yo* changes sign from - to +. The delay time is 0.2s.

```
event(1,yo, 'name=OpenBreaker dtime=0.2')
```

**Example 2:** The example below shows a clock made with DSL using the second event option( , „name=this ...“) which automatically creates and configures a parameter event. The variable named *xclock* will be reset to value *val=0* within *dtime=0*, if the integrator output *xclock* is larger than 1. The input signal is a clock signal with the time period *Tclock*. The identifier "this" is used to apply the event on the same *DSL model* which issued this event.

```
inc(xclock)=0
inc(clockout)=0
xclock.=1/Tclock
reset_clock=select(xclock>1,1,-1)
event(enable,reset_clock, 'name=this value=0 variable=xclock')
clockout=xclock
```

**Example 3:** This example generates an event for the purpose of a simple under-voltage load-shedding relay. The element in the slot "Load" will be disconnected with a switch event "EvtSwitch", when the signal "u-umin" becomes positive. The event in the event list will be called "TripLoad". The default action of the EvtSwitch is "open", therefore no additional arguments are needed. For closing action the additional configuration "i\_switch=1" has to be added. The *DSL Model* has to be stored inside the composite model, which has the slot named "Load".

```
event(1,umin-u, 'create=EvtSwitch name=TripLoad target=Load')
```

**Example 4:** This example shows how an event is generated (which is not a parameter event) and one of its parameters is being changed. This is possible by adding to the *ConfigurationString* the event's parameter name followed by "=" and the assigned value afterwards. This example generates an event that will decrease the tap position, because the parameter *i\_tap* is set to 1. Note that the use of the assignment 'value=1 variable=i\_tap' is not allowed in this case, because 'value=' and 'variable=' are operating only for parameter events (EvtParam):

```
event(enable, trigger, 'create=EvtTap target=TransformerSlot
name=TapEvent_Decrease i_tap=1')
```

**flipflop**

[\[back\]](#)

### **flipflop (boolset, boolreset)**

Logical flip-flop function. Returns the internal logical state: 0 or 1.

#### **Arguments:**

- *boolset* - Set expression to be evaluated as true or false.

- *boolreset* - Reset expression to be evaluated as true or false.

**Return value:** the internal state, with initial value assigned as *boolset*. The *internal state* will change during simulation as described below:

- changes from 0 to 1, if *boolset* is true and *boolreset* is false (SET)
- changes from 1 to 0, if *boolset* is false and *boolreset* is true (RESET)
- remains unaltered in all other situations. (HOLD)

Note that the initial condition *boolset* = *boolreset* = 1 will cause an error message at initialisation.

## gradlim\_const

[\[back\]](#)

### gradlim\_const(*input,min,max*)

The function implements a gradient limiter of the *input* argument. This is a buffer based DSL function.

#### Arguments:

- *input* - input signal to be limited
- *min* - lower bound of gradient limit
- *max* - upper bound of gradient limit

**Return value:** *yo*, the gradient limited value of the *input*.

Example, where *yi* is the function's input signal and *yo* is the gradient limiter output:

```
yi = sin(2*pi() *50*time())
yo = gradlim_const(yi,-314.15,150)
```

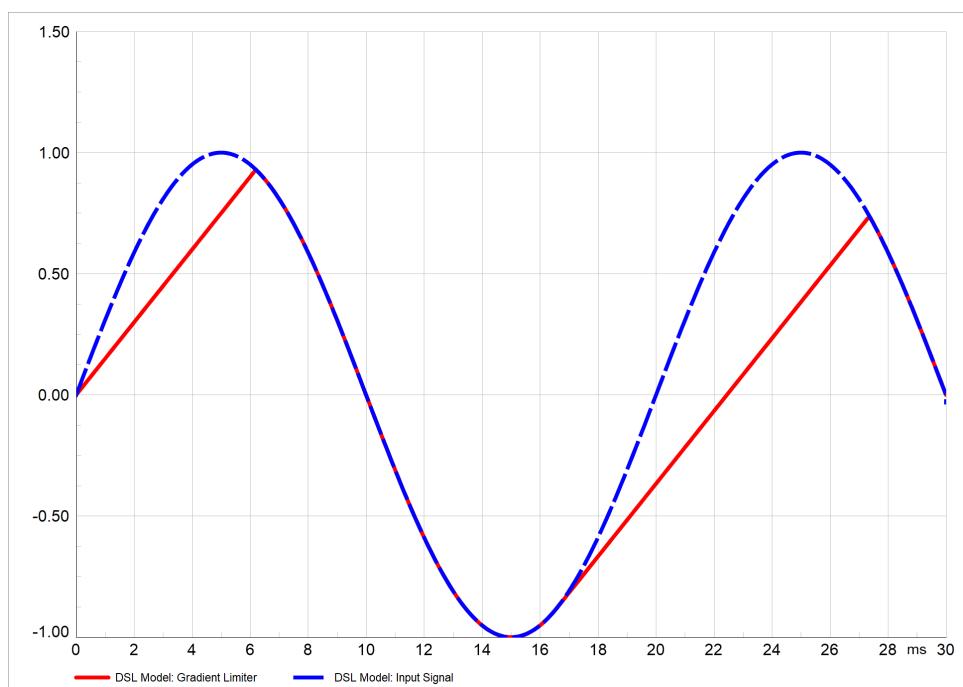


Figure 30.16.1: Gradient limiter example

**invlaprox**[\[back\]](#)

***invlaprox (y, array\_iiii)***  
***invlaprox (y, oarray\_iiii)***  
Inverse lapprox with array.

**Arguments:**

- *y* - Input argument for y-axis value of characteristic
- *array\_iiii* or *oarray\_iiii* - Function characteristic (array) containing two columns (one for x-axis and another one for y-axis). The *array\_iiii* is used if the lookup array is to be defined internally within the *DSL Model*. The *oarray\_iiii* is used if the lookup array is to be defined externally of the *DSL Model*, using *IntMat* objects.

**Return value:** *x*, calculated as the inverse of the linear approximation  $y=f(x)$ , where *f* is defined by the *array\_iiii/oarray\_iiii*. It should be noted that the lookup array must be sorted in ascending order and the y-values should be monotonic in order to avoid possibility of multiple solutions.

---

**Note:** DSL lookup Arrays/Matrices may be defined either directly within the *DSL Model* or externally, using *IntMat* objects. The procedure of declaring that a certain variable is a DSL array/matrix is done by naming the variable using a specific prefix, as described below:

- *array\_* - (e.g. “*array\_K*”), internal array, this option will create an array characteristic for a one dimensional function  $y=f(x)$ , where *x* and *y* characteristic value pairs are defined in two columns. The array characteristic will be possible to be defined within the *DSL Model* (n.b. not within the *DSL Model Type*) should at least one such variable name exist. Internal arrays may be defined in the tab “Advanced 1” of the “Basic Data” page of the *DSL Model* edit dialog. The length of the array is defined via the entry in the left cell of the first row. After adding the dimension an update is needed, this can be done by switching back to the “Basic Data” page and then again to the “Advanced 1” page.
- *matrix\_* - (e.g. “*matrix\_WindPowerCoefficient*”), internal matrix, this option will create a two-dimensional characteristic for a two-dimensional function  $y=f(xr, xc)$ , where *xr* and *xc* are the row and column arguments and *y* the function output value. The matrix characteristic is defined directly within the *DSL Model* (n.b. not within the *DSL Model Type*) should at least one such variable name exist. Internal matrices are defined in the tab “Advanced 2” of the “Basic Data” page of the *DSL Model* edit dialog. The dimension of the matrix is defined via the entry in the first two left cells of the first row. After adding the dimension an update is needed, this can be done by switching back to the “Basic Data” page and then again to the “Advanced 2” page.
- *oarray\_* - (e.g. “*oarray\_K*”), external array, this option will use an external array characteristic for a one dimensional function  $y=f(x)$ , where *x* and *y* characteristic value pairs are defined in two columns. The array characteristic must be defined within an external object of type *IntMat*. The *IntMat* object has to be stored inside the *DSL Model* (i.e. as child object). The object name must match the string following the *\_*. If this is done, then *PowerFactory* automatically assigns the array object to the variable *oarray\_....*. For example, if “*oarray\_K*” is declared in the *DSL Model Type* parameter list, then the *IntMat* object must be named “*K*” and stored inside the *DSL Model* which defines variable “*oarray\_K*”. A combination of internal and external arrays (i.e. mixing *array\_* and *oarray\_* variables) within a single *DSL Model Type* / *DSL Model* is not supported. The “*IntMat*” object must have proper dimensions, this means two columns and at least two rows for linear approximation based functions and at least three rows in case of spline approximation functions.
- *omatrix\_* - (e.g. “*omatrix\_WindPowerCoefficient*”), external matrix, this option will use an external matrix characteristic for a two-dimensional function  $y=f(xr, xc)$ , where *xr* and *xc* are the row and column arguments while *y* is the function output value. The matrix characteristic must be defined within an external object of type *IntMat*. The *IntMat* object has to be stored inside the *DSL Model* (i.e. as child object). The name has to match the string, following the *\_*. For example, in the case of “*omatrix\_WindPowerCoefficient*” *DSL* variable, the *IntMat* object would have to be named “*WindPowerCoefficient*” and stored inside the *DSL*

*Model.* If this is done, then *PowerFactory* automatically assigns the matrix object to the variable *omatrix\_....* A combination of internal and external arrays (i.e. mixing *matrix\_* and *omatrix\_* variables) within a single *DSL Model Type / DSL Model* is not supported. The “*IntMat*” object must have proper dimensions i.e. at least three columns and at least three rows for linear approximation based functions and at least four rows and four columns in case of spline approximation functions. The top row must contain the *xr* values, the left column must contain the *xc* values. Both rows and columns must be defined in ascending order for proper operation.

**lapprox**[\[back\]](#)***lapprox (x, array\_iiii)******lapprox (x, oarray\_iiii)***

Linear approximation function.

**Arguments:**

- *x* - Input argument for x-axis value of characteristic
- *array\_iiii* or *oarray\_iiii* - Function characteristic (array) containing two columns (one for x-axis and another one for y-axis). The *array\_iiii* is used if the array is to be defined internally within the DSL model. The *oarray\_iiii* is used if the array is to be defined externally of the DSL model, using *IntMat* objects. For more information on DSL lookup arrays and matrices, refer to the note within the function [invlapprox](#) description.

**Return value:** output *y*, calculated as the linear approximation  $y=f(x)$ , where *f* is defined by the *array\_iiii*. It should be noted that the array must be sorted in ascending order.

Example using internal array:

*y = lapprox(1.8, array\_valve)*

Example using external array:

*y = lapprox(1.8, oarray\_valve)***lapproxext**[\[back\]](#)***lapproxext (x, array\_iiii)******lapproxext (x, oarray\_iiii)***

Same function as the linear approximation *lapprox*  $y=f(x)$ , but instead to return a constant value in case *x* is lower than the first given point in the *array\_iiii/oarray\_iiii* returns a value calculated as follows:  $y = (y_{\text{val}}(\text{first point}) + x_e - x_{\text{val}}(\text{first point})) * dy/dx$  (the derivative of the first interval is used). The same logic is used if *x* is higher than the last point given in the lookup array. For more information on DSL lookup arrays and matrices, refer to the note within the function [invlapprox](#) description.

**lapprox2**[\[back\]](#)***lapprox2 (xr, xc, matrix\_iiii)******lapprox2 (xr, xc, omatrix\_iiii)***

Returns the linear approximation  $y=f(xr, xc)$  of a two-dimensional array, where *f* is defined by the *matrix\_iiii/omatrix\_iiii*. *xr* represents the row value and *xc* the column value of the lookup matrix. It should be noted that the axis must be sorted in ascending order.

**Arguments:**

- *xr* - Input argument for identifying the row position in the *matrix\_iiii*
- *xc* - Input argument for identifying the column position in the *matrix\_iiii*
- *matrix\_iiii* or *omatrix\_iiii* - Function characteristic (matrix) containing multiple rows and columns. The *matrix\_iiii* is used if the matrix is to be defined internally within the *DSL*

*Model.* The `omatrix_iiii` is used if the lookup array is to be defined externally of the *DSL Model*, using `IntMat` objects. For more information on DSL lookup arrays and matrices, refer to the note within the function `invlapprox` description.

**Return value:**  $y$  the linear approximation  $y=f(xr, xc)$  of a two-dimensional lookup array.

**Example:**

```
y = lapprox2(2.5, 3.7, matrix_cp)
```

**lastvalue**

[\[back\]](#)

**lastvalue(input)**

This function returns the value of the argument `input` at the last valid iteration (different time stamp). This is a buffer based DSL function.

**Arguments:**

- `input` - input argument

**Return value:** the value of the argument `input` at the last valid iteration

**lim\_const**

[\[back\]](#)

**lim\_const (x, min, max)**

Nonlinear limiter function that limits input `x` between `min` and `max`. Expressions `min` and `max` must evaluate to time-independent constants and may therefore only consist of constants and parameter variables. Its usage is encouraged (as opposed to `lim`) for simulation performance reasons whenever `min` and `max` are constants.

**Arguments:**

- `x` - quantity to be limited within the upper and lower bounds.
- `min` - lower bound limit
- `max` - upper bound limit

**Return value:**  $yo$  as defined below:

$$yo = \begin{cases} min, & \text{if } x < min. \\ max, & \text{if } x > max. \\ x, & \text{if } min \leq x \leq max. \end{cases}$$

**Example:**

```
lim_const(yi,YMIN,YMAX)
! limit variable yi between YMIN and YMAX
! with YMIN and YMAX being DSL parameters
```

**lim**

[\[back\]](#)

**lim (x, min, max)**

Nonlinear limiter function with definition as for `lim_const`, but with variable `min` and `max` limits.

**Example:**

```
lim(yi,yi_min,yi_max)
! limit variable yi between yi_min and yi_max
! yi_min and yi_max are DSL variables (e.g. limiting input signals of
a macro)
```

**limstate\_const**[\[back\]](#)***limstate\_const (x, min, max)***

Nonlinear limiter function for creating limited integrators. Argument *x* must be a state variable. Expressions *min* and *max* must evaluate to time-independent constants and may therefore only consist of constants and parameter variables. Its usage is encouraged (as opposed to [limstate](#)) for performance reasons whenever *min* and *max* are constants.

**Arguments:**

- *x* - state variable to be limited within the upper and lower bounds.
- *min* - lower bound limit (constant)
- *max* - upper bound limit (constant)

**Return value:** The value of the state variable *x*.

**Example:**

```
x1. = xe/Ti;
y = limstate_const(x1,X1MIN,X1MAX);
! with X1MIN and X1MAX being DSL parameters
```

**Remark:**

The [limstate](#) and [limstate\\_const](#) are allowed only once per state variable. Use internal variables in case the limited state is needed in several equations.

**limstate**[\[back\]](#)***limstate (x1, min, max)***

Nonlinear limiter function for creating limited integrators.

This function is similar to [limstate\\_const](#) but with variable *min* and *max* limits.

**Example:**

```
x1. = xe/Ti;
y = limstate(x1,xmin,xmax);
! with xmin and xmax being DSL variables
```

**Remark:** The [limstate](#) and [limstate\\_const](#) are allowed only once per state variable. Use internal variables in case the limited state is needed in several equations.

**limfix**[\[back\]](#)***limfix(param)=(min, max)***

Function used to print a warning message to the output window if a parameter is outside the specified limits at initialisation. Brackets [ and ] are used to indicate the inclusion of the end point in the allowed range; Parentheses ( and ) are used to indicate the exclusion of the end point from the allowed range. The use of this function is encouraged (as opposed to [limits](#)) for performance reasons whenever *param* is constant throughout the simulation or if the check should only be performed during the initialisation.

**Arguments:**

- *param* - parameter to be verified for upper and lower bounds.
- *min* - lower bound limit
- *max* - upper bound limit

**Return value:** None.

**Example:**

```
limfix(K)=(0,1] !to warn if parameter K is <=0 or >1 at initialisation
```

**limits**[\[back\]](#)**limits(param)=(min, max)**

Function used to print a warning message to the output window if a parameter is outside the specified limits throughout the simulation. Brackets [ and ] are used to indicate the inclusion of the end points in the range; ( and ) are used to indicate the exclusion of the end points from the range.

**Arguments:**

- *param* - parameter to be verified for upper and lower bounds.
- *min* - lower bound limit
- *max* - upper bound limit

**Return value:** None.

**Example:**

```
!To warn if xpos<=0 or if xpos>=1 throughout the simulation:
```

```
limits(xpos)=(0,1)
```

```
!To warn if xpos<=0 throughout the simulation:
```

```
limits(xpos)=(0, )
```

**movingavg**[\[back\]](#)**movingavg(input,Tdel,Tlength)**

This function allows the modelling of a moving average filter.

**Arguments:**

- *input* - input argument
- *Tdel* - Time delay, in seconds
- *Tlength* - Buffer length, in seconds

**Return value:** It returns the moving average value of the *input* over a given time window of duration *Tlength*. The window starts *Tdel + Tlength* seconds before the present time and ends *Tdel* seconds before the present time. Variables *Tdel* and *Tlength* must evaluate to time-independent constants and may therefore only consist of constants and parameter variables.

**output****output(boolexpr, message\_string)**

This function generates an output window message defined in the *message\_string* depending on the value of argument *boolexpr*. The value of *boolexpr* is evaluated at each time step during the simulation. The first time that this value is found to be true, the *message\_string* is processed and a message is sent to the output window. The *output* function is disabled afterwards until the DSL model is reset, to prevent recurring messages. The parameters of the message can be modified in the string by assigning a new value. The *message\_string* may contain variables and the special function num(*boolexpr*) or num(*expr*):

- Variable names which appear directly after an "=" sign will be substituted by their actual values; hence, the line of code below may generate the message:

```
maximum exceeded: yt=1.2 > ymax=1.0:  
output (yymax,'maximum exceeded: yt=yt > ymax=ymax')
```

- The num(expr) or num(booleanexpr) will be substituted with the calculated value of the expression, e.g.:
 

```
value=num(a+b) may produce value=3.5000  
value=num(a+b) may produce value=3.5000
```

**Arguments:**

- booleanexpr* - Expression to be evaluated as true or false.
- message\_string* - String delimited by single quotes

**Return value:** None.

## outfix

### **outfix(booleanexpr, message\_string)**

This function is a variant of the [output](#) function. It is evaluated only at initialisation. Its usage is encouraged for performance reasons whenever *booleanexpr* evaluates to a constant throughout the simulation.

**Arguments:**

- booleanexpr* - Expression to be evaluated as true or false.
- message\_string* - String delimited by single quotes

**Return value:** None.

## picdro\_const

[\[back\]](#)

### **picdro\_const(booleanexpr, Tpick, Tdrop)**

This function implements a logical pick-up-drop-off function commonly used when designing protection schemes (e.g. fault detection, signal out-of-range, etc.). Arguments *Tpick* and *Tdrop* must evaluate to time-independent constants and may therefore only consist of constants and parameter variables. Its usage is encouraged (as opposed to [picdro](#)) for simulation performance reasons whenever the *Tpick* and *Tdrop* arguments evaluate to time-independent constants.

**Arguments:**

- booleanexpr* - Expression to be evaluated as true or false.
- Tpick* - pick-up time delay in seconds (constant)
- Tdrop* - drop-off time delay in seconds (constant)

**Return value:** the internal logical state: 0 or 1. The state is evaluated as below:

- changes from 0 to 1, if *booleanexpr* = 1 (i.e. `true`), for a duration of at least *Tpick* seconds
- changes from 1 to 0, if *booleanexpr* = 0 (i.e. `false`), for a duration of at least *Tdrop* seconds
- remains unaltered in other situations.

**Example:**

```
picdro_const (u1-0.9 < 0, 0.005, 0.01)  
! detection of under-voltage operation mode
```

## picdro

[\[back\]](#)

### **picdro (booleanexpr, Tpick, Tdrop)**

Logical pick-up-drop-off function useful for relays. This function is similar to [picdro\\_const](#) but with

variable *Tpick* and *Tdrop* arguments.

**Arguments:**

- *boolexpr* - Expression to be evaluated as true or false.
- *Tpick* - pick-up time delay in seconds (may be variable)
- *Tdrop* - drop-off time delay in seconds (may be variable)

**Return value:** the internal logical state: 0 or 1. The state is evaluated as for [picdro\\_const](#).

**picontrol\_const**

[\[back\]](#)

**picontrol\_const(state,min,max,prop\_input)**

The function is commonly used when developing a standard parallel structure of a non-windup PI controller as shown in Figure 30.16.2 and described in IEEE 421.5 (2005) standard. In the figure, *K<sub>p</sub>* and *K<sub>i</sub>* are constant parameters and *input* is an input signal. Its usage is encouraged (as opposed to [picontrol](#)) for simulation performance reasons whenever the *min* and *max* arguments evaluate to time-independent constants.

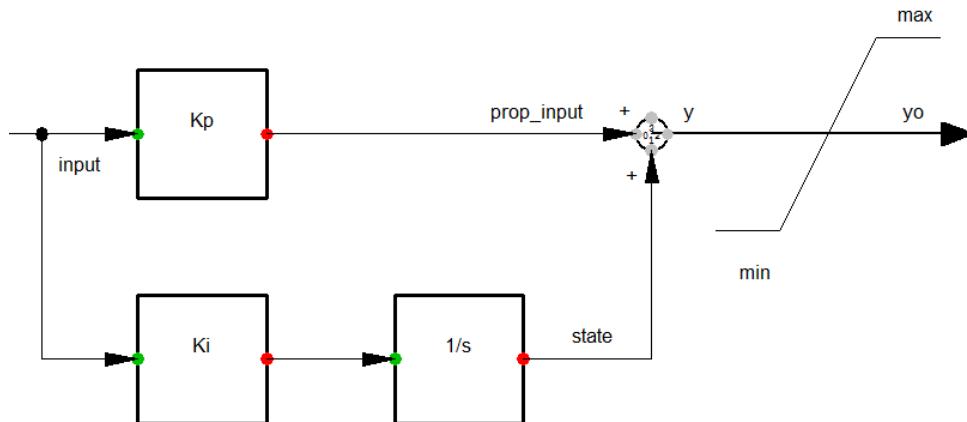


Figure 30.16.2: PI Controller Structure acc. to IEEE 421.5

**Arguments:**

- *state* - state variable of the integral part
- *min* - lower bound of controller output *yo*
- *max* - upper bound of controller output *yo*
- *prop\_input* - PI Controller proportional part (*input* \* *K<sub>p</sub>*)

**Return value:** *yo*, calculated as below:

$$yo = \begin{cases} \min, & \text{if } (\text{prop\_input} + \text{state}) < \min. \\ \max, & \text{if } (\text{prop\_input} + \text{state}) > \max. \\ \text{prop\_input} + \text{state}, & \text{otherwise.} \end{cases}$$

The argument `state` must be defined as a state variable, `min` and `max` arguments must evaluate to time-independent constants and may therefore only consist of constants and parameter variables.

Additionally, this function limits the `state` variable based on the following considerations:

- if  $(\text{prop\_input} + \text{state}) > \text{max}$  then `state` is frozen
- if  $(\text{prop\_input} + \text{state}) < \text{min}$  then `state` is frozen

To implement the PI controller shown in Figure 30.16.2, the following three DSL instructions need be applied:

```
prop_input=Kp*input
state.=Ki*input
yo = picontrol_const(state,min,max,prop_input)
```

## **picontrol**

[\[back\]](#)

### ***picontrol(state,min,max,prop\_input)***

The function is commonly used when developing a standard parallel structure of a non-windup PI controller as shown in Figure 30.16.2 and described in IEEE 421.5 (2005) standard. This function is similar to [picontrol\\_const](#) but with variable `min` and `max` arguments.

**Remark:** `picontrol` requires DSL Level 6 or higher.

## **reset**

[\[back\]](#)

### ***reset(var,rst,val)***

This function resets the variable `var` to the value `val` at specific instants during the simulation depending on argument `rst`. The event is triggered every time when the input signal `rst` crosses the value 0.5 upon a rising edge. The `reset` procedure (unlike the `event()` function) is regarded as being an internal DSL event, and so it only emits *Output Window* messages if the flag *Display internal DSL events* is checked in the *Run Simulation* command dialog. The arguments `var`, `rst` and `val` must refer to DSL model variable names declared within the DSL macro which calls the `reset` function (domain is the DSL macro namespace).

#### **Arguments:**

- `var` - DSL model variable that is reset. The variable can be a state variable or a constant internal variable (i.e. by constant it is understood that the internal variable has an initial condition statement only).
- `rst` - Variable based on which the reset event is triggered.
- `val` - Upon triggering the reset event, the resetted variable `var` will be assigned the value of argument `val`. Argument `val` can be a value, a DSL variable or a DSL function returning a value.

**Return value:** None.

#### **Examples:**

```
!Reset the state variable x_id to 0 when input voltage drops below 0.9 p.u.
x_id.=(id_ref-x_id)/Tid
reset(x_id, u<0.9, 0)
```

or

```
!Set the constant internal variable t_fault to be equal with the simulation
!time when the internal variable fault_detect is activated
inc(t_fault) = 0
fault_detect = picdro(u<UMIN,0.005,0.005)
reset(t_fault, fault_detect>0, time())
```

**Remarks:**

- *reset* function requires DSL Level 6 or higher (refer to Table 30.5.1 for more information). For lower levels use **event** function instead.
- It is allowed to use multiple *reset* functions targeting the same or different variables to be reset.
- If the *reset* function is defined within the *Equations* page of a DSL model (i.e. not of a macro), then the variable names are the DSL model variables (domain is the DSL model namespace).

**rms**

[\[back\]](#)

***rms()***

Function retrieving the type of simulation being executed.

**Arguments:** none.

**Return value:** *yo* representing the dynamic simulation type:

$$yo = \begin{cases} 1 & \text{for RMS simulation (balanced/unbalanced)} \\ 0 & \text{for EMT simulation} \end{cases}$$

**sapprox**

[\[back\]](#)

***sapprox (x, array\_iiii)***  
***sapprox (x, oarray\_iiii)***

Returns the spline approximation  $y=f(x)$ , where  $f$  is defined by the *array\_iiii/oarray\_iiii*. It should be noted that the array must be sorted in ascending order.

**Arguments:**

- *x* - Input argument for x-axis value of characteristic
- *array\_iiii* or *oarray\_iiii* - Function characteristic (array) containing two columns (one for x-axis and another one for y-axis). For more information on DSL lookup arrays and matrices, refer to the note within the function **invlapprox** description.

**Return value:** *y*, the spline approximation of  $y=f(x)$

**Example:**

```
y = sapprox(1.8, array_valve) or y = sapprox(1.8, oarray_valve)
```

---

**Note:** The spline approximation should be used with care due to the risk of overshooting values.

---

**sapprox2**

[\[back\]](#)

***sapprox2 (xr, xc, matrix\_iiii)***  
***sapprox2 (xr, xc, omatrix\_iiii)***

Returns the spline approximation  $y=f(xr,xc)$  of a two-dimensional array, where  $f$  is defined by the *matrix\_iiii/omatrix\_iiii*. *xr* represents the row value and *xc* the column of the matrix.

**Arguments:**

- *xr* - Input argument for identifying the row position in the *matrix\_iiii*
- *xc* - Input argument for identifying the column position in the *matrix\_iiii*
- *matrix\_iiii* or *omatrix\_iiii* - Function characteristic (matrix) containing multiple rows and columns. The *matrix\_iiii* is used if the matrix is to be defined internally within the *DSL Model*. The *omatrix\_iiii* is used if the array is to be defined externally of the *DSL model*, using *IntMat* objects. For more information on DSL arrays and matrices, refer to the note within the function [invlapprox](#) description.

**Return value:** *y*, the spline approximation of  $y=f(xr, xc)$ .

**Example:**

```
y = sapprox2(2.5, 3.7, matrix_cp)
```

---

**Note:** The spline approximation should be used with care due to the risk of overshooting values!

---

## **select\_const**

[\[back\]](#)

### **select\_const(*boolexpr*, *x*, *y*)**

Switch function used to output either argument *x* or *y* depending on the evaluation of argument *boolexpr*. Arguments *x* and *y* must evaluate to time-independent constants and may therefore only consist of constants and parameter variables. Its usage is encouraged (as opposed to [select](#)) for performance reasons whenever *x* and *y* are constants.

**Arguments:**

- *boolexpr* - Expression to be evaluated as true or false.
- *x* - argument for true branch evaluation (constants)
- *y* - argument for false branch evaluation (constants)

**Return value:** *x* if *boolexpr* is true, otherwise *y*.

**Example:**

```
x1.=select_const(input>threshold, -1, 1)
!  return -1 if input strictly greater than threshold
!  return 1 if input smaller than or equal with threshold
!  (evaluation of input and threshold as double numbers)
```

## **select**

[\[back\]](#)

### **select (*boolexpr*, *x*, *y*)**

Switch function used to output either argument *x* or *y* depending on the evaluation of argument *boolexpr*. The arguments *x* and *y* are allowed to be variable.

**Arguments:**

- *boolexpr* - Expression to be evaluated as true or false.
- *x* - argument for true branch evaluation (may be variable)
- *y* - argument for false branch evaluation (may be variable)

**Return value:** *x* if *boolexpr* is true, otherwise *y*.

**Example:**

```
x1.=select(xe1>xe2, xe1, xe2) !to select the biggest derivative
```

## **selfix\_const**

[\[back\]](#)

**selfix\_const(boolean,x,y)**

Fixed switch function, returning *x* throughout the simulation if *boolean* is true at initialisation, otherwise returns *y*. Arguments *x* and *y* must evaluate to time-independent constants and may therefore only consist of constants and parameter variables. Its usage is encouraged (as opposed to [selfix](#)) for simulation performance reasons whenever the *x* and *y* arguments evaluate to time-independent constants.

**Arguments:**

- *boolean* - Expression to be evaluated as true or false.
- *x* - argument for true branch evaluation (constants)
- *y* - argument for false branch evaluation (constants)

**Return value:** *x* throughout the simulation if *boolean* is true at initialisation, otherwise *y*.

**Example:**

```
xramp.=selfix_const(T1>0, 1/T1, 0.0) !to avoid division by zero
```

**selfix**[\[back\]](#)**selfix (boolean, x, y)**

Fixed switch function, returning *x* throughout the simulation if *boolean* is true at initialisation, else *y*. The arguments *x* and *y* are allowed to be variable.

**Arguments:**

- *boolean* - Expression to be evaluated as true or false.
- *x* - argument for true branch evaluation (may be variable)
- *y* - argument for false branch evaluation (may be variable)

**Return value:** *x* throughout the simulation if *boolean* is true at initialisation, otherwise *y*.

**Example:**

```
x1.=selfix(T1>0, xe/T1, 0.0) !to avoid division by zero
```

**time****time ()**

Function that retrieves the current simulation time.

**Arguments:** none.

**Return value:** *yo*, the current simulation time.

**Example:**

```
t=time()  
y = sin(t) or  
y = sin(time())
```

**vardef**[\[back\]](#)**vardef(var)**

Function which assigns to argument variable *var* a text description and a unit. This function may be applied only once to a certain variable, multiple use of vardef() statements on the same variable will lead to error.

**Arguments:**

- *var* - variable whose description and unit are assigned. The variable *var* must be a variable which has been already declared within the *DSL Model Type*.

**Return value:** None.

**Example:**

```
vardef(u1)='p.u.';'Positive sequence voltage'
```

---

**Note:** The function can be used on parameters, input and output signals, internal variables and state variables. The unit and description strings are used in *PowerFactory* for various presentation purposes (within plots, for data export) whenever the corresponding variable is recorded.

---

#### file

**file (ascii-parm, expr)**

**OBSOLETE!** Use an *ElmFile* object in the composite model instead.

#### fault

**OBSOLETE!** Use [event](#) function instead.

#### mavg

**OBSOLETE!** Use [delay](#) function instead.

### 30.16.3 DSL Global Library Macros

Refer to the [DSL Macros Documentation](#) for a description of the global library macros. The *DSL Macros Documentation* is also accessible selecting *Help → Technical References → DSL Macros* from the main menu.

# Chapter 31

## System Parameter Identification

### 31.1 Introduction

The process of parameter identification for power system elements for which certain measurements have been made is performed with the *System Parameter Identification* function using the icon .

The *ComIdent* command object is a high performance non-linear optimisation tool, which is capable of a multi parameter identification for one or more models, given a set of measured input and output signals. This identification is principally performed in the following way:

- A *Measurement File* object (*ElmFile*) is created which maps the raw measured data onto one or more *measurement signals*. These signals may contain measurements from the network, from elements or response signals. The *measured* data can be either a text file or a result object (*ElmRes*) from another simulation.
- The measurement signals can be used as inputs by models of the power system elements for which one or more parameters have to be identified, or they may be used to control voltage or current sources. It is also possible to excite the system with simulation events (such as parameter, switch or short circuit events).
- The output signals of the power system elements are fed into a comparator together with the corresponding measured signals. The comparator (*ElmDiff*) is thus given the measured response on the excitation and the simulated response of the element models.
- The comparator calculates an objective function, which is the weighted sum of the differences between the measured and the simulated response, raised to a whole power (by default to the power of 2).
- The *ComIdent* command will collect all objective functions from all comparator objects in the currently active study case and will minimise the resulting overall objective function. To do this, the *ComIdent* command is given a list of parameters which are to be identified. The objective functions are minimised by altering these parameters.

This whole process is visualised in Figure 31.1.1.

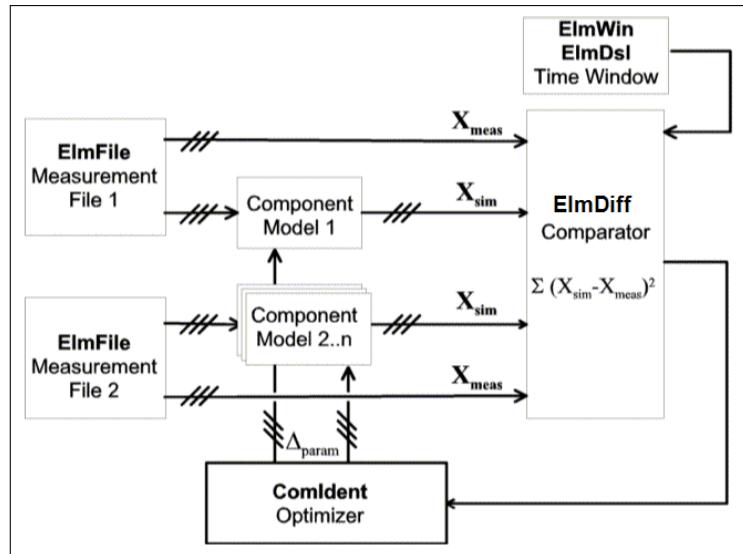


Figure 31.1.1: The identification principle

Of course, Figure 31.1.1 only visualises the principle of the identification. All details of the *PowerFactory* identification functions are described in the following sections. Some hints on selecting the proper algorithm are given in Section 31.4. Advice in case of stagnation can be found in Section 31.5.

## 31.2 Performing a Parameter Identification

For performing a parameter identification there are three main components needed:

- The parameter identification command
- The controls, for defining the parameters which should be optimised
- The comparison object, for calculating the objective function which should be minimised

These major components are described in the following sections. First of all the interface of the parameter identification function is described.

The identification process is executed by the *ComIdent* command. This command can be opened by the icon on the main menu. This icon can be found on the *RMS/EMT Simulation* and *Quasi-Dynamic Simulation* toolbar which is accessible by selecting the *Change Toolbox* icon (.

### 31.2.1 Basic Options

The *Basic Options* page allows the selection of the *Identification type*; there are currently three types supported:

- Load Flow
- Simulation (RMS/EMT)
- Quasi-Dynamic Simulation

The *Load Flow* option allows the optimisation of load flow relevant parameters based on load flow calculations. If this option is selected then the *Optimised calculation* will point to the load flow command from the study case. The *Additional command* will be blank.

The *Simulation* option allows the optimisation of any simulation relevant parameter. The simulation method used can be either an RMS- or an EMT simulation. In this case the *Optimised calculation* will point to the *Calculation of initial conditions (ComInc)* dialog from the active study case. The *Additional command* will point to the *Run Simulation (ComSim)* command from the active study case. The commands can be accessed by pressing the arrow button (→).

The *Quasi-Dynamic Simulation* option allows the optimisation of relevant parameters based on quasi-dynamic simulation. If this option is selected, then the *Optimised calculation* will point to the quasi-dynamic simulation command from the study case. The *Additional command* will be blank.

Also on the *Basic Options* page is the selection of the *Optimisation method* offered. The following algorithms can be selected, described in detail in Section 31.3:

- Particle Swarm Optimisation
- Nelder Mead
- DIRECT (dividing rectangles)
- BFGS
- Legacy (Quasi-Newton)

Below this selection it can be configured whether the modifications done by the *parameter identification command* should be recorded in the existing network (with its active variations and operation scenarios) or in a new variation. It is important to note that the *parameter identification command* can modify any numerical value in the active project. This includes also type data which is out of the range of the variation recording (as long as the type is not stored in the grid). Using the option *Create new Variation* can fail to record the modification if an operation scenario was active and the changed parameter belongs to the operational data!

Finally on the *Basic Options* page, the *Maximum number of iterations* and the *Maximum number of objective function evaluations* (not for the Legacy method) can be entered. These are the two major stopping criteria. A refined configuration for the stopping criteria can be done on the *Stopping criteria* page, as described in Section 31.2.8. The parameters are also described in this section.

### 31.2.2 Controls / Compared Signals

The page *Controls / Compared Signals* offers access to the controls and the comparison object(s). The *controls* define which parameter from which object will be identified. The *comparison object* calculates the objective function signal which will then be minimised by the *ComIdent*.

#### Control:

A new *Control* can be created by pressing the button **Add new control**. This will then show a new control of the type *IntIdentctrl*.

At first an *Element* has to be selected. This can be any object which has an impact on the calculation results (depending on the selected calculation type either load flow, dynamic RMS/EMT simulation or quasi-dynamic simulation).

After this the name of the *parameter* has to be entered. For some object classes (such as ElmDsl) a drop down list with the available parameters is displayed; for other elements the parameter name has to be entered manually. The parameter name can be found by opening the object and then hovering the mouse pointer over the input field. The parameter name will be shown in a balloon text. After entering the correct parameter name the *actual value* of this parameter is displayed below the input field.

The optimisation can be improved by entering *Constraints*. After ticking the checkboxes the *Lower bound* can be entered, which has to be smaller than the initial value of the selected parameter. The *Upper bound* has to be larger than the initial value of the selected parameter.

Some of the algorithms can make use of a *Mesh size definition* (see Section 31.3 for algorithm details). Select for this the option *User defined mesh size* and enter a proper value for the *Mesh size*.

Each *control* has also the *out of service* option. The control will be ignored if this checkbox is selected.

All defined *Controls* can be displayed by pressing the button **Show and edit controls** in the *ComIdent*. This will display a compact list with all existing *Controls*. The data can be modified like in the *Data Manager* by selecting whole columns. New *Control* objects can be defined by pressing the “New Object” button .

#### **Comparison Object:**

The objective function value (i.e. the cumulated difference between measured data and simulated data) is calculated inside a *Comparison object* of the class *ElmDiff*. At least one *comparison object* is required for the execution of the *System Parameter Identification* command. A new *ElmDiff* can be created by pressing the button **Create new comparison object**. This will open a selection window for the location. An active grid (or any sub folder) has to be selected as location. The *System Parameter Identification* command will not consider the *ElmDiff* if it is stored outside the active grid!

The comparison object can be set out of service on the *Basic Options* page. Also an exponent can be entered to weight the objective function. The *Simulation* page allows to enter pairs of signals which will be used for the objective function calculation. A new pair can be added by right-clicking into free space and then selecting *Append Row(s)*. In each row two elements have to be selected. For each element the signal has to be defined which is used for comparison. The signal cell will show a drop down list after selecting the element. For some elements it is possible to manually overwrite the signal name (this may be necessary if the proper signal was not displayed in the drop down field). After defining a signal pair, which consists usually of a measured and a simulated signal, a *weight* has to be entered into the last column. This allows an individual weighting of the different signals. Entering a zero will exclude this signal pair from the objective function calculation.

In the lower section of the *Simulation* page a *Weighting signal* can be connected. This will allow a time dependent weighting of the objective function signal. Via this it is possible to exclude sections of the simulation or to emphasise other parts. Select for this an *Element* which has the weighting signal as output. This is typically either a custom *DSL Model* or a measurement file object (*ElmFile*). Then enter the name of the *Signal*. The measurement file object can either connect measured data (ComTrade or measurement file format) or it can read from another result object (*ElmRes*).

The option *Backward initialisation of signals* can help to solve initialisation issues if the simulation model requires the compared signal for initialisation. In this case the simulated signal has to be connected as Element 2 / Signal 2. With the option *Backward initialisation of signals* enabled the initial value from *Signal 1* will be available on *Signal 2* for initialisation.

The *Load Flow* page of the *comparison object* is very similar to the *Simulation* page. The major difference is that the load flow identification has no time relation. Therefore there is also - beside the definition via signals in the lower section - the possibility to directly enter a *Comparison with reference value*. For this only the simulated signal has to be selected via *Element* and then *Signal*. The measured value can be directly entered as *Reference value*.

It is important to note that only some elements which are considered for load flow calculation should be selected on this page. Selecting a common model will result in an error. The measurement file object (*ElmFile*) can also be used; remember in this case to enter a proper *Load Flow Time* in the measurement file element on the load flow page.

The *Load Flow* page of the *comparison object* is also used for the *quasi-dynamic simulation*. Hence, it is possible to compare signals of elements either to a constant reference value or to time dependent signals, e.g. a measurement file object (*ElmFile*) as described above.

The objective function  $e$  for *load flow* is calculated as the sum of powers of the absolute errors:

$$e = \sum_{i=1}^n |(m_i - c_i)w_i|^p. \quad (31.1)$$

The objective function  $e$  for *simulation* (RMS/EMT and quasi-dynamic) is calculated as the time integral over the sum of powers of the absolute errors:

$$e = \int_{t0}^{tmax} \sum_{i=1}^n |(m_i(t) - c_i(t))w_i|^p dt. \quad (31.2)$$

where

- $m_i$  is the measured response (e.g. *Signal 1*)
- $c_i$  the simulated response (e.g. *Signal 2*)
- $w_i$  is the weighting factor (i.e. for the difference signal nr. 1)
- $p$  is the power

---

**Note:** Optimisation steps with a simulation that is interrupted by an error or that leads to negative results for the objective function are considered to be infinitely bad. This can be used to penalise unwanted solutions by interrupting the simulation with an error thrown by a simulation scan as described in Chapter 29.9. Typical applications are for example FRT violations, pole slip or convergence issues (see also the FAQ: [How can I automatically stop a divergent simulation?](#)). Such results often should be disregarded and the computational effort of the *System Parameter Identification* can be reduced.

---

### 31.2.3 PSO (Particle Swarm Optimisation)

The options on this page will be displayed if the proper *Optimisation Method* was selected on the *Basic Options* page. For details about the algorithm and the associated options see Section 31.3.2.

### 31.2.4 Nelder Mead

The options on this page will be displayed if the proper *Optimisation Method* was selected on the *Basic Options* page. For details about the algorithm and the associated options see Section 31.3.3.

### 31.2.5 DIRECT (DIviding RECTangle)

The options on this page will be displayed if the proper *Optimisation Method* was selected on the *Basic Options* page. For details about the algorithm and the associated options see Section 31.3.4.

### 31.2.6 Random Number Generation

The options on this page will be displayed if the proper *Optimisation Method* was selected on the *Basic Options* page.

**Algorithm:** Where needed, random numbers are generated using a pseudorandom number generator. It accepts a so-called seed. This is an integer used to initialise the internal state of the generator. Same

seeds lead to the same pseudorandom number sequences the generator produces. Therefore, a seed should be set if one wants reproducible results.

**Parameters:**

**Seeding type:** Two options: automatic and user defined. If automatic, a random seed will be drawn. If user defined is selected the number given in the following parameter will be used to seed the random number generator.

**Seed:** Used as a seed for the random number generator if user defined. Should be set if results must be reproducible.

### 31.2.7 Gradient Calculation

The option on this page will be displayed if a gradient based *Optimisation Method* was selected on the *Basic Options* page.

**Algorithm:** Let  $e_i \in \mathbb{R}^d$  be the i-th canonical base vector of  $R^d$ , i.e.  $e_1 = (1, 0, 0, \dots)$ ,  $e_2 = (0, 1, 0, \dots)$ , ...

#### 31.2.7.1 Forward Differences

Partial derivatives are calculated by the formula

$$\partial_i f(x) \approx \frac{f(x + \delta e_i) - f(x)}{\delta} \quad (31.3)$$

with a small  $\delta > 0$ .

For calculation of the gradient, this requires  $d + 1$  function evaluations.

#### 31.2.7.2 Centered Differences

Partial derivatives are calculated by the formula

$$\partial_i f(x) \approx \frac{f(x + \delta e_i) - f(x - \delta e_i)}{2\delta}. \quad (31.4)$$

This is more accurate than the forward differences, but also needs more function evaluations: For calculation of the gradient, this requires  $2d$  function evaluations.

**Parameters:**

*Method for the numeric calculation of gradient* (on page Gradient calculation): Forward or Centered Differences according to the above description.

### 31.2.8 Stopping Criteria

**Algorithm:** The solvers will stop their iterations as soon as one of the stop criteria is satisfied. “As soon as” thereby means “at the end of an iteration, as soon as they are satisfied”.

**Parameters:** The following parameters for stopping criteria can be set for all solvers.

**Maximum number of iterations:** Self explaining. For the Legacy solver, this equals the maximum number of function evaluations.

**Maximum number of objective function evaluations:** The most important parameter for how long the solver will run. It translates much more directly to execution time than the maximum number of

iterations because the number of evaluations made in one iteration differs a lot, depending on the solver and/or dimension.

**Target objective value:** If the solver finds an objective value  $\leq$  this threshold, it will stop and return this value as the optimum.

**Minimal improvement in given number of iterations:** Requirements for the minimum progress the solver has to make may be defined here.

**Number of iterations** The number of iterations considered for the two following options may be specified here: Let  $N$  denote this number of iterations.

**Minimal absolute improvement of objective value:** Min. abs. improvement of values since  $N$  iterations.

**Minimal relative improvement towards target objective value:** Similar to the previous one, but relative: Since  $N$  iterations, the distance to the target value specified in the parameter *Target objective value* must have decreased by this fraction.

### 31.2.9 Output

The *Output* page of the *ComIdent* allows a fine tuning of the information which will be written to the output window during the execution. It enables the user to configure recording in a result object (class *ElmRes*). The output window information can be controlled via the *output per iteration* options:

- Off: No information will be written to the output window
- Short: With this option, only the current iteration number together with the objective value (summed output of all *ElmDiff*) will be written to the output window.
- Detailed: With this option, the output of the calculation command used (load flow or dynamic simulation) will also be written into the output window. This can result in a lot of data, especially if the simulation is running into convergence issues.

For the second and third choice is also the option *output choice of parameters* available. If this option is ticked then the currently used set of parameters (see Section 31.2.2) is also printed to the output window.

The option *record iterations* will create a new result object (class *ElmRes*) with the name *Parameter Identification* which is located in the currently active study case. The result object will contain one row per iteration of the *ComIdent*. The columns will contain the following data for the *ComIdent*:

- Evaluation (b:eval)
- Iteration (b:iter)
- Failed calculation (b:failedCalc)
- Objective function value (b:objectiveValue)
- Best objective function value (b:bestObjective)
- Iteration with best objective value (b:iterBestObjective)
- Evaluation with best objective value (b:evalBestObjective)

The result object will also record the parameters used for each evaluation.

The recorded data can either be used for showing the convergence of the identification process in a plot (select for this the result object in the first column in the plot configuration) or it can be used for further scripted analysis. Exporting the data via the *ComRes* is of course also supported.

## 31.3 Algorithms

### 31.3.1 Problem Description

The goal of parameter identification is to find (global) minima of functions  $f : \Omega \rightarrow \mathbb{R}$  under the following circumstances:

- The domain  $\Omega$  of  $f$  is a bounded or unbounded box  $\mathbb{R}^d \supseteq \Omega = R^d \cap \prod_{i=1}^d [a_i, b_i]$  with  $-\infty \leq a_i < b_i \leq +\infty$ .
- $f$  can be evaluated in arbitrary points  $x \in \Omega$ .
- No properties of  $f$  are known, including information about continuity and differentiability, convexity, number of local minima, etc. In particular, the gradient of  $f$  is not available analytically.

User input:

- A starting point  $x_0 \in \Omega$
- Optional: a “suggested mesh size” (one per dimension). It can be seen as the minimum resolution at which one sees all relevant features of the function, or the inverse of the factor that the dimension has to be multiplied with in order to have “normal” length scales.

Some solvers may rely on properties like differentiability of  $f$ ; however, these are not guaranteed.

### 31.3.2 Particle Swarm Optimisation (PSO)

Particle Swarm Optimisation uses a set of particles evaluating the objective function while moving around in the search space. The movement of the particles is influenced by

1. their current velocity,
2. their memory of the best location they have found so far,
3. knowledge about good locations found by other particles they are in contact with.

These three criteria are weighted (including some randomness of the weights). There exist various communication patterns for point 3 leading to the different topologies supported in *PowerFactory*. Additionally, special events are supported, such as beaming of particles.

**Algorithm:** Particle Swarm Optimisation operates with a cloud of points - the particles - that are scattered over the search space and move in it as a swarm, i.e. they communicate with each other about where to go in order to find small function values.

**Nomenclature:** Let  $x^{glob}$  be the location where the so far best function value was seen. Let accordingly  $x_i^{pers}$  be the best value that particle  $i$  has seen by itself, and  $x_i^{loc}$  be the best location that the particles, with which particle  $i$  communicates, have seen. Depending on the exact variant of the algorithm, this can be different groups of particles, ranging from a few to all other particles (in which case these are equal to the global best location). Let finally  $x_i^{cur}$  be the current position of particle  $i$ , and  $v_i^{cur}$  its velocity, i.e. the displacement since the previous iteration.

#### The algorithm:

**Initialise** the swarm, then iterate the following loop until stopped:

1. Particles **communicate** with each other, i.e. their information about objective function values of neighbours are updated according to the communication strategy used.
2. The particles **move** to their next locations, i.e. the  $x_i^{cur}$  and  $v_i^{cur}$  are updated.

3. **Evaluation** of the objective function at the new particle locations.
4. One or more **special events** may be triggered.

**Communication strategies:**

- **Ring:** Each particle has two neighbours such that the communication topology becomes cyclic.
- **Stochastic Star:** Every particle communicates with every other particle. At each iteration, each particle decides randomly and independently, for which dimensions it wants to "listen".
- **Communication via reference set:** A reference set of locations is maintained. Particles communicate via the reference set.

**Moving strategies:**

- Standard: Let  $w^{momentum}$ ,  $w^{pers}$ ,  $w^{loc}$  and  $w^{glob}$  be different weights. Then the update of the next position of the particles is given by:

$$\begin{aligned}x_i^{cur} = & x_i^{cur} + w^{momentum} * u_1^i \cdot v_i^{cur} \\& + w^{pers} * u_2^i \cdot (x_i^{pers} - x_i^{cur}) \\& + w^{loc} * u_3^i \cdot (x_i^{loc} - x_i^{cur}) \\& + w^{glob} * u_4^i \cdot (x_i^{glob} - x_i^{cur}),\end{aligned}$$

for all particles  $i$ , where  $u_1$  to  $u_4$  are random numbers drawn uniformly from the unit interval (0,1).

- Cyber Swarm: Similar to the standard moving strategy. However, a reference set of locations is maintained and this information is used for the update of the locations of the moving particles.

**Special events:** Beam particles to randomly chosen new locations if some given criteria are satisfied.

**Parameters:**

**PSO Variant** Three main types are available:

- **Ring:**
  - Ring communication strategy
  - Standard moving strategy
- **Stochastic Star:**
  - Stochastic star communication strategy
  - Standard moving strategy
- **Cyber Swarm:**
  - Communication via reference set
  - Cyber swarm moving strategy

**Number of particles** The number of particles may be automatic or user-defined. In case of "Automatic", the algorithm calculates a reasonable number of particles for the problem under consideration. User defined can be any number  $\geq 2$ , but numbers smaller than 10 or larger than 60 should not be chosen without a special reason.

**Movement of particles** The following parameters define the weights for the moving strategies.

**Weight for momentum** The weight for the momentum, i.e. the current velocity, in the moving step. It is best to be chosen between 0 and 1.

**Weight for personal best result** The weight for the personal best. If modified it should best be not larger than 3.

**Weight for local best** The weight for the local best locations, which are the main guiding solutions in the Ring and the Stochastic Star variant. If modified it may be chosen best not larger than 5.

**Weight for global best** (Only for the Cyber Swarm variant) The weight for the global best solution. This is the best member of the reference set. Suitable choices like for Weight for local best.

**Beaming strategy** Settings for the beaming events:

**Beam particle after n iterations without improvement.**  $n =$  Maximum number of successive iterations in which a particle doesn't improve its personal best, until it is beamed to another location.

**Beam all after m iterations without improvement.**  $m =$  Maximum number of successive iterations in which the global best value ever seen by the whole swarm -  $f^{glob}$  - is not improved. After this, the whole swarm is scattered again around in the search space by beaming all particles to new locations.

**Number of evaluations for the exploration of valleys** If  $> 0$ , the beaming trajectory is scanned for possible still unexplored side valleys, and this number of function evaluations is used to try to walk further down into these side valleys. A number of 0 here means that the particles will just be beamed to their new locations.

**Cyber Swarm Configuration** Two parameters to modify the maintenance of the reference set for the cyber swarm algorithm.

**Weight for quality in reference set** This is the weight for the criterion "quality", i.e. low function values, for the maintenance of the reference set. The other criterion is "diversity", i.e. distance between the members of the reference set.

**Cardinality of reference set** The size of the reference set.

### 31.3.3 Nelder Mead

**Algorithm:** First, the initial simplex is constructed around the starting point, resulting in  $d + 1$  points called the vertices of the simplex. The objective function is evaluated at the points, i.e. at the vertices of the simplex. Then the iterations are done as follows:

1. Check if simplex is collapsed and has to be restarted or if a linesearch has to be performed.
2. Sort the values of the simplex vertices.
3. Calculate the gravitational centre of the face opposite to the worst point (which is a  $(d - 1)$ -dimensional simplex itself). Call it  $x_{opp}$ .
4. Reflect the worst point at  $x_{opp}$  and evaluate:

$$x_r := x_{opp} + \varrho \cdot (x_{opp} - x_{worst}); \quad f_r := f(x_r). \quad (31.5)$$

$\varrho$  is a fixed value.

5. If  $f_r$  is better than the best vertex of the simplex, expand further -

$$x_e := x_{opp} + \xi \cdot \varrho \cdot (x_{opp} - x_{worst}); \quad f_e := f(x_e) \quad (31.6)$$

with  $\xi$  some given value - and replace the previously worst vertex by the better point among  $\{x_r, x_e\}$  and go to step 1.

6. If  $f_r$  was still better than the second best vertex replace the previously worst vertex by it and go to step 1.
7. if  $f_r$  was at least better than the previously worst vertex, calculate an outer reflection point -

$$x_{out} := x_{opp} + \gamma \cdot (x_{opp} - x_{worst}); \quad f_{out} := f(x_{out}) \quad (31.7)$$

with  $\gamma$  some given value - and replace the previously worst vertex by the better point among  $\{x_r, x_{out}\}$  and go to step 1.

8. If  $f_r$  was not even better than the previously worst vertex, calculate an inner reflection point -

$$x_{in} := x_{opp} + \gamma \cdot (x_{worst} - x_{opp}); \quad f_{in} := f(x_{in}). \quad (31.8)$$

If now at least this brought an improvement in comparison to the worst vertex, replace the latter one by  $x_{in}$  and go to step 1.

9. If not, shrink the simplex towards its best vertex, i.e. vertex  $x_i$  is replaced with  $x_{best} + \sigma \cdot (x_i - x_{best})$  with  $\sigma$  some given value.

**Behaviour in case of simplex collapse:** If the simplex shrinks too much, it may be restarted. For restart its extents are multiplied by a factor. It may additionally be rotated in a uniformly random fashion.

**Linesearches:** Linesearches may be triggered when specific conditions are met. The direction of the linesearch has two contributions:

1. Gradient of the simplex.
2. The displacements from one point in the history to the next one.

These contributions are then weighted according to the settings and added.

**Parameters:**

**Restart in case of simplex collapse** Whether to restart it when it collapses or just to end the optimisation. Collapsing might mean that the solver found a local minimum, especially in low dimensions. In general, though, it is advisable to activate this, as it can get the simplex out of local minima (and stopping in higher dimensions (more than 5 or so) often doesn't even mean that there was a local minimum).

**If no improvement since last restart, expand simplex** Whether or not to expand the simplex when repeatedly restarted from the same point. Recommended to activate.

**by a factor of** The factor by which the simplex extents are multiplied. A negative value is advised to mirror the simplex and look into other directions.

**Rotate it in a random fashion at every restart** Whether or not to rotate the simplex upon restart. This is considered more explorative than constructing it just along the coordinate directions. For settings of random number generation: see Section 31.2.6.

**Linesearches in moving direction (recommended in dimensions >= 20)** Whether or not to perform the above described linesearch-add-on. In high dimensions, this can be useful, because the simplex spends more time finding a good direction rather than moving into it. The linesearch can then detect this moving direction and go directly into it. In lower dimensions, though, the simplex moves quite well on its own.

**Trigger linesearch after given number of improvements...** One of the possible conditions to trigger linesearch.

**... or after N iterations without improvement. N =** Further possible condition to trigger linesearch.

**Required effectiveness compared to previous simplex search** Quantity determining the effect of the linesearch on the simplex displacement.

**Weight of simlex based gradient for search direction** Weight, with which the gradient of the simplex enters the direction of the linesearch.

### 31.3.4 DIRECT

The method “Dividing RECTangles” natively works on bounded domains, evaluating the function at the nodes of a hyperrectangle grid covering the search space, and iteratively refining this grid in the

“potentially optimal” regions. If the function is known to be Lipschitz continuous, the potentially optimal hyperrectangles in an iteration can be identified based on the function evaluations prior to that iteration.

**Algorithm:** This implements the DIRECT algorithm from

Lipschitzian Optimisation Without the Lipschitz Constant  
by Jones, Pertunen and Stuckman,  
(Journal of Optimisation Theory and Application, Vol 79, No. 1, October 1993)

It begins by normalising the search space into the hypercube  $[0,1]^d$ . The transformation is affine linear, except in case of unbounded domains. The solver evaluates at the centre of the hypercube as initialisation of the following iteration loop:

1. Determine the set of potentially optimal (hyper)rectangles. A rectangle is potentially optimal if there exists an  $L > 0$  such that, if this  $L$  is assumed to be the Lipschitz constant of  $f$ , based on the function values on the rectangle centres, this rectangle contains the point where the smallest function value is still possible (based on centre value and rectangle diameter).
2. Select from these rectangles the ones that promise a relative improvement of the currently known best function value of at least  $\varepsilon > 0$  (assuming the respective Lipschitz constant), i.e. the ones that satisfy

$$f_{\text{possible}} \leq f_{\text{best}} - \varepsilon \cdot |f_{\text{best}}|. \quad (31.9)$$

3. In each of these rectangles, evaluate the function on new centre points: these centre points are those found by going one third of the rectangle length in the coordinate axis directions on those coordinates along which the rectangle is longest (so these are between 2 and  $2d$  points per rectangle).
4. Subdivide the rectangles along the dimension determined in the previous step. There often is a choice on how to subdivide them (for example, a square can be divided into thirds horizontally or vertically). In those cases, subdivision is done such that the biggest subrectangles will contain the best function values found in the previous step. (This is because they will be more likely to be further examined in the future because of a larger diameter)

#### Parameters:

**Minimal relative improvement for potential optimality** This is the  $\varepsilon$  in the algorithm above. Bigger values emphasise global search more, smaller values make the solver focus more on exploitation of good regions it already found.

### 31.3.5 BFGS

**Algorithm:** A limited memory variant is provided. The algorithm maintains an approximation  $H$  of the Hessian matrix and works in the following way:

$$H_0 = 1_{d \times d} \quad (\text{identity matrix})$$

$$s_n = -H \nabla f(x_n)$$

$$x_{n+1} = \text{result of linesearch in direction } s_n$$

$$\varrho_n = \frac{1}{\langle \nabla f(x_{n+1}) - \nabla f(x_n), x_{n+1} - x_n \rangle}$$

$$\begin{aligned} H_{n+1} = H_n - \varrho_n &\left( (x_{n+1} - x_n)(\nabla f(x_{n+1}) - \nabla f(x_n))^T H_n \right) \\ &+ H_n (\nabla f(x_{n+1}) - \nabla f(x_n)) (x_{n+1} - x_n)^T \\ &+ \varrho_n^2 \left( \left( \langle \nabla f(x_{n+1}) - \nabla f(x_n), H_n (\nabla f(x_{n+1}) - \nabla f(x_n)) \rangle + \frac{1}{\varrho_n} \right) (x_{n+1} - x_n) (x_{n+1} - x_n)^T \right) \end{aligned}$$

Before the linesearch, though, it is checked whether  $\partial_{s_n} f(x_n) < 0$  (so if there is expected decrease). If not,  $H_n$  is reset to the identity matrix.

**Parameters:** Settings for gradient calculation, see Section 31.2.7.

### 31.3.6 Legacy (Quasi-Newton)

#### Algorithm:

This solver exists mainly for legacy reasons; it was the only available algorithm in older *PowerFactory* versions and implements a version of the Quasi-Newton method.

**Parameters:** No parameters to set.

## 31.4 What solver should I pick? (Pros and Cons of the different solvers)

The following subsections give a description of the strong and the weak points of the solvers:

### 31.4.1 PSO - Particle Swarm Optimisation

PSO is quite a robust solver that optimises relatively well on a large variety of problems. It is therefore a good choice if you don't know much about your optimisation problem. In case of "easy" problems, though (for instance, convex functions), gradient based solvers like BFGS will probably be faster in convergence to the optimum.

### 31.4.2 Nelder-Mead

Also a relatively robust solver when the restarting option is active (on by default), and is especially good in low dimensions. In dimensions above 20, performance decreases noticeably, though.

### 31.4.3 DIRECT

This is a global optimiser, which means that with a sufficient number of function evaluations, it will find the global optimum in the specified bounds. However, the needed number of function evaluations can be very high, especially in high dimensions, or if the bounds are set unnecessarily wide (or the mesh size too big). It is therefore a good choice for a thorough examination of the search space when you are sure about reasonable bounds and you can afford doing many function evaluations, but other solvers get stuck in local minima. Another possibility is to run DIRECT with a moderate number of function evaluations, and then start another, more local solver with the resulting parameters as the starting point.

### 31.4.4 BFGS

This gradient-based solver is fast on "easy" problems like convex, differentiable functions, and performance decrease in high dimensions is relatively moderate. However, it does not handle well functions with a "rougher surface", i.e. functions that have many local minima and/or discontinuities, piecewise constant functions, etc.

### 31.4.5 Legacy (Quasi-Newton)

Another gradient-based solver, provided mainly for backward compatibility. Offers optimisation in dimensions  $\leq 49$  only. For higher dimensions and the general case, take one of the above.

## 31.5 What can I do if a solver has difficulties in finding good parameters?

Obviously, relaxing your stopping criteria (such as allowing more iterations and function evaluations) potentially leads to better optimisation results.

The major possibility that you have when a solver has difficulties in the optimisation, though, is to try a different solver. They are quite different in their approaches, so a problem that challenges one solver might be solved with less effort by another one. See also Section [31.4](#).

In addition, here is an overview of what could be good ideas if you keep the same solver:

### 31.5.1 PSO - Particle Swarm Optimisation

As PSO is stochastic, it might yield better function values already when restarted with a different random seed. You can also try a different starting point for the optimisation. Another idea is to increase the beaming frequency by entering smaller values for the parameters “Beam particle after n iterations without improvement. n = ” and “Beam all after m iterations without improvement. m = ” and a larger one for “Number of evaluations for the exploration of valleys”. This can allow the swarm to jump out of local minima again. Alternatively, you might also want to try a different PSO variant, like the Cyber Swarm, that searches more slowly, but also more thoroughly.

### 31.5.2 Nelder Mead

Nelder-Mead is sensitive to its starting point, so you could try a different one.

In general, make sure the restarting option is active, unless your problem is low-dimensional and you just want to walk down into the nearest local minimum. If you are unsure if your suggested mesh size is reasonable, start with a small one and set the parameter “If no improvement since last restart, expand simplex by a factor of” to a larger value, like 10.0 or 15.0.

If you optimise in higher dimensions (more than 10 or 20), you can try enabling the “Linesearches in moving direction” option with its settings.

### 31.5.3 DIRECT

DIRECT crucially depends on reasonable bounds and mesh sizes: If a parameter has lower and upper bounds specified, the solver thoroughly examines the range between them, so giving an unnecessarily wide range slows the solver down significantly. If a parameter is unbounded below or above (or totally unbounded), the solver examines the area around the starting value most intensively, with the intensity of the examination decreasing exponentially with distance to the starting point. The radius of intensive search then depends linearly from the suggested mesh size, so again, reasonable values here are very helpful to the solver.

If you optimise in high dimensions, be aware that DIRECT, as a global optimiser, is necessarily relatively slow. You might run it a little and then switch to another, more local solver (see Section [31.4](#)).

### 31.5.4 BFGS

If the solver stopped by itself (i.e. not by the stopping criteria you gave, such as maximum number of function evaluations), you most likely found a local minimum of the function. Restarting the solver then does not make sense. What you can do is try a different starting point for the optimisation, or pick another solver.

### 31.5.5 Legacy

Try a different starting point or pick another solver. Lbfgs and Bfgs, for example, are similar algorithms, but often more accurate.

## Chapter 32

# Modal Analysis / Eigenvalue Calculation

The Modal/Eigenvalue Analysis calculates the eigenvalues and eigenvectors of a dynamic multi-machine system including all controllers and power plant models. This calculation can be completed at the beginning of a transient simulation or at a time step when the simulation is stopped. Note that sometimes in the literature Modal Analysis is referred to as *Eigenvalue Calculation* or *Small Signal Stability*. Throughout this chapter the calculation will generally be referred to as Modal Analysis.

This chapter provides a brief background on the theory of Modal Analysis, followed by a detailed explanation on how to complete such an analysis in *PowerFactory*. The various methods of analysing the results are also presented.

### 32.1 Theory of Modal Analysis

The calculation of eigenvalues and eigenvectors is the most powerful tool for oscillatory stability studies. When doing such a study, it is highly recommended to first compute the “natural” system oscillation modes. These are the oscillation modes of the system when all controller and power plant models are deactivated so every synchronous machine will have constant turbine power and constant excitation voltage. After determining these ‘natural’ modes, the effects of controllers (structure, gain, time constants etc.) and other models can be investigated.

After the initial conditions have been calculated successfully, which means that all time-derivatives of the state variables should be zero (the system is in steady state), or the simulation has been stopped at a point in time, the modal analysis calculates the complete system A-matrix (or at least a selection of it) using numerical, iterative algorithms. The representation of the electrodynamic network model can either be symmetrical or unsymmetrical, depending on the choice of a balanced or unbalanced RMS simulation.

The computation time for the Modal Analysis is approximately proportional to the number of state space variables to the power of three. Considering, that most power system objects and models will contain at least several states (perhaps up to a dozen or more for some complex controllers), the calculation time can rapidly increase as the size of the system being considered increases. For this reason, alternative methods for calculating the system eigenvalues and eigenvectors must be used when the system grows very large. *PowerFactory* supports two types of analysis methods.

A multi-machine system exhibits oscillatory stability if all conjugate complex eigenvalues making up the rotor oscillations have negative real parts. This means that they lie in the left complex half-plane. Electro-mechanical oscillations for each generator are then damped.

More formally, assuming that one of the conjugate complex pair of eigenvalues is given by:

$$\lambda_i = \sigma_i \pm j\omega_i \quad (32.1)$$

then the oscillatory mode will be stable, if the real part of the eigenvalue is negative

$$\sigma_i < 0 \quad (32.2)$$

The period and damping of this mode are given by:

$$T_i = \frac{2 \cdot \pi}{\omega_i} \quad (32.3)$$

$$d_i = -\sigma_i = \frac{1}{T_p} \cdot \ln \left( \frac{A_n}{A_{n+1}} \right) \quad (32.4)$$

where  $A_n$  and  $A_{n+1}$  are amplitudes of two consecutive swing maxima or minima respectively.

The oscillatory frequencies of local generator oscillations are typically in the range of 0.5 to 5 Hz. Higher frequency natural oscillations (those that are not normally regulated), are often damped to a greater extent than slower oscillations. The oscillatory frequency of the between areas (inter-area) oscillations is normally a factor of 5 to 20 times lower than that of the local generator oscillations.

The absolute contribution of an individual generator to the oscillation mode which has been excited as a result of a disturbance can be calculated by:

$$\vec{\omega}(t) = \sum_{i=1}^n c_i \cdot \vec{\phi}_i \cdot e^{\lambda_i \cdot t} \quad (32.5)$$

where:

$\vec{\omega}(t)$	generator speed vector
$\lambda_i$	i'th eigenvalue
$\vec{\phi}_i$	i'th right eigenvector
$c_i$	magnitude of excitation of the i'th mode of the system (at t=0) (depending on the disturbance)
$n$	number of conjugate complex eigenvalues (i.e. number of generators - 1)

In the following  $c$  is set to the unit vector, i.e.  $c = [1, \dots, 1]$ , which corresponds to a theoretical disturbance which would equally excite all generators with all natural resonance frequencies simultaneously.

The elements of the eigenvectors  $\Phi_i$  then represents the mode shape of the eigenvalue  $i$  and shows the relative activity of a state variable, when a particular mode is excited. For example, the speed amplitudes of the generators when an eigenfrequency is excited, whereby those generators with opposite signs in  $\Phi_i$  oscillate in opposite phase.

The right eigenvectors  $\Phi_i$  can thus be termed the "observability vectors". The left eigenvectors  $\Psi_i$  measures the activity of a state variable  $x$  in the  $i$ -th mode, thus the left eigenvectors can be termed the "relative contribution vectors".

Normalisation is done by assigning the generator with the greatest amplitude contribution the relative contribution factor 1 or -1 respectively.

For a n-machine power system, n-1 generator oscillation modes will exist and n-1 conjugate complex pairs of eigenvalues  $\lambda_i$  will be found. The mechanical speed  $\omega$  of the n generators will then be described by:

$$\begin{bmatrix} \omega_1 \\ \omega_2 \\ \dots \\ \omega_n \end{bmatrix} = c_1 \cdot \begin{bmatrix} \phi_{11} \\ \phi_{12} \\ \dots \\ \phi_{1n} \end{bmatrix} \cdot e^{\lambda_1 t} + c_2 \cdot \begin{bmatrix} \phi_{21} \\ \phi_{22} \\ \dots \\ \phi_{2n} \end{bmatrix} \cdot e^{\lambda_2 t} + \dots + c_n \cdot \begin{bmatrix} \phi_{n1} \\ \phi_{n2} \\ \dots \\ \phi_{nn} \end{bmatrix} \cdot e^{\lambda_n t} \quad (32.6)$$

The problem of using the right or left eigenvectors for analysing the participation of a generator in a particular mode i is the dependency on the scales and units of the vector elements. Hence the eigenvectors  $\Phi_i$  and  $\Psi_i$  are combined to a matrix  $\mathbf{P}$  of participation factor by:

$$\underline{P}_i = \begin{bmatrix} P_{1i} \\ P_{2i} \\ \dots \\ P_{ni} \end{bmatrix} = \begin{bmatrix} \phi_{1i} \cdot \Psi_{i1} \\ \phi_{2i} \cdot \Psi_{i2} \\ \dots \\ \phi_{ni} \cdot \Psi_{in} \end{bmatrix} \quad (32.7)$$

The elements of the matrix  $p_{ij}$  are called the participation factors. They give a good indication of the general system dynamic oscillation pattern. They can be used to determine the location of eventually needed stabilising devices to influence the system damping efficiently. Furthermore, the participation factor is normalised so that the sum for any mode is equal to 1.

The participation factors can be calculated not only for the generator speed variables, but for all variables listed in Table 32.1.1.

Name	Unit	Description
s:speed	p.u.	Speed
s:phi	rad	Rotor position angle
s:psifd	p.u.	Excitation flux
s:psi1d	p.u.	Flux in 1d-damper winding, d-axis
s:psi1q	p.u.	Flux in 1q-damper winding, q-axis
s:psi2q	p.u.	Flux in 2q-damper winding, q-axis

Table 32.1.1: Variables accessible for eigenvalue calculation

### When are modal analysis results valid?

A modal analysis can be started when a steady-state condition is reached in a dynamic calculation. Normally, such a state is reached following a load-flow calculation, followed by a calculation of initial conditions. However, it is also possible to do a balanced/unbalanced RMS simulation and start a modal analysis after the end of a simulation or during a simulation when you have manually stopped it.

Although, the modal analysis can be executed at any time in a transient simulation it is not recommended that you do so when the system is not in a quasi-steady state. This is because each modal analysis is only valid for a unique system operating point. Furthermore, the theory behind modal analysis shows that the results are only valid for 'small' perturbations of the system. So although you can complete a modal analysis during a large system transient, the results obtained would change significantly if the analysis was repeated a short time step later when the operating point of the system would be significantly different.

## 32.2 How to Execute a Modal Analysis

The Modal Analysis command may be initiated by:

1. Using the toolbar selection button ▾ to choose the *Modal/Eigenvalue Analysis* toolbar.
2. Selecting the *Calculation* → *Modal/Eigenvalue Analysis* option from the main menu.

To quickly complete the modal analysis and capture all eigenvalues using the default options, one can press **Execute** in the subsequent dialog box and the calculation will start. All the options for the *Modal Analysis* command are explained in the following sections.

### Internal Calculation Procedure

When executing the Modal Analysis command by pressing **Execute**, the configured simulation state will be used or recalculated (depending on the configuration settings). Then the modal analysis constructs a system matrix from the load-flow and the dynamic data. The eigenvalues and eigenvectors are calculated directly from that matrix. During this process, *PowerFactory* automatically linearises all relevant power system models.

## 32.3 Modal/Eigenvalue Analysis Command

Within this section, the configuration settings of the *Modal/Eigenvalue Analysis* command are described, based on their appearance in the command dialog.

### 32.3.1 Basic Options

#### 32.3.1.1 Calculation method

**QR/QZ-Method:** uses the QR/QZ eigenvalue calculation method. Further configuration options are provided within the *Algorithm* page (refer to Section 32.3.2).

---

**Note:** The QR method is the “standard” method used in power system analysis. This method, originally developed by J.G.F Francis, can compute **eigenvalues of widely differing magnitudes for higher-order systems**. The QZ algorithm, on the other hand, is a numerically backward stable method for computing **generalised eigenvalues problems**. Both methods calculate all of the system’s eigenvalues.

---

**Selective method(Arnoldi/Lanczos):** uses the Selective method for eigenvalue calculation. The *Selective method* can be configured in further detail via the available options within this dialog (see information provided in Section 32.3.1.3) and via the *Algorithm* page (refer to Section 32.3.2).

---

**Note:** The Selective method (known as the Arnoldi/Lanczos method) only calculates a subset of the system’s eigenvalues based on various constraints related with a particular reference point. This method is often used in very large systems, where the use of either QR or QZ methods could be very time consuming. The *Selective method* is especially useful if the user knows the target area of interest for the eigenvalues.

---

### 32.3.1.2 Operating Point for Calculation

**Use current time in active simulation:** if a simulation has already been executed, then apply this option in order to use the system state at the current simulation time as basis for the eigenvalue calculation.

---

**Note:** Various system states can be obtained depending on the active simulation:

- If only the *Calculation of initial conditions* command has been executed, then the system state used will be the one at initialisation;
- If a simulation is run until time  $t_{sim}$ , the system state will be chosen at that particular time.

In all cases, a correct calculation of the eigenvalues can be done only if the system state is in an equilibrium point, whereas all transients should have already been extinguished.

---

**Recalculate initial conditions:** the system state used for the calculation is obtained by executing a pre-configured *Calculation of initial conditions* command. The command is provided as reference link next to this option and can be modified accordingly.

The Modal/Eigenvalue Analysis supports balanced and unbalanced RMS simulation methods. More information about the options in the *Calculation of initial conditions* command can be found in Chapter 29: RMS/EMT Simulations, Section 29.3.

### 32.3.1.3 Basic configuration of Selective method (Arnoldi/Lanczos)

**Complex reference point (RP):** Complex plane reference point defined using real part and imaginary part/damped frequency, as below.

- *Real part* ( $\sigma_{RP}$ )
- *Imaginary part* ( $\omega_{RP}$ )
- *Damped frequency* (equal with  $\omega_{RP}/2\pi$ )

**Selection of eigenvalues closest to RP:** the selective method determines eigenvalues closer to the reference point using one of three different measures for *closeness*. The options are:

- Based on magnitude: eigenvalues whose magnitudes are closest to the magnitude of the reference point.
- Based on imaginary part: eigenvalues whose imaginary parts are closest to that of the reference point.
- Based on real part: eigenvalues whose real parts are closest to that of the reference point.

This option can be further clarified using a diagram as shown in Figure 32.3.1. The three eigenvalue pairs are as follows:

- A; -0.8 +/- 1.4. Magnitude 1.61
- B; -0.7 +/- 1.5. Magnitude 1.65
- C; -0.5 +/- 2.0. Magnitude 2.06

Say the reference point was set to the origin (0,0. Magnitude 0). Then using the first method above, the closest eigenvalue pair would be A because this pair has a magnitude closest to the magnitude of the reference point. Using method 2, the closest pair would also be A because this pair has the smallest imaginary part with respect to the imaginary part of the reference point. Finally, using the third method, the closest pair would be C because this pair has a real component closest to the real part of the reference point.

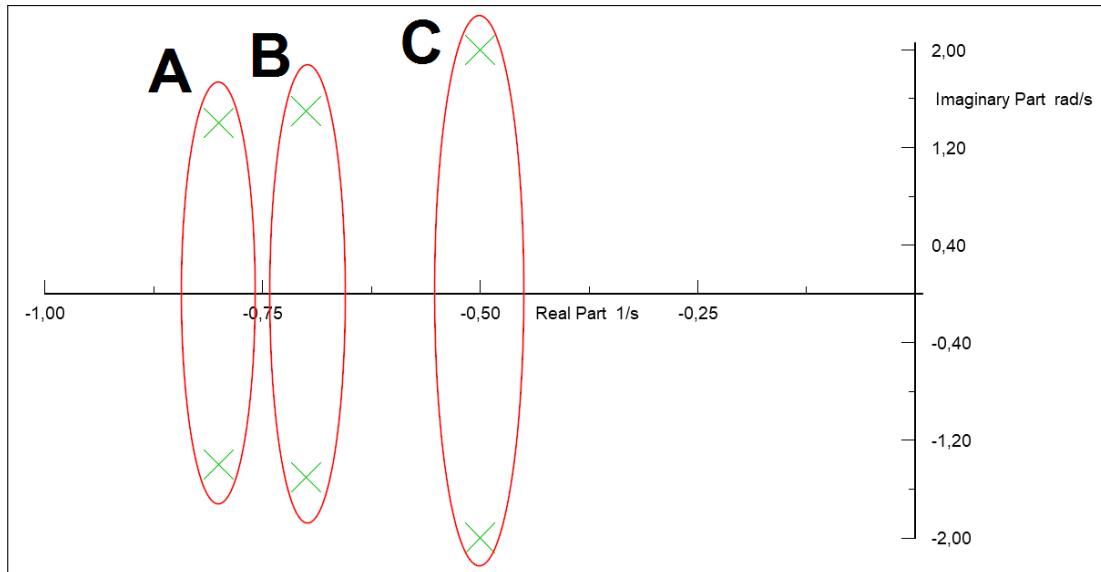


Figure 32.3.1: Illustration of different eigenvalue selection methods

**Number of Eigenvalues:** the maximum number of calculated eigenvalues. Using this parameter it is possible to limit the total number of eigenvalues calculated by the *Selective method*. An eigenvalue pair is defined as *one* eigenvalue mode for this calculation.

### 32.3.2 Algorithm

#### 32.3.2.1 General

##### Calculate

- Left Eigenvectors (Controllability): calculates the left eigenvectors that are used to identify the *Controllability*, i.e. the contribution of a variable to the activity of a particular mode. This option is enabled by default.
- Right Eigenvectors (Observability): calculates the right eigenvectors that are used to identify the *Observability*, i.e. the degree of activity of a state variable when a particular mode is excited. This option is disabled by default.
- Participation Factors: calculates the participation factors for each state variable, these measure the relative participation of the state variable in the mode. This option is disabled by default.

---

**Note:** The *Controllability*, *Observability* and *Participation Factors* for any mode can be visualised using the *Mode Polar Plot* or *Mode Bar Plot* described in Section 32.4.2.

---

##### Algorithm options for QR/QZ-Method

If the *QR/QZ-Method* is selected in the *Basic Options* page, then specific customisation settings are provided here.

- *QR method using system reduction*: use this option to use the QR method for the analysis. If, for any reason, the QR method fails to generate a solution, then the QZ method is automatically employed as fallback option.

- *QZ method for generalised system*: use this option to apply the QZ method directly. The calculation using this method is slower compared to the QR method. The option should only be used, when it is known that the QR-method will fail for the evaluated system.
- *Apply algorithm for improved conditioning*: this option computes a balancing transformation to improve the conditioning of the eigenvalues and eigenvectors. It should be used if there are elements with no impedance or virtual impedance in the network and/or if the results of the modal analysis are outside of the expected range.

#### Algorithm options for **Selective method**

If the *Selective method* is selected in the *Basic Options* page, then specific customisation settings are provided here.

- *Initialisation of Arnoldi Iteration*: the start vector of the iterative algorithm can be here customised. The user may choose to use a *random start vector* or a *standard unit vector(s)* to initialise the Arnoldi algorithm.
- *Eigenvalue convergence tolerance*: this parameter sets the eigenvalue convergence tolerance within the calculation.

#### 32.3.2.2 Advanced

##### Advanced options for **QR/QZ-Method**

- *Direct construction of A-matrix (as in V13.2)*: legacy option, used to reproduce results generated with *PowerFactory* version 13.2. Unless specifically required, it is recommended not to use this option in any newer version.

##### Advanced options for **Selective method**

- *Identification of Eigenvalues - consider identical if distance smaller*: use this parameter to specify the tolerance for eigenvalues when its eigenvectors are associated. Left and right eigenvectors are calculated separately with this algorithm in *PowerFactory*.
- *Omit eigenvalues if eigenvector-transformation fails*: when this option is checked, eigenvalues where the association between the left a right eigenvectors can not be found, will be ignored.

#### 32.3.3 Results

**Recording filter for eigenvalues:** settings to restrict the recorded eigenvalues:

- *Oscillatory modes only*: record only modes which have non-zero imaginary part;
- *Unstable modes only*: record only modes which are on the right half side of the complex plane
- *Min. real part / Max. real part*: constrain eigenvalue results within a minimum/maximum real part range
- *Min. imag. part / Max. imag. part*: constrain eigenvalue results within a minimum/maximum imaginary part range
- *Min. magnitude / Max. magnitude*: constrain eigenvalue results within a minimum/maximum magnitude range
- *Min. damp. ratio / Max. damp. ratio*: constrain eigenvalue results within a minimum/maximum damping ratio range

**Report filtered out eigenvalues in output window:** set this option in order to report to the output window any eigenvalues which have been filtered out by the *Recording filter for eigenvalues*.

**Results:** a reference link to the *Results* object is provided. The *Results* object can also be changed here, if needed.

### 32.3.4 Output

#### 32.3.4.1 Output to MATLAB files

The Modal/Eigenvalue Analysis results can be exported to a MATLAB readable file format (sparse matrix file). The user can select which results are to be exported. A short explanation of several of the exported matrices is provided below.

The system representation of a non-linear dynamic system can be described in general terms using two nonlinear functions  $f$  and  $g$ , as below:

$$\begin{aligned}\dot{\mathbf{x}} &= f(\mathbf{x}, \mathbf{v}) \\ 0 &= g(\mathbf{x}, \mathbf{v}, \dot{\mathbf{v}})\end{aligned}$$

where  $\mathbf{x}$  is a vector of  $n$  state variables,  $\mathbf{v}$  is an  $m$  length vector containing algebraic variables such as passive power network variables and control variables of DSL models.

The linearised combined control and passive power system can be expressed using a set of equations containing the partial derivatives of the state variables and network quantities, as follows:

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_{22} \end{bmatrix} \begin{bmatrix} \Delta \dot{\mathbf{x}} \\ \Delta \dot{\mathbf{v}} \end{bmatrix} = \begin{bmatrix} \mathbf{J}_{\mathbf{xx}} & \mathbf{J}_{\mathbf{xv}} \\ \mathbf{J}_{\mathbf{vx}} & \mathbf{J}_{\mathbf{vv}} \end{bmatrix} \begin{bmatrix} \Delta \mathbf{x} \\ \Delta \mathbf{v} \end{bmatrix} \quad (32.8)$$

where:

- $\mathbf{I}$  is the identity matrix ( $n$  by  $n$ ),  $\mathbf{M}_{22}$  is an  $m$  by  $m$  matrix.
- $\mathbf{J}_{\mathbf{xx}}$  ( $n$  by  $n$  matrix) contains partial derivatives of a differential equation to the state variables;
- $\mathbf{J}_{\mathbf{xv}}$  ( $n$  by  $m$  matrix) contains partial derivatives of a differential equation to the network quantities;
- $\mathbf{J}_{\mathbf{vx}}$  ( $m$  by  $n$  matrix) contains partial derivatives of an algebraic equation to the state variables;
- $\mathbf{J}_{\mathbf{vv}}$  ( $m$  by  $m$  matrix) contains partial derivatives of an algebraic equation to the network quantities.

---

**Note:** Matrix  $\mathbf{M}_{22}$  is normally zero, but not always, e.g. components in the DC power system contain the DC voltage derivative, etc.

---

In a more concise representation, the linearised power system can be expressed as below:

$$\mathbf{M} \begin{bmatrix} \Delta \dot{\mathbf{x}} \\ \Delta \dot{\mathbf{v}} \end{bmatrix} = \mathbf{J} \begin{bmatrix} \Delta \mathbf{x} \\ \Delta \mathbf{v} \end{bmatrix} \quad (32.9)$$

where  $\mathbf{J}$  ( $n + m \times n + m$ ) is the Jacobian matrix.

Based on the above, the system matrix  $\mathbf{A}_{\text{mat}}$  can be defined as below:

$$\mathbf{A}_{\text{mat}} = \mathbf{J}_{\mathbf{xx}} - \mathbf{J}_{\mathbf{xv}} \mathbf{J}_{\mathbf{vv}}^{-1} \mathbf{J}_{\mathbf{vx}} \quad (32.10)$$

For the system matrix  $\mathbf{A}_{\text{mat}}$ , the following is true:

$$\Delta \dot{\mathbf{x}} = \mathbf{A}_{\text{mat}} \cdot \Delta \mathbf{x} \quad (32.11)$$

---

**Note:** The input (B), output (C) and feedforward (D) matrices of a state space representation of a system are not used within the linearised power system formulation, hence they are not available within the Modal Analysis calculation.

---

The data that can be exported externally are:

- System matrix  $\mathbf{A}_{\text{mat}}$ : file **Amat.mtl**;
- System eigenvalues: file **EVals.mtl**;
- Left eigenvectors: file **IEV.mtl**;
- Right eigenvectors: file **rEV.mtl**;
- Jacobian matrix  $\mathbf{J}$ : File **Jacobian.mtl**;
- $\mathbf{M}$  matrix: file **M.mtl**;
- Participation factors: file **PartFacs.mtl**.

Along with the above files, a description of the rows and columns for the Jacobian  $\mathbf{J}$  and the system  $\mathbf{A}_{\text{mat}}$  matrices is provided within two text documents named **VariableToldx\_Jacobian.txt** and **VariableToldx\_Amat.txt**, such that the corresponding states or algebraic variables can be identified.

The exported files are in an ASCII sparse matrix format “.mtl”. For conversion into a full matrix format in the MATLAB environment, please refer to the MATLAB documentation (e.g.  $H = \text{spconvert}(D)$ ;  $\text{full}(H)$ ). To import and expand a sparse matrix in MATLAB, execute, for the example of the system matrix and the right eigenvectors, the following commands:

```

» load Amat.mtl;
» load rEV.mtl;
» Amat = full(spconvert(Amat));
» rEV = full(spconvert(rEV));

```

Using the commands above, the variables *rEV* and *Amat* should have been loaded in the MATLAB workspace as full matrices/vectors. A short list of useful MATLAB commands is provided below, depending on the calculation type.

**QR Method:** this method is the only one which computes the system matrix  $\mathbf{A}_{\text{mat}}$ .

- To verify the *j*'th right eigenvector, the following applies:

$$\mathbf{A}_{\text{mat}} \cdot \mathbf{rEV}(\text{j-th column}) = \text{Evals}(\text{j-th column}) \cdot \mathbf{rEV}(\text{j-th column}) \quad (32.12)$$

MATLAB code (returns the difference between the left and right terms):

- »  $\text{Amat} * \text{rEV}(:,j) - \text{Eval}(j) * \text{rEV}(:,j)$
- To verify the j'th left eigenvector, the following applies:

$$\text{IEV}(\text{j-th column})^* \cdot \text{A}_{\text{mat}} = \text{Eval}(\text{j-th column}) \cdot \text{IEV}(\text{j-th column})^* \quad (32.13)$$

MATLAB code (returns the difference between the left and right terms):

- »  $(\text{conj}(\text{IEV}(:,j))' * \text{Amat})' - \text{Eval}(j) * \text{conj}(\text{IEV}(:,j))$

---

**Note:** The left eigenvectors require an additional conjugation before being used

- To verify the eigenvalues, the following commands need to be entered:

- »  $\text{D} = \text{eig}(\text{Amat})$
- »  $\text{D} = \text{eig}(\text{Amat}, \text{M})$

**QZ and Arnoldi Methods:** these two calculation methods do not generate the system matrix  $\text{A}_{\text{mat}}$ .

- To verify the right eigenvectors, the following applies, where  $\text{D}$  is a diagonal matrix containing the eigenvalues:

$$\text{J} \cdot \text{rEV} = \text{M} \cdot \text{rEV} \cdot \text{D} \quad (32.14)$$

MATLAB code to verify the j-th right eigenvector (returns the difference between the left and right terms above):

- »  $\text{Jacobian} * \text{rEV}(:,j) - \text{M} * \text{rEV}(:,j) * \text{Eval}(j)$
- To verify the left eigenvectors, the following applies, where  $\text{D}$  is a diagonal matrix containing the eigenvalues:

$$\text{IEV}^* \cdot \text{J} = \text{D} \cdot \text{IEV}^* \cdot \text{M} \quad (32.15)$$

MATLAB code to verify the j-th left eigenvector (returns the difference between the left and right terms above):

- »  $\text{transpose}(\text{conj}(\text{IEV}(:,j))) * \text{J} - \text{Eval}(j) * \text{transpose}(\text{conj}(\text{IEV}(:,j))) * \text{M};$
- To verify the eigenvalues, the following command needs to be entered:

- »  $\text{EIGEN} = \text{eig}(\text{Jacobian}, \text{M})$

### 32.3.4.2 Output

The calculation command provides output window information. The contents can be customised as follows:

- *Short*: brief output window messages documenting only the main steps in the calculation;
- *Detailed*: the calculation generates detailed information, useful sometimes for debugging various scenarios.

### 32.3.4.3 Report DSL models containing buffers or external function

This option will generate a message if a DSL model is using any of the buffer based DSL functions (e.g. delay, movingavg, lastvalue) or an external DLL model (based on digexdyn, digexfun or IEC61400-27-1 interfaces).

---

**Note:** C-Interface DLL models (i.e. compiled DSL models), described in [30.14.3](#), are not reported by this option, unless a buffer based DSL function is used within this model. The reason is that C-Interface DLL models are behaving within Modal Analysis identically as a DSL model (uncompiled).

---

It should be noted that for buffer based functions and external DLL based models the linearisation algorithm will generate a linear equation of the linearised output of the form  $\frac{\Delta y_o}{\Delta y_i} = 0$ , where  $y_o$  and  $y_i$  are the output and the input of the buffer based function (or of the external DLL), respectively. The buffer based functions are marked accordingly with a '\*' symbol within Table [30.16.2](#). Whenever such functions are reported, it is recommended to verify the specific models in order to ensure correctness of results.

---

**Note:** One example in which the use of buffer based functions does not affect Modal Analysis results is the case when the function is being applied to a not connected input signal or to a time-independent constant expression. In such a situation the result is clearly correct since the linearised equation should be zero. If these functions are directly applied within a control feedback path (e.g. on signals like voltage, current, power or others) then the eigenvalue analysis may lead to inaccurate results. Further investigations are recommended in these cases.

---

More information about DSL Models is available in Chapter [30](#): Models for Dynamic Simulations.

### 32.3.4.4 Report state variables that are assumed to be constant

This option will generate a message for those state variables of models that are assumed to be constant at the current operating point.

For example, consider the following equation:

```
x. = select(yi>0,yi,select(yi<0,yi,0))
```

If  $y_i=0$  then the derivative of  $x.$  will always be zero.

When the option *Report state variables that are assumed to be constant at the current operating point* is selected, a message will be displayed in the output window pointing to the DSL model where the state variable was identified. The user can then check the model and determine whether the reported states are within the supposed active path. If this is the case, an adjustment of the model code or parameterisation may be indicated.

## 32.4 Viewing Modal Analysis Results

There are several ways to visualise the results of the modal analysis, including using the built-in plots within *PowerFactory*, using the Modal/Eigenvalue Analysis Results command, visualising the eigenvalues in the single line diagrams or using the predefined reports to print the result in the output window. This section describes these options.

### 32.4.1 Modal/Eigenvalue Analysis Results Command

The modal analysis results can be displayed in a convenient data browser specially designed for working with these results, which is accessible by clicking on the *Modal/Eigenvalue Analysis Results* icon (grid with red cross) from the Modal/Eigenvalue Analysis toolbar. The options of this command are explained below.

#### Show Modes

When the option *Modes* is selected from the *Shown values* field, a table with the calculated variables for each eigenvalue is displayed (see Figure 32.4.1). The variables and the corresponding equations for an eigenvalue  $\lambda = \sigma + j\omega$  are shown in Table 32.4.1.

Name	Unit	Equation
Real part	1/s	$\sigma$
Imaginary part	rad/s	$\omega$
Magnitude	1/s	$\sqrt{\sigma^2 + \omega^2}$
Angle	deg	$\text{atan2}(\sigma/\omega)180/\pi$
Damped Frequency	Hz	$ \omega/2\pi $
Period	s	$ 2\pi/\omega $
Damping	1/s	$-\sigma$
Damping Ratio	%	$-100\sigma/\sqrt{\sigma^2 + \omega^2}$
Ratio A1/A2		$e^{-2\pi\sigma/ \omega }$
Logarithmic decrement		$-2\pi\sigma/ \omega $

Table 32.4.1: Variables calculated for each eigenvalue

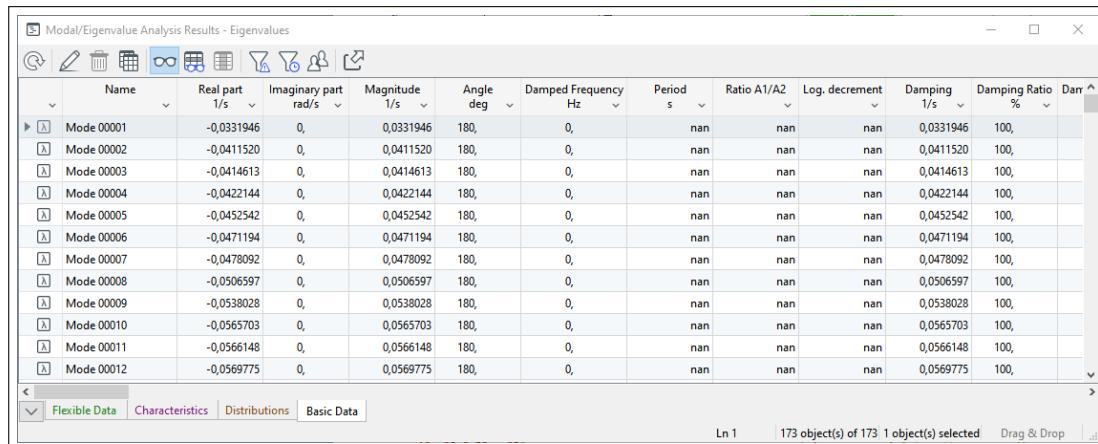


Figure 32.4.1: Result variables for each mode

#### Show States

When selecting the option *States* from the *Shown values* field, the user can choose to show the state variables of one mode, of all oscillatory modes or of a group of modes. In this way it is easier to identify all the modes where a particular state variable participates.

The results in the eigenvalue *Modes* or *States* tables can be then sorted and filtered as shown in Figure 32.4.2, where the states in which the speed of a certain generator participates are further filtered to show just modes with a damping ratio less than 10 %.

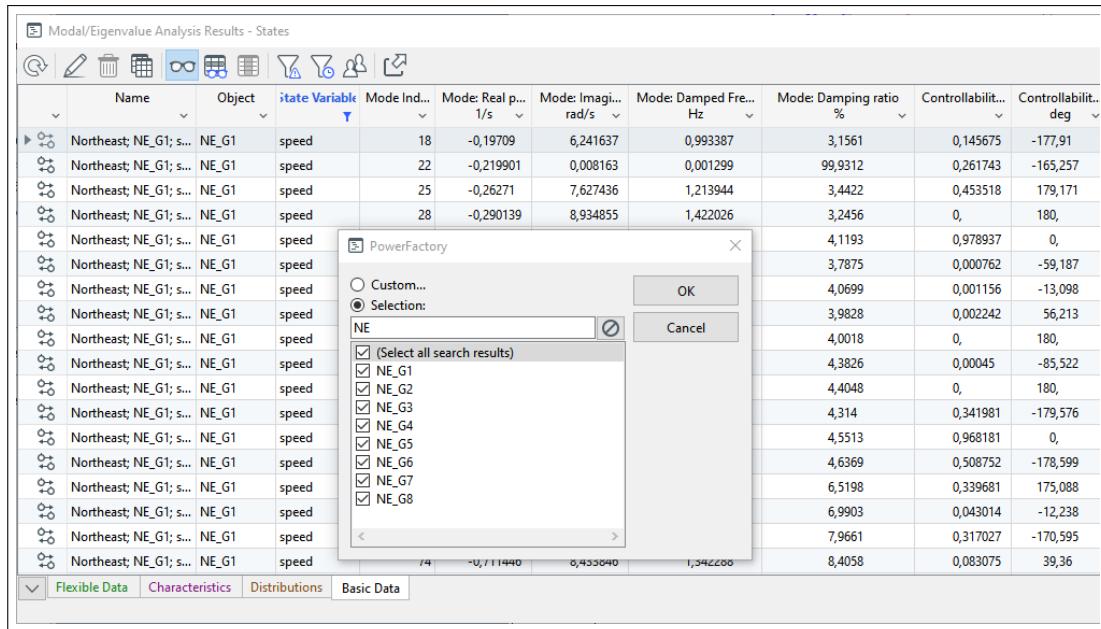


Figure 32.4.2: Results Filtering

It is also possible to display the list of state variables of a particular mode from the modes result table by right-clicking on a mode and selecting *Show State results*.

### Exporting the results from the Eigenvalues or States tables

The results shown in the Modal Analysis data browser can be exported to an external software program (such as a spreadsheet tool) following these steps:

1. Select the data to be exported. To select all data press **ctrl-A**.
2. Click on the *Copy (with column headers)* icon ( ). Alternatively right-click within the selection and choose the option *Spread Sheet Format → Copy (with column headers)*.
3. Open the external software and paste the data from the windows clipboard.

### 32.4.2 Modal Analysis Results in Built-in Plots

There are three special plot types in *PowerFactory* for visualising the results of a modal analysis calculation: the eigenvalues plot, the mode bar plot and the mode polar plot.

Each type of plot can be automatically created by clicking on the *Create Plot* icon ( ) and selecting the desired plot type from the *Insert Plot* dialog. The Modal/Eigenvalue Analysis plots are shown in Figure 32.4.3.

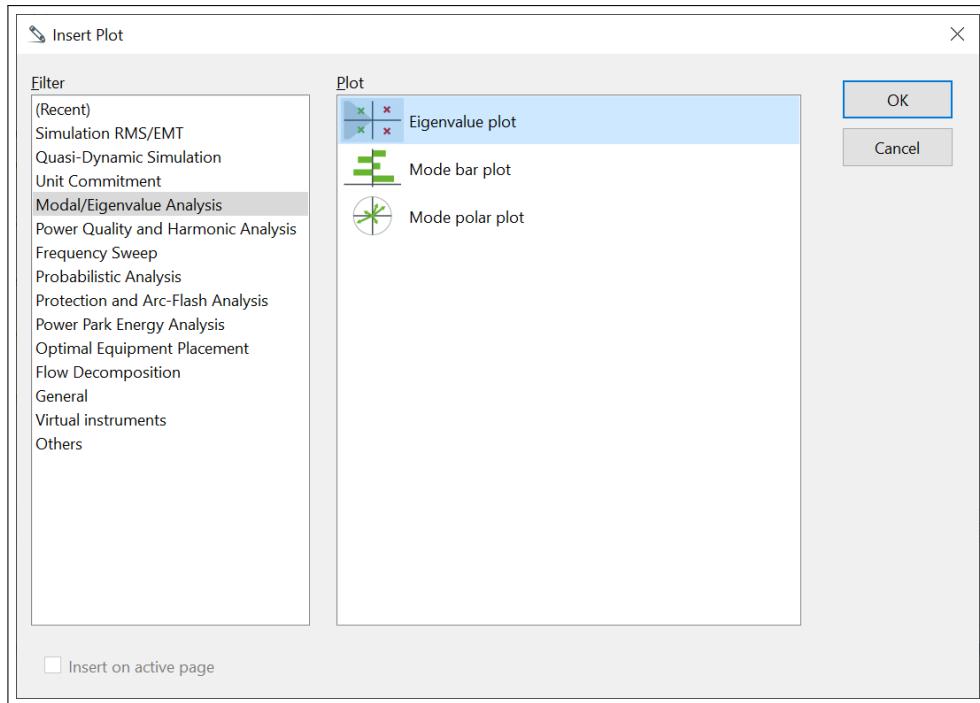


Figure 32.4.3: Selection of the Modal Analysis plots

### 32.4.2.1 Eigenvalue Plot

Eigenvalue plots are inserted by choosing *Eigenvalue plot* from the insert dialog shown in Figure 32.4.3. The plot will appear in a new window. Note, that each time the option is selected, a new plot window will be created.

#### Interpreting the Eigenvalues Plot

An example of an eigenvalue plot is shown in Figure 32.4.4.

The Eigenvalue Plot displays the calculated eigenvalues in a two axis coordinate system. For the vertical axis, it is possible to select among the imaginary part ( $rad/s$ ), or the damped frequency ( $Hz$ ) of the eigenvalue. The horizontal axis shows the real part.

Stable eigenvalues are shown in green (default) and unstable eigenvalues in red (default). Each eigenvalue can be inspected in detail by double clicking on it; this will open a dialog where the index, the complex representation, the polar representation and oscillation parameters of the mode are displayed.

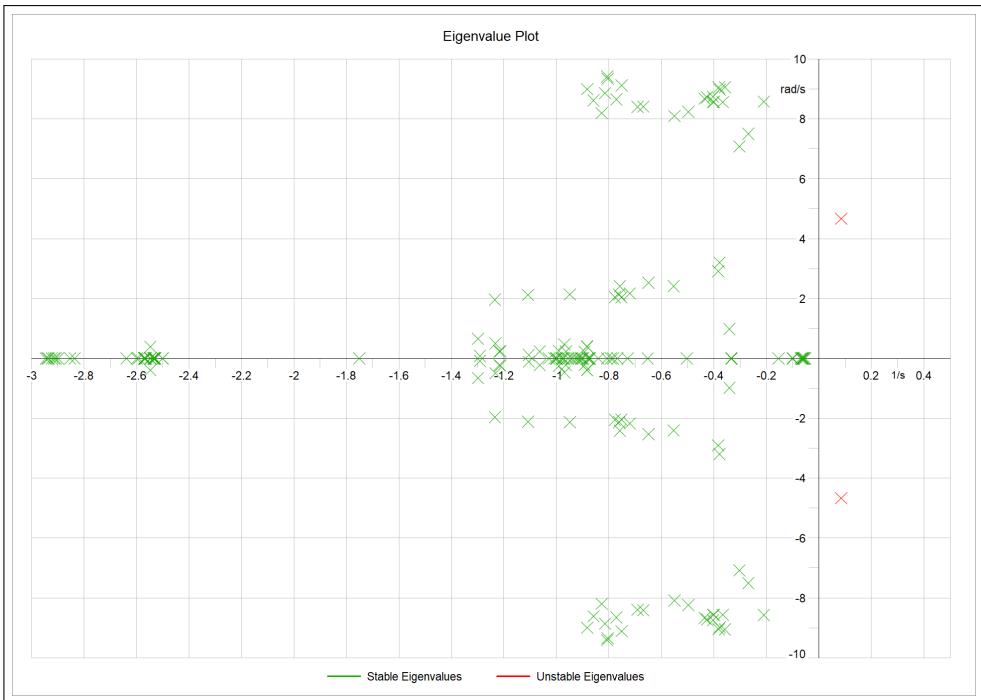


Figure 32.4.4: The Eigenvalue Plot

### Editing the Eigenvalue Plot

All settings that control the appearance of the eigenvalue plot can be accessed by double clicking on an empty area of the plot. On the *Shown Data* page, the following options are available:

- Modal Analysis Results: to compare various scenarios and visualise the results of Modal Analysis calculations, it is possible to add multiple result files corresponding to different scenarios to the eigenvalue plots. Additionally, the colour and the size of the stable and unstable eigenvalues can be adjusted.
- Plot Features: it is also possible to decide whether to display the stability borders when the imaginary part is shown in the y axis. The *Show Stability Borders* option is used to shade the area of the plot containing all the stable modes present in the displayed plot.
- Filter Options: here the display of eigenvalues on the plot is restricted according to defined criteria. Eigenvalues can be restricted by range (independently in either the x or y axes) by selecting the *Restrict Range* option. The *Restrict Indexes* options allows the user to choose, from the complete list of eigenvalues, a limited subset to display on the plot. Alternatively, just the oscillatory modes can be displayed by choosing the option *Show Oscillatory Modes*.

On the *Style and Layout* page, the following additional display options can be selected:

- Curve Area Position (mm): if *Auto-Position Curve Area* is selected, the position of the curve is automatically set to fit the page size. If it is deselected, then the user can manually define the size and position of the plot.
- Border and Background
  - **Draw border:** to define outside border and colour of the curves area.
  - **Fill background:** to define if the curves area should be filled, and select the colour.
- Colour palette: link to the colour palette being used. A new colour palette can be selected using the *Select arrow* .

### Damping Ratio Constant

A constant showing a predefined damping ratio can be added into the plot by right-clicking on the plot and choosing *Add Constant damping ratio line*. The use of this constant, shown in Figure 32.4.5, simplifies the graphical identification of the less damped modes. The damping ratio constant is automatically adapted when changing the y-axis between *Damped Frequency* and *Imaginary Part*.

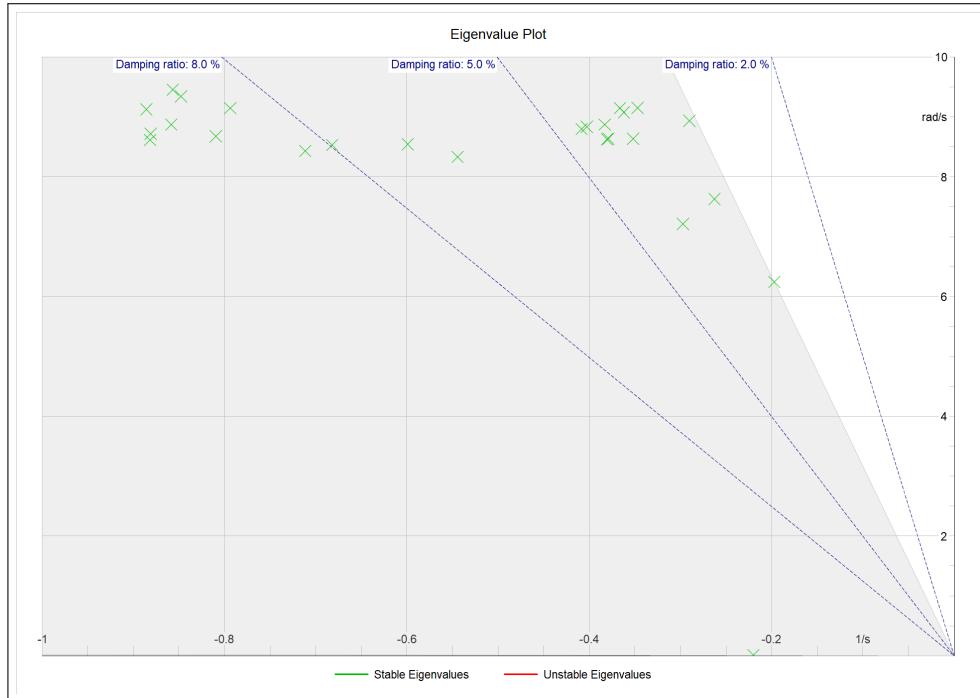


Figure 32.4.5: Damping Ratio Constants in Eigenvalues Plot

#### 32.4.2.2 Mode Bar Plot

Mode bar plots can be inserted by choosing *Mode bar plot* from the insert dialog shown in Figure 32.4.3. When this option is selected, an edit dialog of the plot will open.

##### Editing the Mode Bar Plot

On the *Shown Data* page, the following options are available:

- Mode Selection: to choose the mode displayed on the plot. The observability, controllability or participation factors will then be displayed for this mode. Note that it is also possible to enter the real and imaginary values if the mode index is not known; *PowerFactory* will then automatically select the closest mode.
- Shown values: to select to display either the Controllability, Observability or Participation Factors for the selected mode.
- Filter Options: to choose to restrict the display of variables on the plot according to defined criteria. Displayed variables can be restricted to a minimum contribution by selecting the *Min. Contribution* option, or for greater control the variables to display can be selected manually by selecting the *User Defined States* option and manually choosing the variables to display.
- Appearance: to adjust the colour and style of the bars and choose to show the state variable legend annotation (value) for each bar.
- Scaling: configures the mode bar plot scale. If the *Auto scale* checkbox is ticked, then the plot is autoscaled. Otherwise, the plot bar limit can be specified in the corresponding *Radius limit* field.

- Plot Linking: this enables the user to link the mode bar plot to the eigenvalue plot. Moreover, there is also the possibility to create linked mode bar plots directly from eigenvalue plots. This is discussed in Section 32.4.2.4.

On the *Style and Layout* page, the options are the same as the Eigenvalue Plot, described in Section 32.4.2.1.

### Interpreting the Mode Bar Plot

An sample Mode Bar Plot is shown in Figure 32.4.6. The Mode Bar Plot displays the controllability, observability or participation factors of variables for a user selected eigenvalue in bar chart form. This allows easy visual interpretation of these parameters.

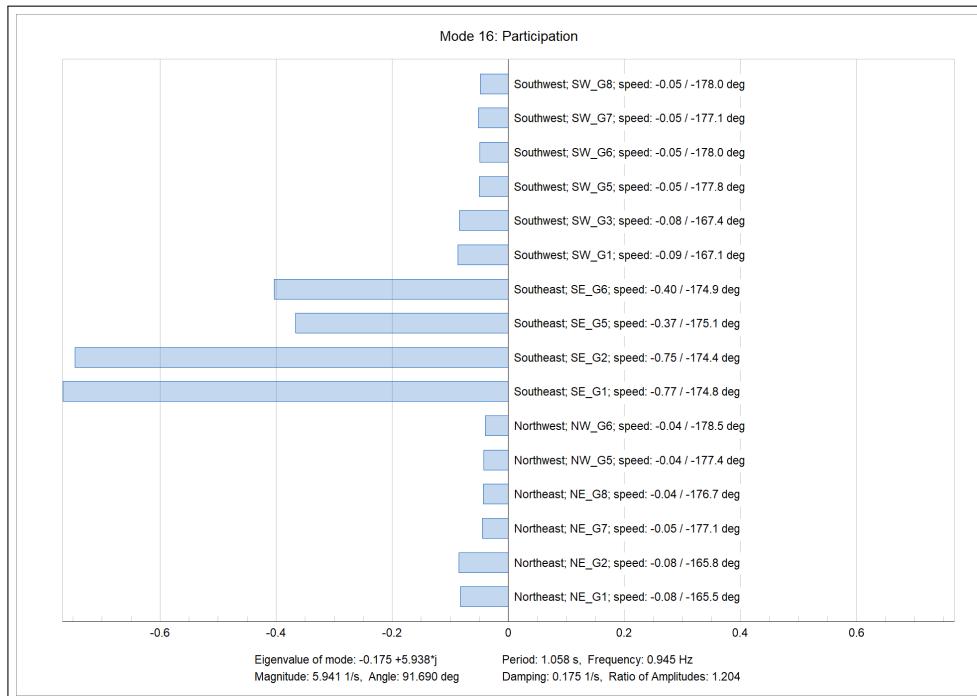


Figure 32.4.6: Example Mode Bar Plot

The mode bar plot can also be obtained in one of the following ways:

- From the eigenvalues plot: right-click on a mode and select *Create Bar Plot for Mode* → *Controllability/Observability/Participation*.
- From the modal analysis modes result: right-click on a mode and select *Plots* → *Bar plot*→ *Controllability/Observability/Participation*.

Note that the observability and participation factors are only shown if these calculations were enabled in the Modal Analysis command as described in Section 32.3.2.1.

### 32.4.2.3 Mode Polar Plot

Mode polar plots can be inserted by choosing *Mode polar plot* from the insert dialog shown in Figure 32.4.3. When this option is selected an edit dialog of the plot will open, with options available as explained below. Note, every time this option is selected, a new plot window will be created.

## Editing the Mode Polar Plot

All settings that control the appearance of the Mode Polar Plot can also be accessed by double clicking on an empty area of the plot. On the *Shown Data* page, the available options are very similar to the Mode Bar Plot as described in Section 32.4.2.2 and the Mode Selection, Filter Options, Appearance, Scaling and Plot Linking can be altered in the same way. There are some additional options:

- **Cluster:** enabling this option will *cluster* variables with an angular separation less than the entered value. A cluster shares the same diagram colour.
- **Draw arrow heads:** arrow heads can be drawn on the displayed vectors by enabling this option.

On the *Style and Layout* page, the options are the same as the Eigenvalue Plot, described in Section 32.4.2.1.

## Interpreting the Mode Polar Plot

An example of a mode polar plot is shown in Figure 32.4.7. The mode polar plot displays the controllability, observability or participation factors of variables for a user selected eigenvalue in polar form. Variables are grouped and coloured identically if their angular separation is less than a user defined parameter (default 3 degrees).

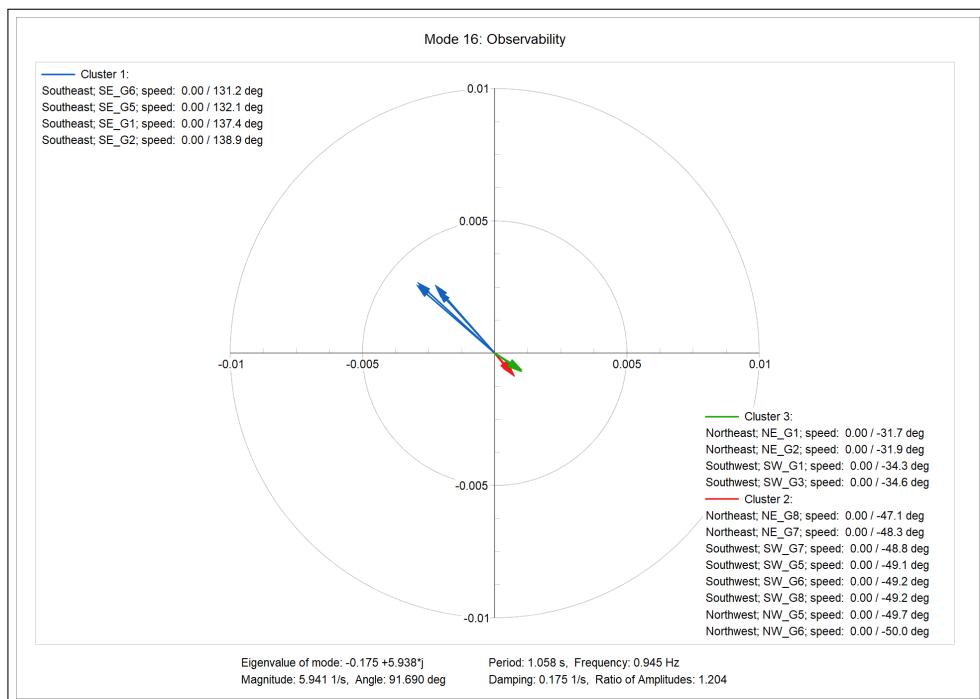


Figure 32.4.7: The Mode polar plot

The mode polar plot can also be obtained in one of the following ways:

- From the eigenvalues plot: right-click on a mode and select *Create Polar Plot for Mode* → *Controllability/ Observability/Participation*.
- From the modal analysis modes result: right-click on a mode and select *Plots* → *Polar plot*→ *Controllability/Observability/Participation*.

Note that the observability and participation factors are only shown if these calculations were enabled in the Modal Analysis Command as described in Section 32.3.2.1.

#### 32.4.2.4 Linked Mode Bar and Polar Plots

The eigenvalue plot is linkable to one or more polar or modal bar plots. Once the plots are linked, the selection of an eigenvalue in the eigenvalue plot will automatically update the mode plots and display the respective plots accordingly.

To create a Linked Mode Bar Plot, right-click on a mode (or group of modes) in the eigenvalue plot and select *Create Linked Mode Bar Plot* → *Controllability/Observability/Participation*. To create a Linked Mode polar plot, select *Create Linked Mode Polar Plot* → *Controllability/Observability/Participation*.

#### 32.4.3 Eigenvalues Results in Single Line Diagrams

After the execution of the Modal Analysis command, it is possible to draw the eigenvectors of a particular mode directly on a graphic (single line, overview or geographical diagram), with options to show right eigenvectors, left eigenvectors or participation factors.

The eigenvectors can be drawn using the *Draw Eigenvectors Arrow* command ( from the Modal Analysis toolbar or directly from the desired mode in the eigenvalues plot. The arrows will only be drawn if the corresponding generator, or the substation/site where the generator belongs, is displayed in the graphic.

#### 32.4.4 Modal Analysis Results in the Output Window

The modal/eigenvalues analysis results can be displayed in the output window using the *Report Generation* command, accessible by clicking on *Generate reports* icon ( from the main toolbar or via the menu *Output* → *Generate reports*...).

The following options are available when selecting the *Modal/Eigenvalues Analysis* from the command:

##### Modes

A report of all the calculated eigenvalues is printed in the output window.

##### Left Eigenvectors (Controllability)/Right Eigenvectors (Observability)/Participations

Selecting any of these options changes the dialog format to the one shown in Figure 32.4.8. The available options are:

- **Mode Selection:** the index of a specific mode can be selected. Alternatively the modes can be *Filtered* by specifying the maximal damping and maximal period.
- **Variable Selection:** the variables to be displayed are selected
  - *Show all* is used to show all variables (for example, speed, phi, psiD)
  - *Min. contribution* is used to filter the displayed variables according to their contribution.
  - *User Defined States* provides greater control over which variables are displayed. The button **Show** shows the currently selected variables. More variables can be added using the **Add** button whereas all variables can be removed by using the **Remove All** button.

---

**Note:** The *Detailed* check-box shows the bar chart in the report, whereas the normal report shows only numerical values.

---

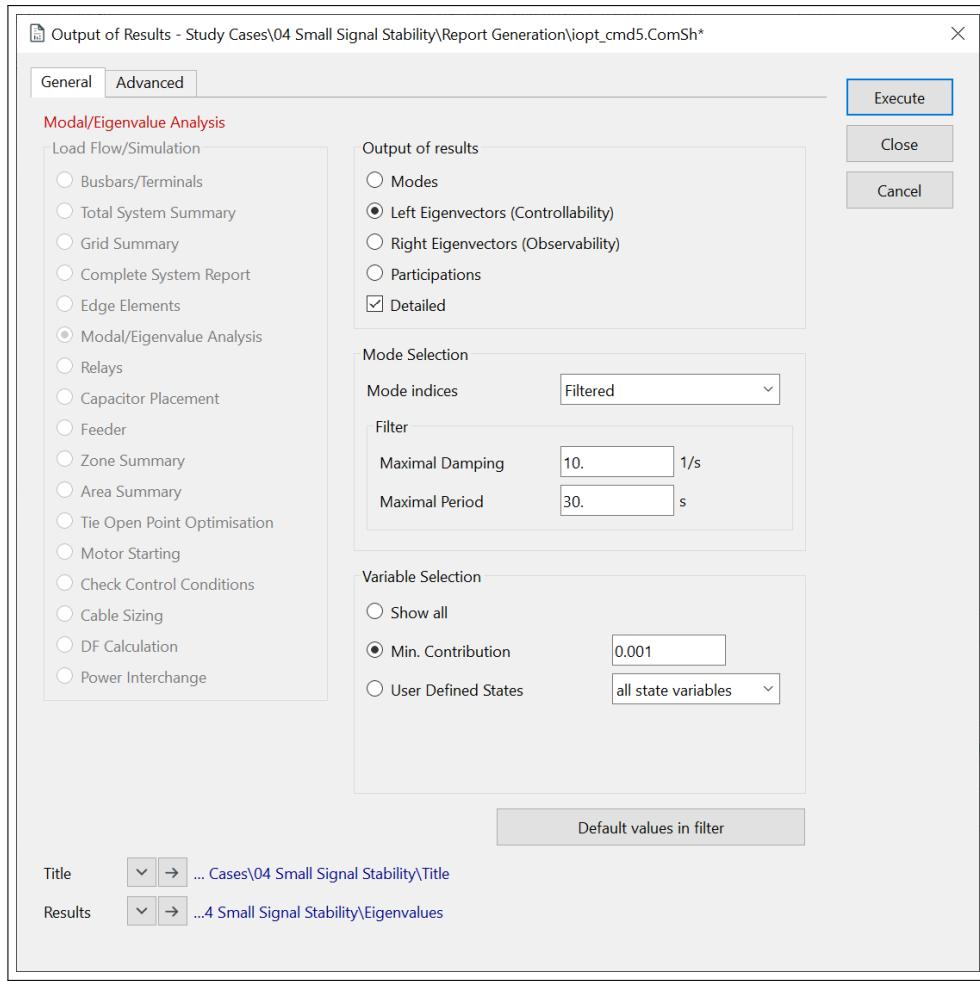


Figure 32.4.8: Output of Results Command

### 32.4.5 Modal Analysis Results in the Data Browser

The Data Manager and Network Model Manager can be used to view the participations, controllability or observability for power system elements such as synchronous machines after completing an modal/eigenvalues analysis. This option should only be used if none of the previous ways of presenting the results is satisfactory.

To use the data browsers to see the modal analysis results, the following steps should be followed after a modal analysis is executed:

1. Choose the mode index (eigenvalue) and state variable:
  - Click on the *Set Eigenvalue* icon  from the modal/eigenvalues analysis toolbar.
  - Choose the *Eigenvalue index* for which the results are to be displayed.
  - Choose the *State Variable* to view the results for by using the drop-down selection menu.
  - Press the **Execute** button. It will appear as if nothing has happened - this is normal.
2. View the results in the browser:
  - Select the synchronous machine icon from the object filter menu.
  - Define the flexible data page by clicking on the corresponding icon (.
  - Use the filter *Group* to get the *Results* and the filter *Calculation* to select *Modal/Eigenvalue Analysis*

- In the *Available variables* window, scroll to near the bottom until you see the variables Participation, Magnitude. Select this variable and all eight other variables down to Observability, Magnitude (signed).
  - Press the **OK** button. Now you can scroll to the right in the flexible data page to view the values of these variables.
- 

**Note:** The results can only be displayed for one eigenvalue and variable at a time. For instance, eigenvalue 3 and speed. Using the *Modal Analysis Results Command* (Section 32.4.1) is easier.

---

# Chapter 33

# Protection

## 33.1 Introduction

*PowerFactory* enables the user to define a protection scheme by integrating protective devices into the system defined by a project's network model. The software can be used to assist with the coordination of protective devices and for generating graphical representations of protection system characteristics. A relay model library is available in the "Protection Devices" folder of the *DlgSILENT* library which contains models of both generic and manufacturer-specific relays.

The following plot types are supported by *PowerFactory* to assist in the visualisation of the protection system performance:

- Current vs time plots (Time-overcurrent plots, Section 33.4)
- Impedance plots (R-X diagrams, Section 33.6)
- Relay operational limits plots (Section 33.7)
- Distance vs time plots (Time-distance diagrams, Section 33.8)
- Differential plots (Section 33.10)
- Short circuit sweep plots (Section 33.12)

Furthermore, *PowerFactory* allows for the creation of single line diagrams within which, the locations of the protection devices can be illustrated. (for general information on single line diagrams refer to Section 33.2.3)

The relay models in *PowerFactory* are fully integrated with many of the standard calculation functions included in the software. Notably:

- Load flow calculation (Chapter 25)
- Short circuit calculation (Chapter 26)
- Dynamic simulations using the RMS calculation or the EMT calculation (Chapter 29)
- Arc flash calculation (Chapter 34)

However, there are also a number of protection specific calculation commands available which are accessible via the Protection and Arc-Flash Analysis toolbox. The available commands are located in the sections of this chapter as follows:

- Short circuit sweep command (Section 33.11)
- Protection coordination assistant (Section 33.13)

- Protection Audit (Section 33.15)
- Short Circuit Trace (Section 33.16)
- Protection Graphic Assistant (Section 33.17)

This chapter will describe how to setup a network model for protection analysis, how to use *PowerFactory*'s protection analysis functions and plots and how to output results from the analysis in convenient settings reports.

The following section presents a general introductory overview of protection modelling within *PowerFactory*. Although it is not a pre-requisite for the user to have an understanding of the internal modelling approach in order to use *PowerFactory*'s basic protection functions, an understanding will help the user to appreciate the structure of the dialogs encountered when setting up a protection scheme. Users who wish to move straight into the creation of a protection scheme may wish to skip this section.

### 33.1.1 The modelling structure

Protection devices form a group of highly complex and non-uniform power system devices. Any program tasked with modelling these devices faces a difficult dilemma. On the one hand, the relay models should be as flexible and versatile as possible to ensure that all types of protection relays can be modelled with all of their features. On the other hand, the relay models should be as simple as possible to reduce the amount of work and knowledge needed to define power system protection devices.

This dilemma is solved in *PowerFactory* by modelling protection devices using a tiered approach consisting of three different levels. These levels are:

- the *relay frame*
- the *relay type*
- the *relay element*

Each of these levels fulfil a different role in the modelling of a protection device. Figure 33.1.1 shows the relation of these three levels graphically.

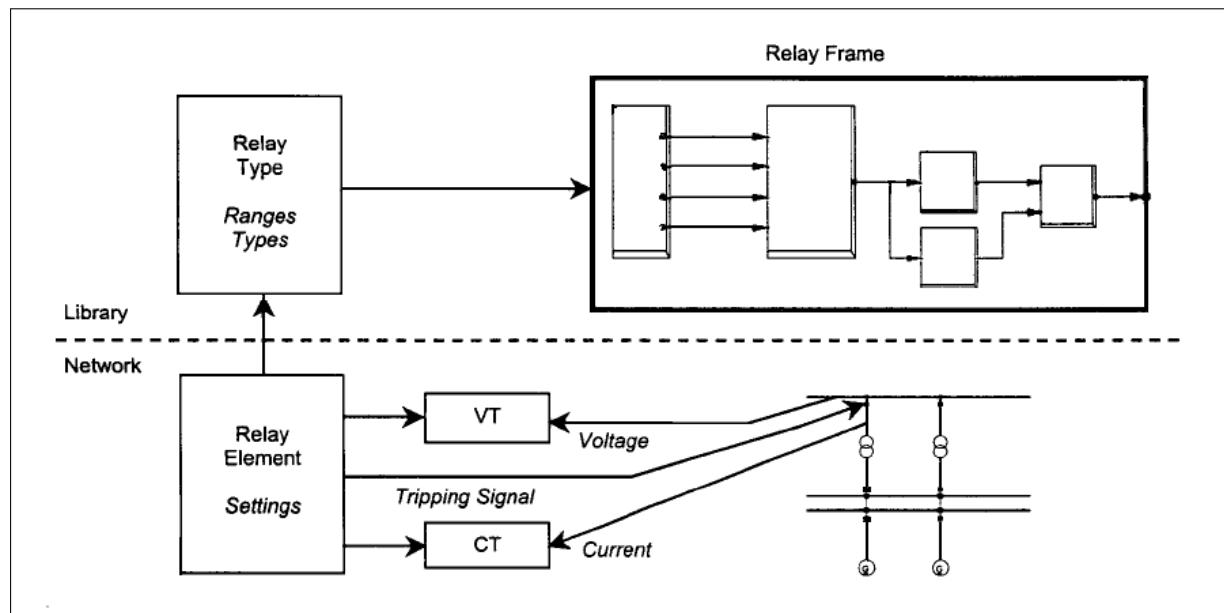


Figure 33.1.1: Modelling structure for protection devices

### 33.1.2 The relay frame

The relay frame specifies the general relay functionality using a diagram where functional blocks known as slots are connected by signals. Slots for timers, measurement and logic elements can be defined. It defines how many stages the relay consists of and how these stages interact. However, the relay frame contains no mathematical or logical functions, rather these are specified by the internal types referenced by the slots.

Each slot is defined by the number of input and output signals. The signal lines define how the slots are interconnected. Relay frames are similar to the frames of *Composite Models* and are created in the same way. See Chapter 30: Models for Dynamic Simulations, Section 30.2 (High-level Control System Representation) for more information. Figure 33.1.2 shows an example of a relay frame for a two stage overcurrent relay. The illustrated relay frame contains a measurement slot, two instantaneous overcurrent slots (each representing one stage of the overcurrent relay) and a logic slot. Connections between slots are illustrated by lines with arrowheads.

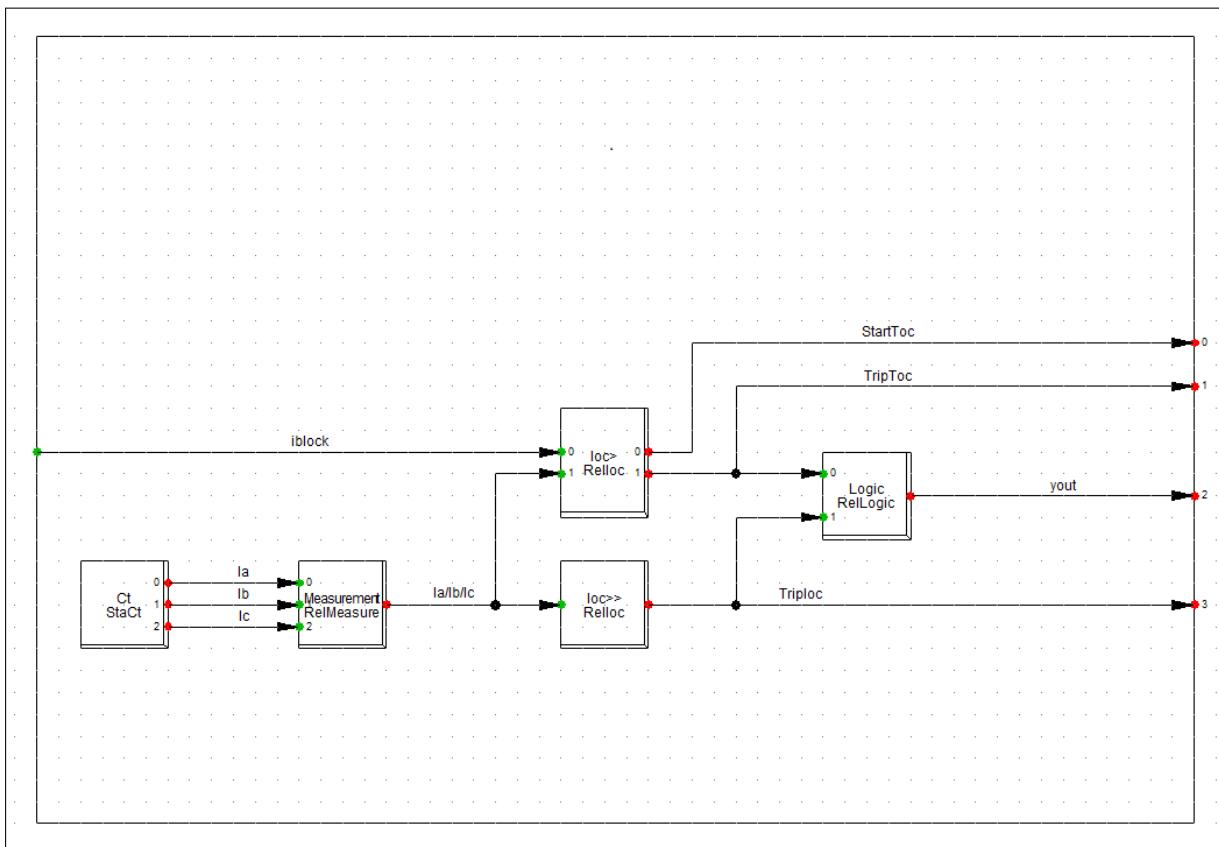


Figure 33.1.2: Typical relay frame

### 33.1.3 The relay type

The relay type associated with a specific relay frame, is defined by selecting a block definition for each slot of the frame. Assigning a block definition to a slot converts the slot to a block, representing a mathematical function which describes the behaviour of a physical element. For example, the type of filter used for processing the input signals, or the type of relay operating characteristic. Because many relays support more than one type of characteristic, a set of characteristics or functions can be defined. In addition, the relay type specifies the ranges for the various relay settings, including whether the parameters are set continuously or in discrete steps.

The relay type defines the library information for a specific manufacturer's relay, which does not yet

have any settings applied to it. The complete information described in the data sheet and manual is contained in the relay type. An advantage of this split concept is the possibility of re-using a relay frame for more than one relay type.

Figure 33.1.3 shows the type dialog associated with an instantaneous overcurrent slot as an example. Parameters that normally cannot be influenced by the user, like the Pick-up Time, are defined in the type as well.

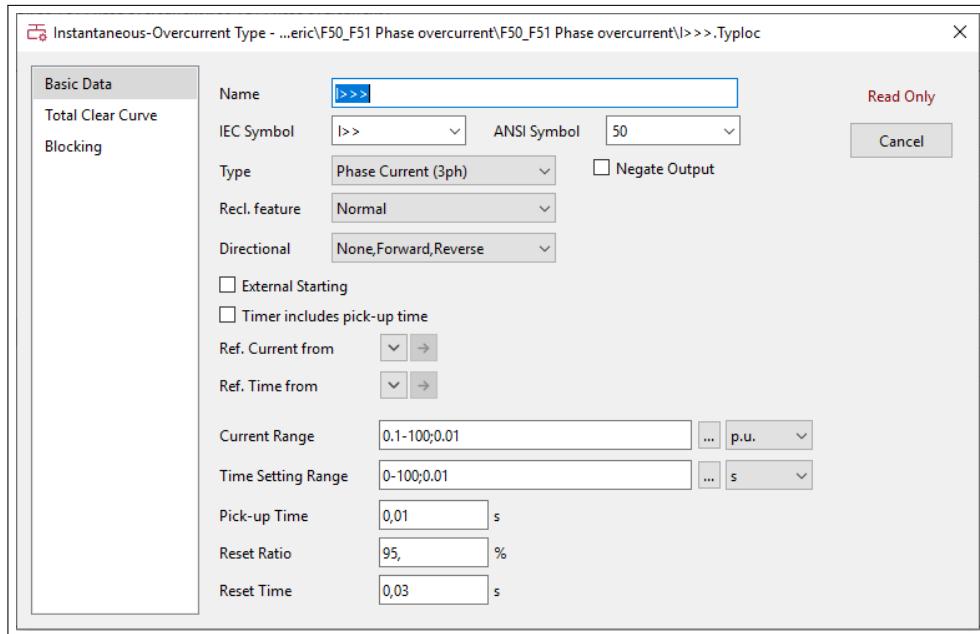


Figure 33.1.3: Type dialog of an instantaneous overcurrent block

### 33.1.4 The relay element

The relay element models the actual relay in a power system. It refers to a relay type in the library, which provides the complete relay structure including the setting ranges for all parameters. The actual settings of the relay, for example, the reach or the pick-up settings, are part of the relay element settings, considering the range limitations defined by the relay type.

CT and VT models are the input link between a relay element and the electrical network. For the relay output, a tripping signal is sent directly from the relay element to a circuit breaker in the network. To simulate busbar protection, differential protection, or tele-protection schemes, a relay element can operate more than one circuit breaker.

Figure 33.1.4 shows the block element dialog belonging to the type dialog in Figure 33.1.3.

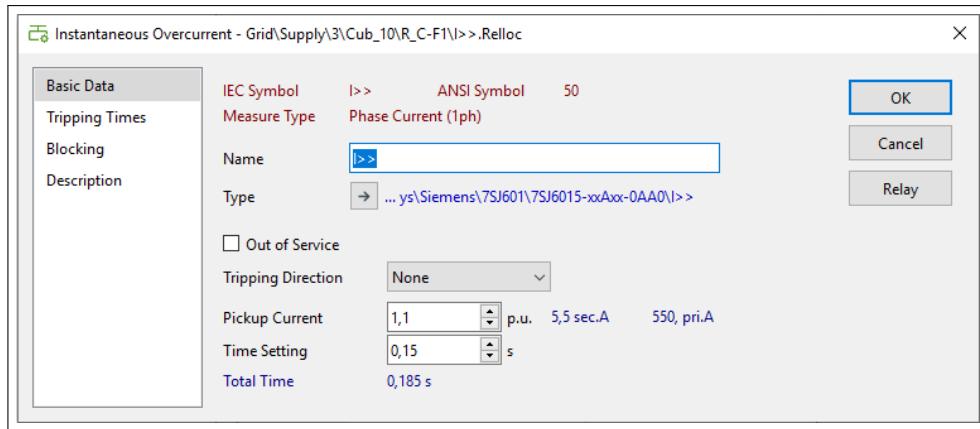


Figure 33.1.4: Element dialog of an instantaneous overcurrent relay

## 33.2 How to define a protection scheme in *PowerFactory*

This section describes the procedures necessary for defining a protection scheme within *PowerFactory*. It begins with a brief overview of the procedure followed by detailed instructions for how to define protection devices within the *PowerFactory* model.

### 33.2.1 Overview

Before starting the modelling of a protection scheme, it is necessary to construct a model of the network to be protected. See Section 12.2.

A protection scheme is defined by adding relays (or fuses) and their associated instrument transformers at appropriate places within the network model. After adding the device models, the settings can be manually adjusted, by using the automated coordination tools and plots, or by importing the relay settings directly from *StationWare* (refer to Section 24.15).

The *PowerFactory* protection modelling features have been designed to support the use of “generic” relays or “detailed” models of relays based on manufacturer specific devices.

For “generic” relays, *PowerFactory* includes the *DlgSILENT Library* containing some predefined generic relays, fuses and instrument transformers that can be used to design schemes without requiring specific details of a particular relay manufacturer’s range of products. This can be useful during the early stages of the definition of a protection scheme. By creating a model with generic protection devices, the user can confirm the general functionality of a scheme before relay procurement decisions are finalised.

For detailed definition and analysis of protection schemes, it is recommended to use detailed relay manufacturer specific models. *DlgSILENT* offers many such models in the global *DlgSILENT Library*. Of course, with thousands of different relay models in existence and more being created, in some instances a model will not exist. In such cases, advanced users can define their own relay models or contact *DlgSILENT* support for further advice.

The following section will explain how to add predefined protective devices (generic or manufacturer specific) to a network model.

### 33.2.2 Adding protective devices to the network model

Protection devices in *PowerFactory* must be placed within cubicles (refer to Section 4.6.4 for more information on cubicles). There are several methods to add or edit the protection devices in a cubicle:

1. Through the protection single line diagram. Refer to Section 33.2.3.
2. Right-clicking on a switch-symbol (New devices):
  - (a) Right-click a switch symbol in the single line graphic as illustrated in Figure 33.2.1. This will show a context menu.
  - (b) Choose *Devices* → *Relay/Fuse/Current Transformer/Voltage Transformer* → *New...*. A dialog for the chosen device will appear.
3. Right-clicking a switch symbol (Show All Devices):
  - (a) Right-click a switch symbol in the single line graphic as illustrated in Figure 33.2.1. This will show a context menu.
  - (b) Choose the option *Devices* → *Show All Devices*. A dialog showing the devices currently within the cubicle will appear.
  - (c) Click the  icon. A dialog will appear.
  - (d) Choose the desired device type.
  - (e) Click **OK** and the dialog for the new device will appear.
4. Through the protected device (line, transformer, load etc):
  - (a) Double-click the target element to protect. A dialog showing the device basic data should appear.
  - (b) Click the  button next to the end of element where you want to place the protective device. For a line element this will say “Terminal i/j” and for a transformer this will say “HV/LV-side”. A menu will appear. See Figure 33.2.2 for example.
  - (c) Click *Devices* → *Show All Devices*.
  - (d) Click the  icon. A dialog will appear.
  - (e) Choose the desired device type.
  - (f) Click **OK** and the dialog for the new device will appear.
5. Through the substation:
  - (a) Open a detailed graphic of the substation. Refer Section 12.2.7 for more information on substation objects.
  - (b) Right-click the specific part of the substation where you would like to add the relay. A context menu will appear. See Figure 33.2.3 for an example substation showing possible locations where protection devices can reside.
  - (c) Choose *Devices* → *Relay/Fuse/Current Transformer/Voltage Transformer* → *New...* or *Devices* → *Show All Devices* and follow the remaining steps from 2b or 3c respectively. The areas which can be right-clicked in a typical detailed substation graphic are ringed in Figure 33.2.3.

After completing one of the methods above, the created device also must be configured. Usually this involves selecting a type and entering settings. Further information about configuring overcurrent protection device elements is explained in Section 33.3 and distance protection devices in Section 33.5.

---

**Note:** When adding a protection device by right-clicking on a switch (Method 2), ensure the element connected to the switch is not already selected. Otherwise, you will create devices at both ends of the element. If you select the switch successfully, only half of the connected element will be marked when the context menu appears.

---

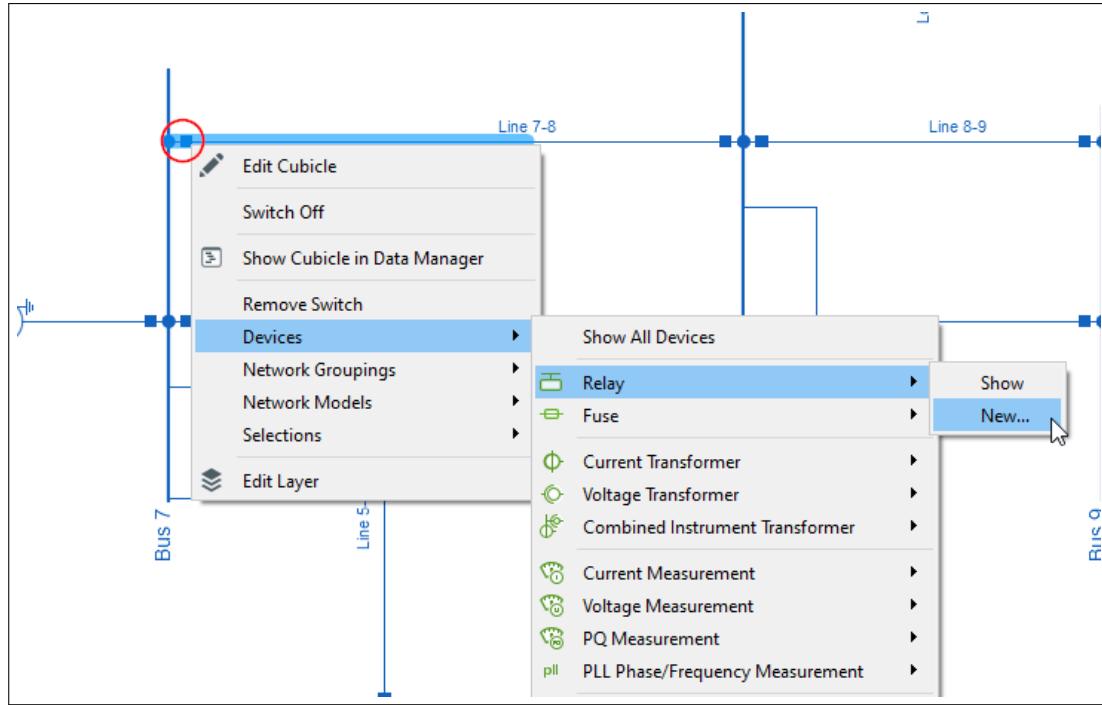


Figure 33.2.1: Adding a new relay to a single line diagram

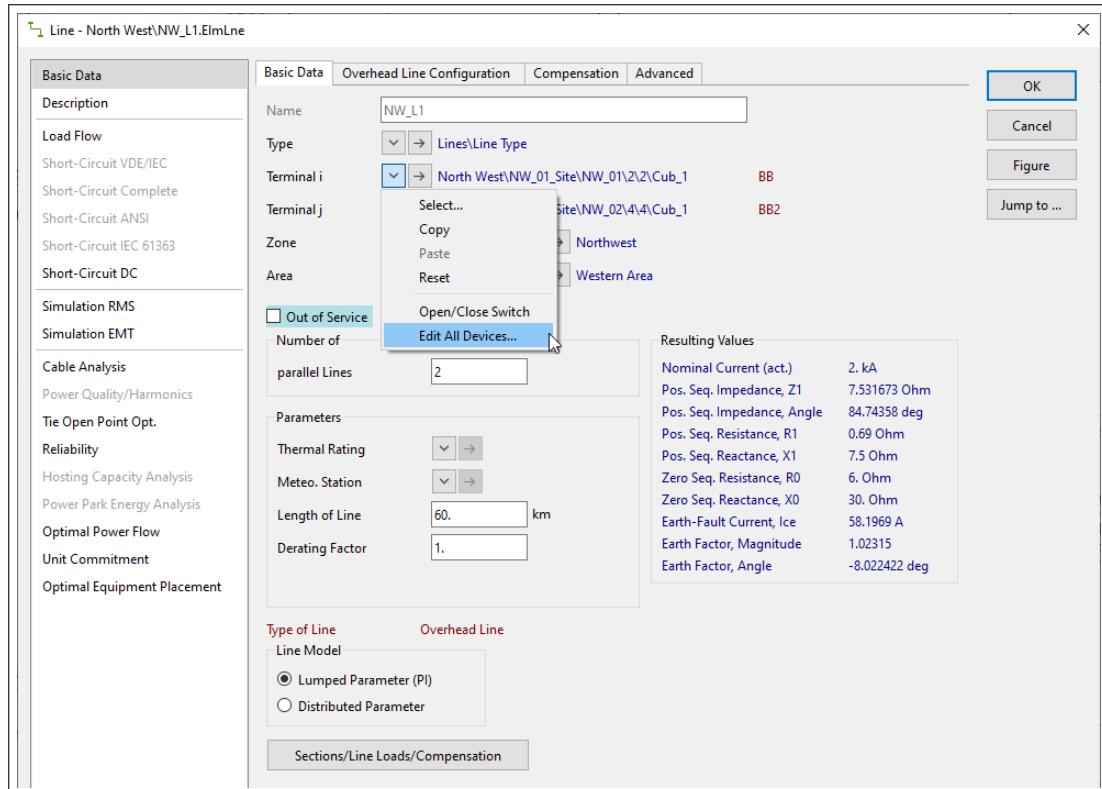


Figure 33.2.2: Editing line protection devices

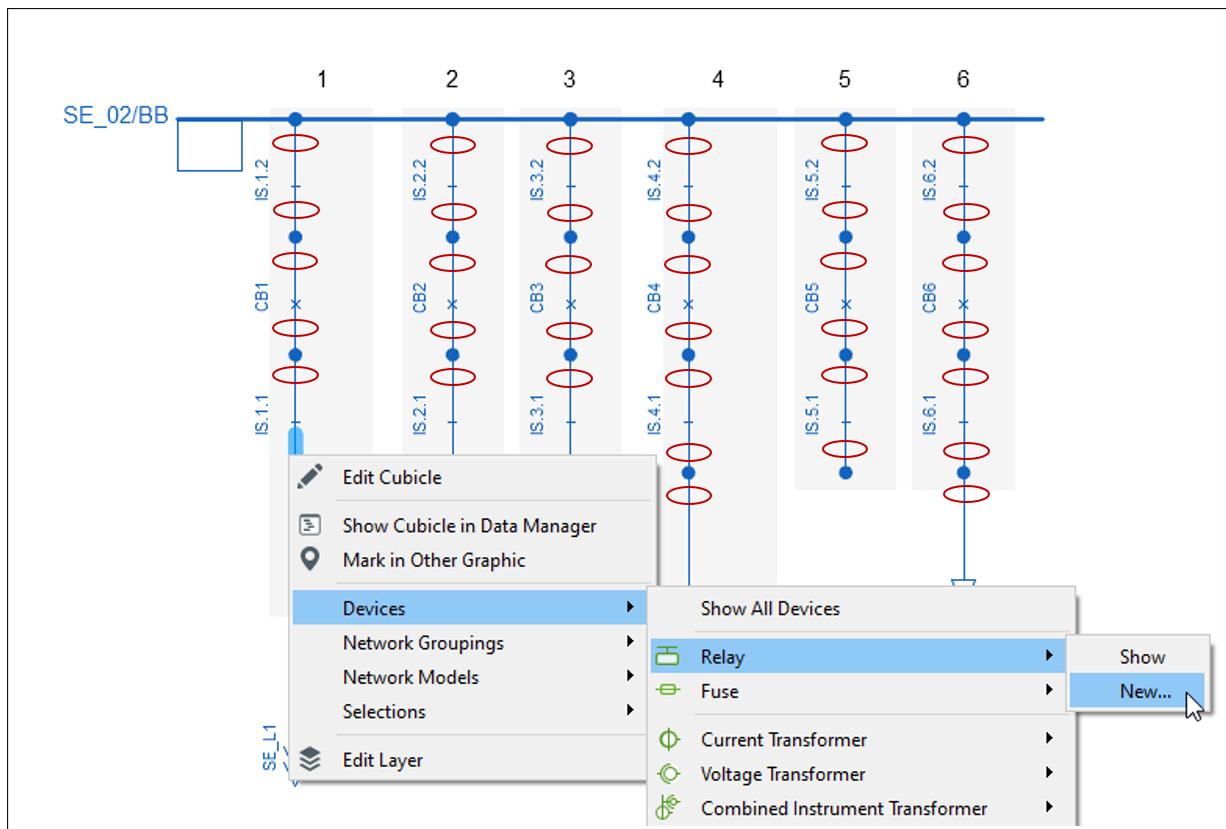


Figure 33.2.3: Adding a new protective device to a detailed substation graphic

### 33.2.3 Graphical representations of protection devices in single line diagrams

Depending on the task at hand it may be useful to show graphical representations of relay models in one or more single line diagrams. Users will commonly find themselves in one of two positions:

1) The network model already contains the protection devices of interest and the user simply want to show the already existing devices in the single line diagram.

or

2) The user is about to start the process of adding new protection devices to the network model and they wish to add graphical representations of the protection devices to the single line diagram as part of this set-up process.

In the first instance, existing protection devices located within cubicles can be manually or automatically added to the diagram using the *Diagram Layout Tool* tool (refer to Section 12.6).

In the second instance, protection devices can be added to the model as described in the remainder of this section.

#### 33.2.3.1 How to add relays to the protection single line diagram

To add a relay to the protection single line diagram:

1. Open an existing network diagram.
2. Click on the button on the drawing toolbar.

3. Click on the cubicle or switch where the relay should be placed.
  4. Optional: click and drag to reposition the relay to an alternative location.
- 

**Note:** The relay icon in the protection diagram can also be resized. Select the relay and then click and drag from the corner of the device.

---

### 33.2.3.2 How to add current transformers to the protection single line diagram

To add a current transformer to the single line diagram:

1. Open an existing network diagram.
  2. Click on the  button on the drawing toolbar.
  3. Click on the cubicle or switch where the device should be placed.
  4. Click on the relay to connect the secondary side of the CT.
- 

**Note:** Before placing current transformers in the single line diagram it is recommended to first place the relays to which the secondary side of the device has to be connected, in the diagram.

---

### 33.2.3.3 How to add voltage transformers to the protection single line diagram

To add a voltage transformer to the single line diagram:

1. Open an existing network diagram.
  2. Click on the  button on the drawing toolbar.
  3. Click on the bus where the primary side of the VT should connect.
  4. Click on the relay to connect the secondary side of the VT.
- 

**Note:** Before placing voltage transformers in the single line diagram it is recommended to first place the relays to which the secondary side of the device has to be connected, in the diagram.

---

### 33.2.3.4 How to add combined instrument transformer to the protection single line diagram

To add a combined instrument transformer to the single line diagram:

1. Open an existing network diagram.
  2. Click on the  button on the drawing toolbar.
  3. Click on the cubicle or switch where the device should be placed.
  4. Click on the relay to connect the secondary side of the CT.
  5. Click on the relay to connect the secondary side of the VT.
-

**Note:** Before placing combined instrument transformers in the single line diagram it is recommended to first place the relays to which the secondary side of the device has to be connected, in the diagram.

---

### 33.2.3.5 How to connect an instrument transformer to multiple relays

In some cases it might be desirable to connect a CT or VT to multiple relays. To do so follow these steps:

1. Open an existing network diagram.
2. Click on the  button on the drawing toolbar.
3. Click on the CT or VT that requires another connection.
4. Optional: Click at multiple points within the single line diagram to create a more complicated connection path.
5. Click on the target relay for the connection.

## 33.2.4 Locating protection devices within the network model

Protection devices can be added to the network model by placing them in the single line diagram directly as described in Section [33.2.3](#). However, in cases where the devices are not drawn directly in the single line diagram, there are several methods to highlight the location of the devices in the single line diagram. This section describes these methods.

### 33.2.4.1 Colouring the single line diagram to show protection devices

The single line diagram can be coloured to indicate the location of protective devices. To do this:

1. Click the  button on the graphics toolbar. The diagram colouring dialog will appear.
2. Check the box for *3. Other*.
3. Select *Secondary Equipment* from the first drop down menu.
4. Select *Relays, Fuses, Current and Voltage Transformers* from the second drop down menu.
5. Click **OK** to update the diagram colouring. The cubicles containing protection devices will be coloured according to the legend settings.

### 33.2.4.2 Locating protective devices using the Network Model Manager

To locate protection devices using the Network Model Manager follow these steps:

1. Open the Network Model Manager by clicking on the  icon from the main toolbar.
2. Click  for relays,  for fuses,  for current transformers or  for voltage transformers on the left side of the window to show a list of all calculation relevant objects of one class in a tabular list on the right side of the window.
3. Right-click the icon of one or more objects in the list. A context menu will appear.
4. Select *Mark in Graphic*. The cubicle/s containing the object/s will be highlighted in the single line diagram.

### 33.3 Basics of an overcurrent protection scheme

Section 33.2.2, explained the initial steps required to add a protective device to the network model. When a new device is created within a network model there are a number of parameters to define in the dialog which appears. This section will describe the basic steps required to specify these parameters for overcurrent relays and fuses.

The following section, 33.4 describes the use of the main tool for analysing overcurrent protection schemes, the time-overcurrent diagram.

#### 33.3.1 Overcurrent relay model setup - basic data page

The basic data page in the relay model (*ElmRelay*) dialog is where the basic configuration of the relay is completed. Generally it is required to complete the following steps:

- Select the relay type (generic or manufacturer specific). Refer to Section 33.3.1.1.
- Select the instrumentation transformers. Refer to Section 33.3.1.2
- Enter the relay settings. Refer to Section 33.3.1.3.

##### 33.3.1.1 Selecting the relay type

To select a generic relay type from the relay basic data page:

1. Click on the  icon. A menu will appear.
2. Choose *Select Type....*. A data page will appear. Click on the button **DIGSILENT Library** to access the global library on the left side of the page.
3. Navigate into the sub-folders under “Relays” and select the required relay type.
4. Click **OK** to assign the relay type. Note that the basic data page of the relay will now show many different slots which are based on the configuration of the relay type.

To select a project relay type from the relay basic data page:

1. Click on the  icon. A menu will appear.
2. Choose *Select Type....*. A data page will appear. Click on the button **Project Library** to access the project library on the left side of the page.
3. Locate the relay either within the local library, or within any “Relay Library” in the user area.
4. Click **OK** to assign the relay type. Note that the basic data page of the relay will now show many different slots. These are the functional protection blocks such as time-overcurrent, measurement, differential, impedance and so on, which contain the relay settings. The number and types of slots within the relay is determined by the relay type selected.

##### 33.3.1.2 Selecting the relay instrument transformers

If there were some instrument transformers within the cubicle when the relay was created, then these will automatically be assigned to the appropriate slots within the relay. However, if it is desired to select an alternative instrument transformer then follow these steps:

1. Right-click the cell containing the instrument transformer. A menu will appear.

2. Choose *Select Element/Type*.... A data browser will appear showing the contents of the relay cubicle.
3. Select an alternative instrument transformer here, or navigate to another cubicle within your network model.
4. Click **OK** to choose the instrument transformer.

If the cubicle where the relay was created does not contain any current transformers, then a **Create CT** will appear at the bottom of the dialog. If the relay also has at least one VT slot, a **Create VT** button will also appear. By clicking on these buttons it is possible to create a VT or CT and have them automatically assigned to vacant slots within the relay. For instructions for configuring a CT refer to Section 33.3.3 and for configuring a VT refer to Section 33.3.4.

### 33.3.1.3 Entering the relay settings

To edit relay settings:

1. Locate the desired slot that you would like to modify. You may need to scroll down to locate some slots in complicated relay models.
2. Double-click the target slot. The dialog for the clicked element will appear.
3. Enter or modify the settings.

### 33.3.1.4 Other fields on the relay basic data page

There are several other fields on the relay basic data page:

**Application** This field is for documentation and searching purposes only.

**Device Number** This field is also for documentation and searching purposes only.

**Location** By default these fields give information about the relay location within the network model based upon the cubicle that it is stored within. However, it is possible to select an alternative *Reference* cubicle. If an alternative reference cubicle is selected, then the relay will control the switch within this cubicle. Furthermore, changing the reference location will also affect the automatic assignment of instrument transformers and the cubicle where any instrument transformers created using the **Create VT** or **Create CT** buttons will be placed.

## 33.3.2 Overcurrent relay model setup - max/min fault currents tab

This tab can be used to record the minimum and maximum fault currents likely to be observed at the relay location. This information can be useful when coordinating the relay characteristic using a time overcurrent plot. The minimum values are stored for reference purposes only. In the case of the maximum currents these settings can be used to limit the extent to which the tripping characteristics of the relay will be shown in time overcurrent plots. For example if a relay is likely to see a maximum current of 50kA there is little benefit in showing the tripping time of the relay at values above 50kA. This extended part of the characteristic is extraneous and will simply clutter the plot. This visualisation is a property of the time overcurrent plot itself and can be configured using the *Cut Curves At* setting in the time overcurrent plot settings dialog described in Section 33.4.7.

The values can be entered either manually or calculated automatically as follows. For maximum currents the *Calculate Max. Fault Currents* button can be pressed, in which case PowerFactory will automatically calculate maximum short circuit currents at the local busbar according to the short circuit method specified in the referenced *Short-Circuit Command*. For Maximum Phase fault current, a 3

phase fault current will be used, whilst for the maximum earth fault current a single phase to ground fault will be used. The calculated currents will automatically be assigned to the corresponding parameters.

For the minimum currents, the individual *Get I branch* buttons can be pressed. When these buttons are pressed the current flowing in the branch during the active calculation will be extracted and assigned to the corresponding parameter in the ElmRelay object. For example, if a load flow calculation is executed before pressing the button, then the load flow current flowing in the relay's branch will be entered as the minimum current.

The *Coordination Paths* section of the dialog provide options based on a topological search extending outwards from the relay location which can be used to differentiate between network elements located in the forward direction relative to elements located in the reverse direction. The *show* buttons bring up a browser containing network elements located in the associated direction, whilst the *mark* buttons can be used to mark those elements in single line diagrams.

### 33.3.3 Configuring the current transformer

A new current transformer (CT) can be created as described in Section 33.2.2 (Adding protective devices to the network model). Alternatively a CT can be created by using the **Create CT** button in the relay model dialog.

The process of configuring the CT is:

1. Select/Create the CT type. Refer to Section 33.3.3.1 for information about the CT type.
2. Optional: If you would like to setup the CT to measure a location other than its parent cubicle or as an auxiliary CT, you can choose this through the *Reference* parameter. Refer to Section 33.3.2 for further instructions.
3. Optional: Alter the *Orientation*. Positive current is measured when the flow is away from the node towards the branch and the *Orientation* is set to *Branch*.
4. Set the *Primary* ratio through the drop down menu next to *Tap*. The available ratios are determined by the selected CT type. If you choose the option *Detect primary tap*, PowerFactory will try to automatically determine an appropriate tap based on the rating of the associated primary equipment. If no type is selected the only ratio available will be 1A.
5. Set the *Secondary* ratio through the drop down menu next to *Tap*. The available ratios are determined by the selected CT type.
6. Optional: Select the number of phases from the drop down menu next to *No. Phases*.
7. Optional: Choose a Y or D connection for the secondary side winding. This field is only available for a 2- or 3-phase CT.
8. Optional: If the CT is 1 or 2-phase, the measured phases must be selected. These can be:
  - a, b or c phase current;
  - $N = 3 \cdot I_0$  (1-phase CT's only); or
  - $I_0$  (1-phase CT's only)
9. Optional: If the CT is 3-phase, select the *Phase Rotation*. This defines how the phases of the secondary side map to the phases of the primary side. For example, if you wanted the A and B Phases to be flipped on the secondary side of the transformer, then you would choose a *Phase Rotation* of "b-a-c".
10. Optional: If the transformer includes additional cores, these can be modelled by selecting the *Add Core* button. Added *Current Transformer Core* objects, will be stored inside the parent current transformer object and can be easily accessed either by pressing the *Edit Cores* button or by navigating and editing entries in the *Additional Cores* table which appears when the first core is added. Additional core *StaCtcore* objects should reference objects of the type class *TypCtcore*, see Section 33.3.3.1. If no type is selected a ratio of 1:1 is assumed.

If it is desired to model CT saturation, saturation information about the CT can be entered on the “Additional Data” page of the CT element. This information is used only when the “detailed model” tick box is selected, otherwise it is ignored by the calculation engine.

### 33.3.3.1 Configuring the current transformer type

The current transformer type dialog defines the range of ratios available to a referencing CT object. The information about the connection of phases (Y or D) is defined in the CT element as discussed in Section 33.3.3.

A CT element *StaCt* can be linked to two alternative type classes. The standard CT type class is *TypCt* but alternatively a *current transformer core* type class *TypCtcore* can be selected. Both type classes are suitable for most purposes and can both be used to represent multicore CTs as well as single core CTs both multi and single phase. However, there may be circumstances, for example for data exchange between *PowerFactory* and other software or vice versa where one type class may be preferred over the other. Please note the following key differences between the two representations:

1. For a *TypCt* the represented winding ratios depend on the specified primary and secondary tap ratings. The specific ratio in use depends on the user's selection in the associated CT element object *StaCt*.
2. For a *TypCtcore* the represented winding ratios are independent of the primary rating. The winding ratios instead depend on the relevant ratio settings and the specified secondary rating. The specific ratio in use depends on the user's selection in the associated CT element object *StaCt*.
3. A CT element object (*StaCt*) can refer to either a *TypCt* or a *TypCtcore* object.
4. A *StaCtcore* object can only refer to a *TypCtcore* type object. It cannot refer to a *TypCt* type object.
5. A *StaCt* object can store additional *StaCtcore* objects within it. This is not the case for *StaCtcore* objects.

To add additional *Primary* or *Secondary Taps* for a *TypCt* object:

1. Right-click one of the cells in the tabular list of available taps. A menu will appear.
2. Choose *Insert Row/s*, *Append Row/s* or *Append n Rows* to add one or more rows to the table.
3. Enter the ratio of the tap. Note the tabular list must be in ascending order.
4. On the additional information page update the fields with the datasheet data.

To add additional ratio options for a *TypCtcore* object:

1. Specify a primary and secondary current rating for the device as a whole.
2. Right-click in one of the cells or in empty space in the tabular list of available ratios. A context menu will appear.
3. Choose *Insert Row/s*, *Append Row/s* or *Append n Rows* to add one or more rows to the table.
4. Enter the new ratio to be considered. Note the tabular list must be in ascending order.
5. On the additional information page update the fields with the datasheet data.

For the additional data page the following accuracy parameters can be selected:

- The accuracy class
- The accuracy limit factor
- either
  - The apparent power (acc. to IEC)
  - The burden impedance (ANSI-C)

- The voltage at the acc. limit (ANSI-C)

For more information on the CT model please refer to the measurement devices section of the [Technical References Overview Document](#).

### 33.3.3.2 Configuring a CT as an auxiliary unit, measuring circulating current in the delta winding of a 3 winding transformer or changing the measurement location

By default the CT measures the current within its parent cubicle. The *Location* fields *Busbar* and *Branch* show information about the measurement location automatically. However, it is possible to configure the CT to measure current in a different location. To do this:

1. Click the  icon next to *Reference*. A data browser will appear.
2. Select either another cubicle, a switch, a three winding transformer or another CT where you would like to measure current.

If you install a current transformer in one of the connecting cubicles of a delta winding of a three winding transformer then if required, you can select the transformer itself as the reference location. The CT will then measure the circulating current in the corresponding delta winding of the transformer instead of measuring the line currents or I<sub>0</sub> currents derived from the line currents. In this case the CT will interact with the I<sub>0delta\_</sub> signals of the winding. With such a connection it is assumed that the CT will be single phase and measuring I<sub>0</sub> so there is no need to for any further configuration of the device.

If you select another CT, then this CT becomes an auxiliary CT with the final ratio from the primary circuit to the CT secondary side the product of the ratios of the two CTs - this is indicated in the field *Complete Ratio*. If you select another cubicle or switch then the CT will measure current at location of the selected switch or cubicle.

### 33.3.4 Configuring the voltage transformer

A voltage transformer (VT) can be created as described in Section [33.2.2](#). Alternatively a VT can be created by using the [Create VT](#) button in the relay element dialog.

The process of configuring the VT is then:

1. Select the VT type. Refer to Section [33.3.4.1](#) for information about the VT type.
2. Optional: Select the *secondary* winding type. If no type is selected, the available ratios will be 1, 100, 110, 120 and 130. More information about the secondary type can be found in Section [33.3.4.2](#).
3. Optional: If you would like to setup the VT to measure a location other than its parent cubicle or as an auxiliary VT, you can choose this through the *Reference* parameter. Refer to Section [33.3.4.4](#) for further instructions.
4. Set the *Primary* voltage rating through the drop down menu in the *Tap selection* field. The available ratios are determined by the selected VT type. If no type is selected, the available ratios will be 1, 100, 110, 120 and 130.
5. Set the *Secondary* voltage rating through the drop down menu in the *Tap Selection* field. The available ratios are determined by the selected secondary winding type.
6. Specify the number of phases for the VT. If an open delta VT arrangement as illustrated in Figure [33.3.1](#) is required then the number of phases should be specified as 1 and N chosen for the phase.
7. Optional: For 3 phase VTs Choose a Y or V (broken delta) connection for the winding connection. A V connection is illustrated in Figure [33.3.2](#). Note that this selection effectively configures both

the primary and secondary connections of the VT since it is assumed that a V connected primary winding will always correspond with a V connected secondary, likewise a Y connected primary winding will always correspond with a Y connected secondary.

8. Optional: Click **Additional Secondary Windings** to open a dialog where extra secondary windings can be added. See Section [33.3.4.3](#) for more information about configuring additional secondary windings.

When a VT is created it is stored in the cubicle that was right-clicked or the cubicle the relay is stored in.

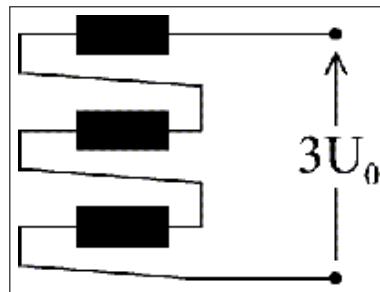


Figure 33.3.1: The open delta (O) winding connection

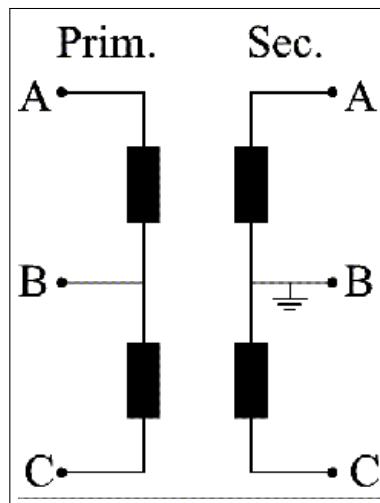


Figure 33.3.2: The V winding connection

#### 33.3.4.1 The voltage transformer type

The voltage transformer type defines the datasheet data of the voltage transformer along with a range of potential voltage ratings for the primary winding. The voltage transformer can be configured as:

- **Ideal Voltage Transformer.** In this case no saturation or transformer leakage impedance values are considered and the voltage transformer has a perfect transformation of primary measured values into secondary quantities.
- **Voltage Transformer.** In this case saturation and transformer leakage effects are modelled according to data entered on the *Transformer Data* page.
- **Capacitive Voltage Transformer.** In this case, the VT is modelled as a CVT according to the parameters entered on the *Transformer Data* and *Additional CVT Data* page.

For more information on the VT model please refer to the measurement devices section of the [Technical References Overview Document](#).

To configure additional *Primary Taps*:

1. Right-click one of the cells in the tabular list of available taps. A menu will appear.
2. Choose *Insert Row/s*, *Append Row/s* or *Append n Rows* to add one or more rows to the table.
3. Enter the ratio of the tap. Note the tabular list must be in ascending order.

#### 33.3.4.2 Configuring the secondary winding type

The secondary winding is defined by the secondary winding type, and is similar to the primary VT type where multiple *Secondary Tap* ratios can be defined. If a secondary winding is not selected, it has the standard tap settings of 1, 100, 110, 120 and 130 V available.

The burden and power factor on this page are not calculation relevant and for information purposes only. Therefore, the secondary winding type is always treated as an ideal transformer.

#### 33.3.4.3 Additional VT secondary winding types

In some cases a VT has multiple secondary windings. For example, some VTs might have a regular winding and then also an 'open delta' winding for measuring the zero sequence voltage. It is possible to configure a *PowerFactory* VT in the same way. To define an additional secondary winding type:

1. Click the VT element **Additional Secondary Windings** button.
2. Click the  button. A dialog for the Secondary Voltage Transformer will appear.
3. Click the  button.
4. Choose *Select Type...* and click on the **Project Library** button. The type is a secondary winding type as described in Section [33.3.4.2](#).
5. Choose the *Tap*.
6. Select the *Connection*.

#### 33.3.4.4 Configuring the VT as an auxiliary VT or changing the measurement location

By default the VT measures the voltage within its parent cubicle. The *Location* fields *Busbar* and *Branch* show information about the measurement location automatically. However, it is possible to configure the VT to measure in a different location. To do this:

1. Click the  icon next to *Location*. A data browser will appear.
2. Select either another cubicle, a bus or another VT where you would like to measure voltage. If you select another VT, then this VT becomes an auxiliary VT with the final ratio from the primary circuit to the VT secondary side the product of the ratios of the two VTs - this is indicated in the field *Complete Ratio*. If you select another cubicle or busbar then the VT will measure voltage at location of the selected switch or cubicle.

#### 33.3.5 Configuring a combined Instrument transformer

A combined instrument transformer is configured in much the same way as the standard current transformer and voltage transformer models described in the previous sections. The main difference is that this model combines references to CT and VT type class objects within a single station element class namely the *StaCombi* class.

This has various benefits:

1. Real network combined instrument transformers can be more closely represented.
2. Facilitates easier data exchange between *PowerFactory* and other softwares.
3. Simplification of the network model is possible by grouping CTs and VTs into a single element object.

Reference to the previous sections [33.3.3](#) and [33.3.4](#) provide detailed handling descriptions which can be easily extrapolated for application to this model.

### 33.3.6 How to add a fuse to the network model

In *PowerFactory* the fuse element operates to some extent like an inverse time over-current relay with a 1/1 CT. The fuse will “melt” when the current in the fuse element exceeds the current specified by the fuse’s melt characteristic.

To add a fuse to the network model:

1. Either:
  - (a) Right-click a target cubicle and select the option *Devices* → *Fuse* → *New...*. This is an internal (or implicit fuse) located within the cubicle. Or:
  - (b) Add an explicit fuse model to the network by clicking the  and connecting the device as you would connect a line or transformer.
2. On the fuse dialog, click the  button and choose *Select Type...*. A data page with the following buttons will appear, select either:
  - (a) *DlgSILENT Library*. A global library will appear on the left side of the page showing you a list of built-in fuses where you can select an appropriate one; or
  - (b) *Project Library*. A dialog will appear showing you the local project library where you can choose a fuse type that you have created yourself.
3. Adjust other options on the basic data page. The options are as follows:

**Closed** If this is checked, the fuse will be in the closed (non melted) state for the calculation.

**Open all phases automatically** If this option is enabled, then should the fuse be determined to melt, *PowerFactory* will automatically open all three phases on the switch during a time domain simulation or short circuit sweep. This field has no effect on the load-flow or short-circuit calculations.

**No. of Phases** This field specifies whether the fuse consists of three separate fuses (3 phase), two fuses (2 phase) or a single fuse (1 phase). Note, when the one or two phase option is selected and the fuse is modelled explicitly in the network model, the actual phase connectivity of the fuse is defined within the cubicles that connect to the fuse. When the fuse is modelled implicitly, a selection box will appear that allows you to select which phase/s the fuse connects to.

**Fuse type** This field is used for information and reporting purposes only.

**Device Number** This field is used for information and reporting purposes only.

**Compute Time Using** Many fuses are defined using a minimum melt curve and a total clear curve as illustrated in Figure [33.3.3](#) - the idea is that for a given current, the fuse would generally melt at some time between these two times. In *PowerFactory* it is possible to choose whether the trip/melt time calculations are based on the minimum melt time or the total clear time.

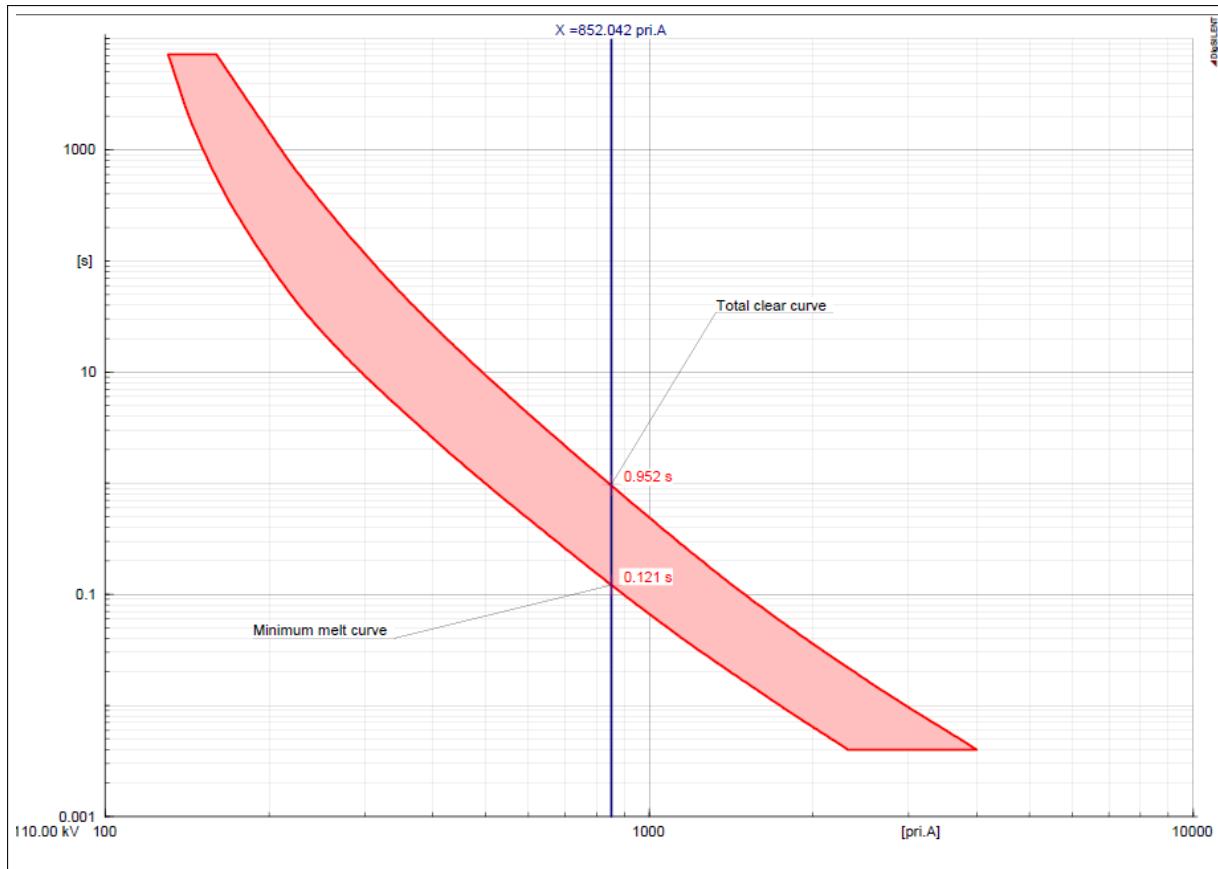


Figure 33.3.3: Fuse melt characteristics

### 33.3.6.1 Fuse model setup - other pages

On the VDE/IEC Short-Circuit and Complete Short-Circuit pages there is the option to configure the fuse *Break Time*. This variable is used in the short circuit calculation of “lb” when the *Used Break Time* variable is set to *local*, or *min. of local*. Refer to Chapter 26 for more information on the calculation of short circuits in *PowerFactory*.

On the Optimal Power Flow page, there is the option *Exclude from Optimisation* which if checked means that the fuse will be ignored by the OPF and open tie optimisation algorithms. See Chapter 42 for further information.

On the *Reliability* page, the fuse can be configured for Fault separation and power restoration. These options are explained in detail in Chapter 47.

### 33.3.7 Basic relay blocks for overcurrent relays

Section 33.1 explained that all relay models contain slots which are placeholders for block (protection function) definitions. There are many types of protection blocks in *PowerFactory* and each type has a different function. Furthermore, there are various options and parameters within each of these blocks that enable mimicking in detail the functionality offered by many relays. The relay model is completed by interconnecting these different slots containing block definitions in various ways. Hence it is possible to produce relay models with a large variety of operating characteristics. Advanced users are able to define their own types of protection device. The creation of user defined protection devices is covered in the Section 33.18.

The blocks contained within a relay are listed in the slot definition section of the relay model dialog. In

general the user will need to define parameters within these relay blocks. The settings dialog can be reached by double clicking on the block of interest in the net elements column.

If the user is interested in viewing a graphical representation of the interconnection of slots for a particular relay then the user should navigate to relay type of interest in the Data Manager. By right-clicking on this relay type icon and selecting *Diagrams* → *Show Diagram*, a graphical representation of the relay frame will appear in a new window.

The following sections provide a brief overview of some of the basic protection blocks that can be used to develop a relay model in *PowerFactory*. For more information on these blocks please refer to the [Protection Devices Library](#).

### 33.3.7.1 The measurement block

The measurement block takes the real and imaginary components of the secondary voltages and currents from the VTs and CTs, and processes these into the quantities used by other protection blocks in the relay model. Quantities calculated by the measurement block include absolute values of each current and voltage phase and the positive and negative sequence components of voltage and current.

Depending on how the measurement block type is configured, it also allows for the selection of different nominal currents and voltages. For example, this feature can be utilised to support relays that have both 1A and 5A versions. If a relay does not need a nominal voltage, for instance an overcurrent relay without directional elements, or if there is only one nominal value to choose from, the nominal voltage and/or current selection field is disabled.

For EMT simulations, the measurement block type can also be configured for different types of signal processing. This determines what type of algorithm is used for translating the input current and voltage waveforms into phasor quantities for use by the protection blocks. Various DFT and FFT functions along with harmonic filtering are available.

### 33.3.7.2 The directional block

A detailed discussion of the principles of directional protection is outside the scope of this user manual. The reader is encouraged to refer to a protection text for more information on the general principles. A very brief high level overview is presented in the following paragraphs.

In *PowerFactory*, there are two directional blocks the “RelDir” and the “RelDisDir”. The “RelDir” block is the basic direction block and is typically used by over-current relay models to determine the direction of the current flow. It provides a forward or reverse direction determination signal which can be fed into subsequent overcurrent blocks. The block can also send a trip signal.

In its normal operating configuration, the block is determining the direction by comparing the angle between a “polarisation” voltage and an “operating” current phasor. Various polarisation methods are supported by the block including common ones such as self and cross polarisation. The block also has a so-called Maximum Torque Angle (MTA). This is the angle by which the polarising voltage is rotated. Consequently, the forward direction is determined by the MTA  $\pm$ Angle Operating Sector (often 180°). This principle is illustrated in Figure 33.3.4.

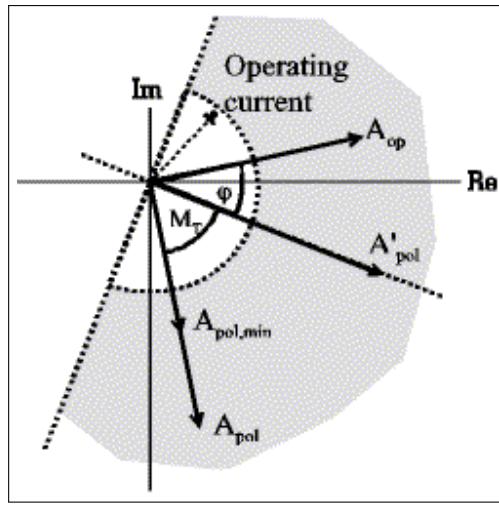


Figure 33.3.4: Directional relay principle diagram

The polarisation quantity  $A_{pol}$  is rotated over the angle  $M_T$  (MTA). The rotated polarisation quantity  $A'_{pol}$   $\pm$  AOS defines a half plane which forms the forward operating plane. The block will produce a tripping signal if the operating quantity is detected in the selected direction, and if it exceeds the threshold operating current, illustrated by the semi-circle Figure 33.3.4.

The second type of directional block in *PowerFactory* is the “RelDisDir”, this is normally used with distance protection relays and is discussed in Section 33.5.3.8.

### 33.3.7.3 The instantaneous overcurrent block

The instantaneous overcurrent block is a protection block that trips based on current exceeding a set threshold (pickup current setting). The block also supports the inclusion of an optional delay time and directional features. Hence this block can be used to represent instantaneous, definite time and directional overcurrent relay functionality. The available setting ranges for the pickup and the time delay are defined within the type. The relay characteristic is shown in Figure 33.3.5. The total tripping time is the sum of the delay time and the pickup time also configured within the relay type.

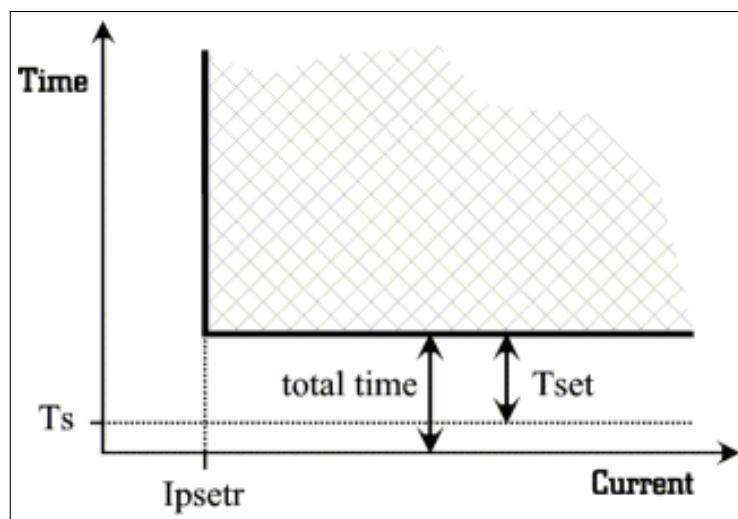


Figure 33.3.5: Instantaneous overcurrent tripping area

The block will not reset until the current drops under the reset level, which is specified by the relay type

in percent of the pickup current:  $I_{reset} = I_{set}K_r/100\%$ . See Figure 33.3.6 for a typical timing diagram.

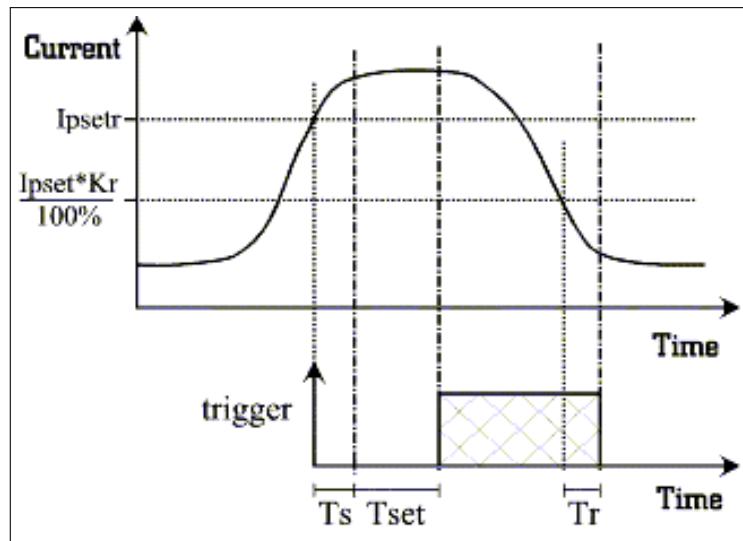


Figure 33.3.6: Instantaneous overcurrent timing diagram

#### 33.3.7.4 The time overcurrent block

The time-overcurrent block is a protection block that trips based on current exceeding a threshold defined by an I-t characteristic. Most relays support the selection of several different I-t characteristics. These characteristics can be shifted for higher or lower delay times by altering the time settings or shifted for higher or lower currents by altering the pickup current. The ranges for these two settings and the characteristics of the I-t curve are defined within the block type. Typical curves are shown in Figure 33.3.7.

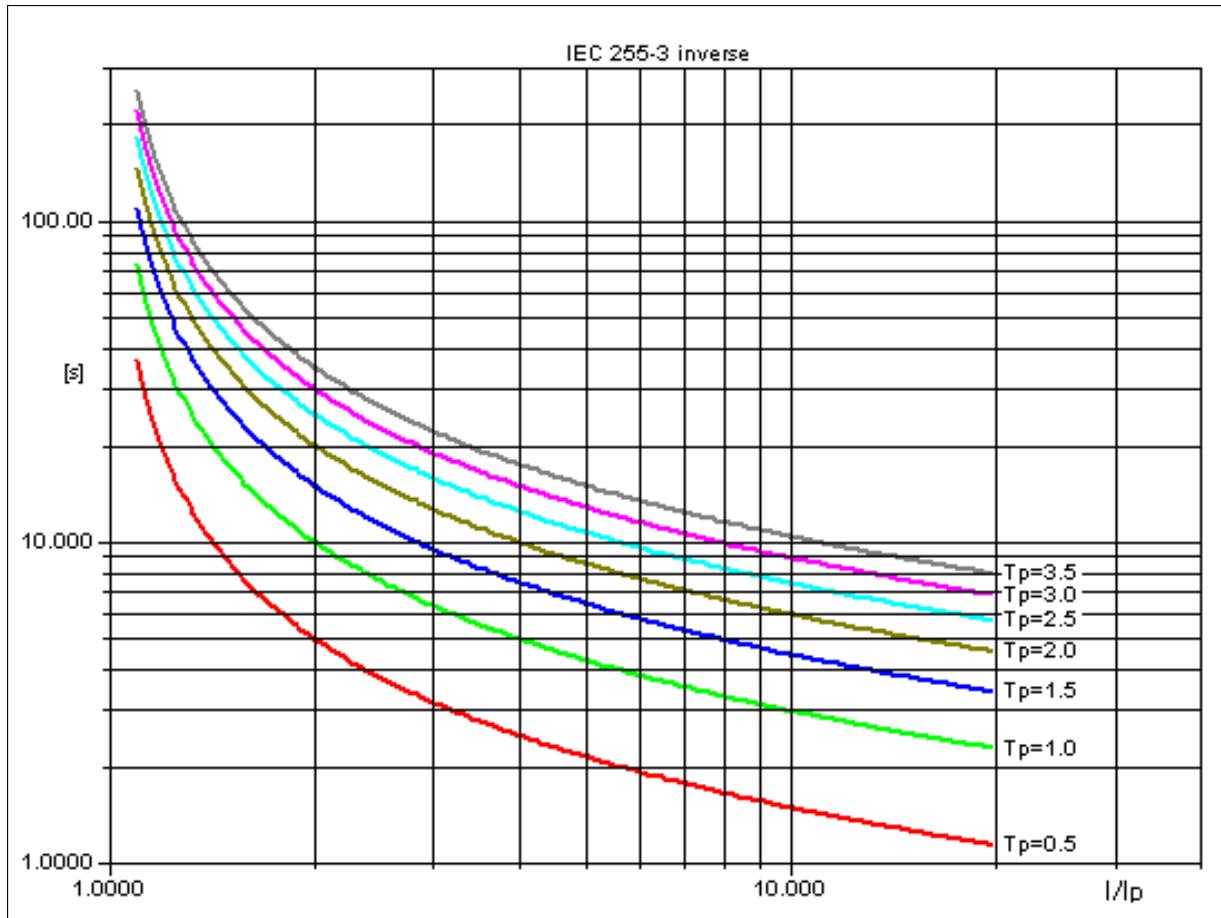


Figure 33.3.7: I-t curves for different time dials

The pickup current defines the nominal value  $I_p$  which is used to calculate the tripping time. The I-t curve definition states a minimum and a maximum per unit current. Lower currents will not trip the relay (infinite tripping time), higher currents will not decrease the tripping time any further. These limits are shown in Figure 33.3.8.

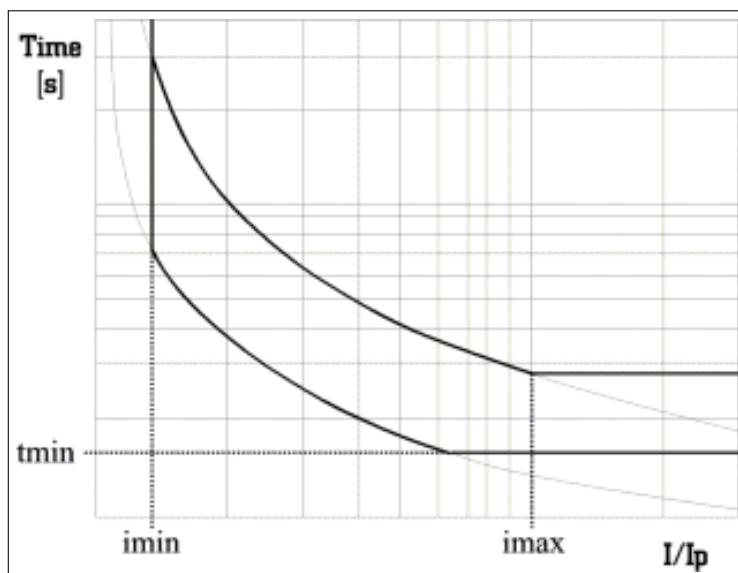


Figure 33.3.8: I-t curve limits

The pickup current may be defined by the relay type to be a per unit value, or a relay current. The nominal current defined by the measurement block (refer to Section 33.3.7.1) is used to calculate  $I_p$ . In the case of a per unit value, the relay current value already equals  $I_p$ .

Altering the pickup current will thus not change the I-t curve, but will scale the measured current to different per unit values. The following example may illustrate this:

- Suppose the minimum current defined by the I-t curve is  $i_{min}=1.1 \text{ I}/I_p$ .
- Suppose the measurement unit defines  $I_{nom}=5.0 \text{ rel.A}$ .
- Suppose pickup current  $I_{pset}=1.5 \text{ p.u.}$ 
  - relay will not trip for  $I < 1.10 \cdot 1.5 \cdot 5.0 \text{ rel.A} = 8.25 \text{ rel.A}$
- Suppose pickup current  $I_{pset}=10.0 \text{ rel.A}$ 
  - relay will not trip for  $I < 1.1 \cdot 10.0 \text{ rel.A} = 11.0 \text{ rel.A}$

### 33.3.7.5 The logic block

The logic block in *PowerFactory* is responsible for two functions in the relay. Firstly, it combines the internal trip signals from the other functional blocks, either with logical AND or OR functions and produces an overall trip status and time for the relay in a single output. Secondly, it controls one or more switches in the power system model that will be opened by the relay in the time determined by the logical combination of the various tripping signals. If the relay is located in a cubicle and no switch is explicitly specified within the logic block, the default behaviour is for the logic block to open the switch within that cubicle.

## 33.4 The time-overcurrent plot

The time-overcurrent plot (*PlotOvercurrent*) can be used for graphical analysis of an overcurrent protection scheme to show multiple relay and fuse characteristics on one diagram. Additionally, thermal damage curves for lines and transformers can be added to the plot along with motor starting curves. These plots can be used to determine relay tripping times and hence assist with protection coordination and the determination of relay settings and fuses' characteristics.

For simplified reporting of protection schemes, the time-overcurrent plot also supports visualisation of the network diagram next to the plot like that illustrated in Figure 33.4.1. This diagram also shows the relevant protection relays and instrumentation transformers with a colour scheme that matches the colour settings of the main diagram to enable easy identification of protection devices, their characteristics and their position in the network being analysed.

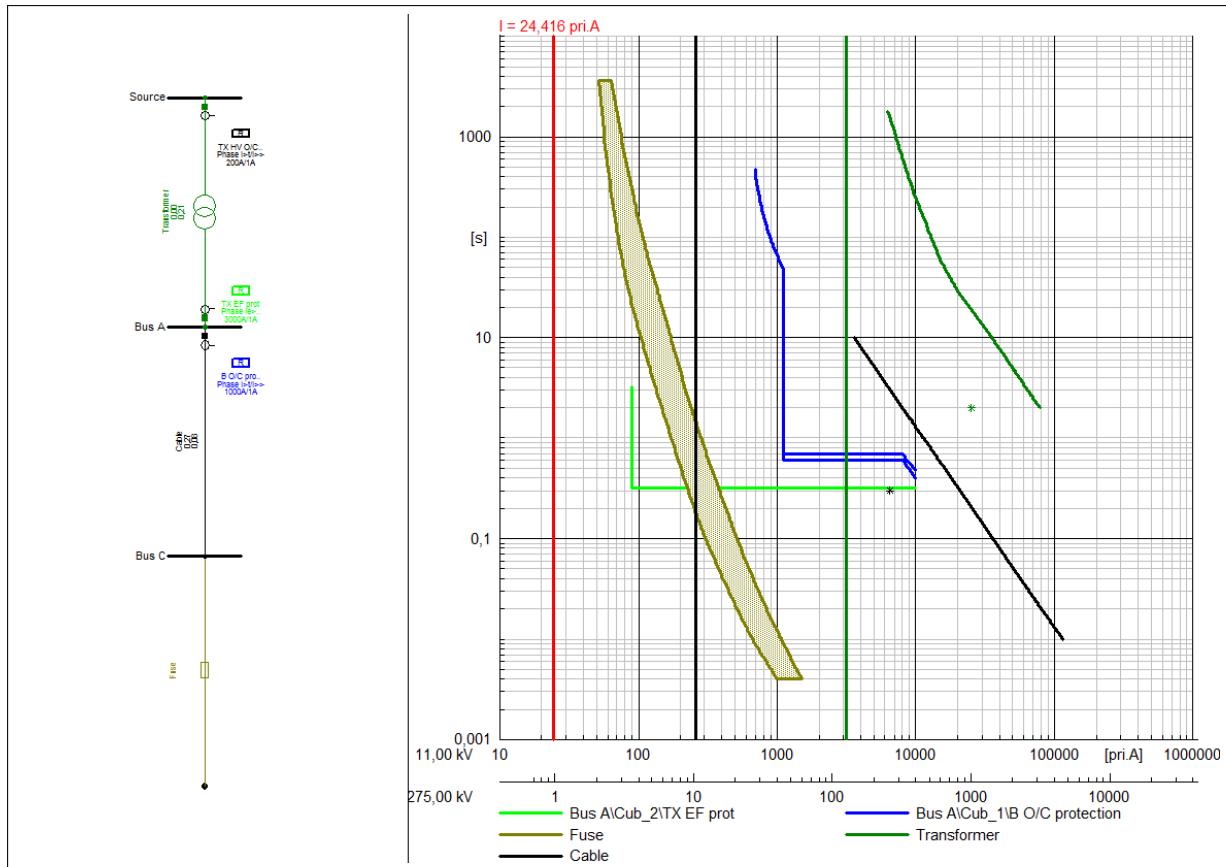


Figure 33.4.1: Time-overcurrent plot showing the auto-generated graphic for the protection path

### 33.4.1 How to create a time-overcurrent plot

There are five different methods to create a time-overcurrent plot (*VisOcplot*). You can create this plot by right-clicking the cubicle, the power system object, the protection device or the protection path. The first three methods do not show the protection single line diagram to the left of the plot, while the last two methods show it. These methods are explained in further detail in the following sections.

#### 1. From the cubicle

- Right-click a cubicle containing overcurrent relays or fuses. The context menu will appear.
- Select the option *Plots → Insert Time-Overcurrent Plot*. PowerFactory will create a diagram showing the time-overcurrent plot for all protection devices and fuses within the cubicle. See Section 33.4.7 for how to configure the presentation of the plot.

#### 2. From the power system object (line, cable, transformer)

- Select one or more objects such as transformers or lines. The context menu will appear.
- Select the option *Plots → Insert Time-Overcurrent Plot*. PowerFactory will create a diagram showing the time-overcurrent plot with the defined cable/line or transformer overload characteristic.

#### 3. From the protection device

- Open a tabular view of the protection device either from the list of calculation relevant objects (Network Model Manager) or in the Data Manager.
- Right-click the icon. A context menu will appear.
- Select *Plots → Insert Time-Overcurrent Plot*.

#### 4. From the network graphic

- (a) Select the elements from the network graphic to be included in the protection path. For more information regarding the graphical selection, please refer to the Section [15.10](#).
- (b) Right-click on any element from the selection. A context menu will appear.
- (c) Select *Plots* → *Insert Time-Overcurrent Plot*.
- (d) As a result, a path will automatically be defined for the selected graphics and an auto-generated schematic single-line diagram of the path will also be displayed to the left side of the time-overcurrent plot.

#### 5. From the protection path

- (a) Navigate to the protection path in the Data Manager.
- (b) Right-click the  icon. A context menu will appear.
- (c) Select *Plots* → *Insert Time-Overcurrent Plot*. Refer to Section [15.10](#) for more information on defining paths. In this case, an auto-generated schematic single-line diagram of the path will also be displayed to the left of the time-overcurrent plot. Additionally, multiple paths can be selected by using the left mouse button in combination with the CTRL Key.

With the last two methods, it is possible to create the time-overcurrent plots for multiple paths on separate pages or a single page. After selecting the desired paths, press the CTRL key and Select *Plots* → *Insert Time-Overcurrent Plot* simultaneously to display the plots on separate pages. If the CTRL key is not pressed then the plots will be displayed on the same page.

In methods 1-3, it is also possible to select the option *Plots* → *Add to Time-overcurrent plot* instead of *Plots* → *Insert Time-Overcurrent Plot*. This will open a list of previously defined over current plots from which any one can be selected to add the selected device to.

---

**Note:** To show the relay locations and thus to visualise cubicles containing relays, you can set the colour representation of the single-line diagram to Relay Locations. If one of these locations is then right-clicked, the option *Plots* → *Insert Time-Overcurrent Plot* is available.

---

#### 33.4.2 Understanding the time-overcurrent plot

The time-overcurrent plot shows the following characteristics:

- Time-current characteristics of relays;
- Time-current characteristics of fuses, including optionally the minimum and maximum clearing time;
- Damage curves of transformers, lines and cables;
- Motor starting curves; and
- The currents calculated by a short-circuit or load-flow analysis and the resulting tripping times of the relays.
- If defined from a path, then the simplified single line graphic showing the main power system objects, the protection devices and instrumentation transformers is displayed on the left of the diagram.

See Figure [33.4.1](#) for an example.

#### 33.4.3 Showing the calculation results on the time-overcurrent plot

The time-overcurrent plot shows the results of the short-circuit or load-flow analysis automatically as a vertical 'x-value' line through the graph. Because the current 'seen' by each device could be different

(due to parallel paths, meshed networks etc), a current line is drawn for each device that measures a unique current. If the intersection of the calculated current with the time-overcurrent characteristic causes the shown characteristic to trip, then the intersection is labelled with the tripping time. These lines automatically update when a new load-flow or short-circuit calculation is completed.

### 33.4.4 Displaying the grading margins

To show a 'grading margin' line, which shows the difference between the tripping times of each protection device:

1. After executing the calculation, right-click on the time-overcurrent plot. A context menu will appear.
2. Select the option *Show Grading Margins*. A dialog box will appear.
3. Enter the desired position of the vertical line in the 'Value' field. Note this can later be adjusted by dragging with the mouse.
4. Optional: Adjust the curve Colour, Width and Style to your preferences.
5. Optional: Choose the 'type' of the current from the radio selection control.
6. Optional: Select 'User-defined' and enter a custom label for the curve.
7. Press **OK** to show the grading margins on the plot. An example with the grading margins shown using the default blue coloured curve is shown in Figure 33.4.2

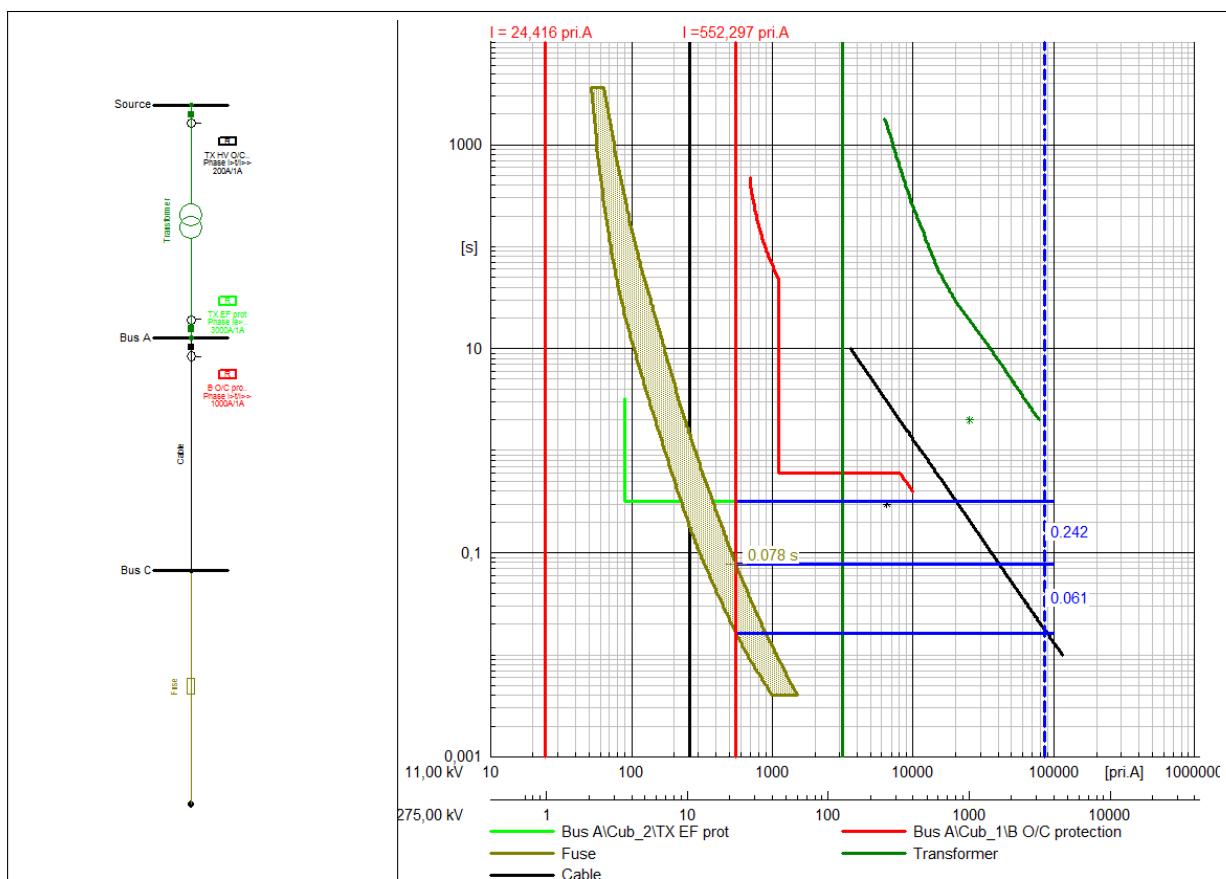


Figure 33.4.2: Time-overcurrent plot with grading margins displayed in blue

**Note:** The displayed grading margins shown by this method are the calculated grading margins based on the relay settings and the calculated current. 'Predicted' grading margins can also be shown when dragging the sub-characteristics to alter the settings. Refer to Section [33.4.11.2](#).

### 33.4.5 Adding a user defined permanent current line to the time-overcurrent plot

There are two ways to create a permanent vertical line on the time-overcurrent plot:

1. From any existing calculated short-circuit or load-flow calculated line:
  - (a) Right-click the line. A context menu will appear.
  - (b) Choose the option *Set user defined*. The line will now remain on the diagram when the calculation is reset or another calculation is completed.
  - (c) Optional: Double-click the user defined line to edit its colour, width, style and alter the displayed label. More information on the available options can be found in Section [19.8.7](#).
  - (d) Optional: It is possible to drag the line using the mouse to alter its position on the diagram.
2. A new line not based on an existing calculation:
  - (a) Right-click the time-overcurrent plot avoiding clicking on any existing curve or characteristic.
  - (b) Choose the option *Add Intersection Line → x-Constant...*. A dialog will appear that allows you to configure the properties of the line.
  - (c) Optional: Adjust the line properties such as width, colour, style and set a user defined label. More information on the available options can be found in Section [19.8.7](#).
  - (d) Optional: It is possible to drag the line using the mouse to alter its position on the diagram.
  - (e) Press **OK** to add the line to the diagram.

### 33.4.6 Configuring the single line diagram associated with time-overcurrent plots

The protection diagram that is automatically created when a time-overcurrent diagram is generated from the protection path (see option 5 in Section [33.4.1](#)) can also be manually adjusted by the user.

To edit this graphic:

1. Right-click the protection diagram within the time-overcurrent plot;
2. Select the option *Open diagram in new tab*. A single line graphic showing the diagram will appear in a new tab. The diagram can be edited like a regular *PowerFactory* single line diagram and it will automatically update in the time-overcurrent plot following any changes.

### 33.4.7 Time-overcurrent plot configuration options

To access the time-overcurrent plot settings, right-click the time-overcurrent plot and select *Edit Shown Data...* or double-click on the time-overcurrent plot.

#### 33.4.7.1 Curves

**Elements** Is a table used to specify the protection device and network elements which will be represented in the plot. An entire relay (*ElmRelay*) object potentially comprised of multiple elements each with different characteristics may be selected or alternatively individual relay block elements representing a

single characteristic may be selected. Additionally, primary network elements such as transformers, motors and cables can be selected so that their damage characteristics will be displayed in the plot. Each line in the *Curves* table is used to specify an additional relay or element. For each selected element, the way that its characteristic is displayed can be configured in the other columns of the table:

- **Visible:** Used to toggle the visibility of the curves associated with the corresponding element.
- **Element:** Specifies the element for which the characteristics should be plotted.
- **Colour/Style/Width/Fill Style:** Settings allow the user to specify the general appearance of the associated time overcurrent characteristic.
- **Split Curve:** Can be used to split protection characteristics. More information can be found in Section 33.4.11.
- **Label:** Can be used to specify custom information in the plot legend.

**Protection path.** This option is used to specify a pre-defined protection path. The **Capture Elements** button can then be used to choose elements from the selected protection path which are not listed in the *Elements* table and whose characteristics are to be plotted on the time-overcurrent plot.

**Show section of single line graphic.** When a time-overcurrent plot is generated for a particular protection path, then the single line diagram of that path is displayed together with the time-overcurrent plot. The *Show section of single line graphic* checkbox is specifically used to show or hide the single line graphic of the protection path in the time-overcurrent plot. The button **Edit** can be used to manually adjust the single line diagram of the protection path as an alternative to the method described in the Section 33.4.6.

**Plot Features.** The option *Show coordination results* can be selected to display the characteristics of protection devices after the Protection Coordination Assistant has calculated relevant settings based on a predefined set of rules (see Section 33.13). The Overcurrent Coordination Assistant tool stores the results in a result file which can be selected in the *Result File* column of the *Elements* table. The *Result File* column is only shown in the *Elements* table when the option *Show coordination results* is selected. This allows the user to compare the newly calculated protection device settings with the original ones. To display the original settings, no result file should be selected and to display the settings calculated by the Overcurrent Coordination Assistant, the corresponding result file should be selected.

The option *Select sub curves* can be selected to show the following additional columns in the *Elements* table:

- **Direction:** This option can be used to display the characteristics based on their configured tripping direction. The option *All* displays all the characteristics of a protection device irrespective of the tripping direction. However, the options *Forward* or *Reverse* can be selected to display only characteristics where the corresponding tripping direction has been selected.
- **Characteristics:** This option defines whether or not the displayed characteristic includes the additional circuit breaker delays. The default option *All* shows both the minimum clearing time characteristic (not including the breaker delay) and the total clearing time characteristic (including the breaker delay). Alternatively, it is possible to display only one of these characteristics. An example is highlighted in Figure 33.4.2. Note that the breaker delay time is specified in the basic data of the switch type *TypSwitch*.
- **Sub Curves:** For primary network elements such as transformers, cables and motors, this option can be used to include/exclude particular components of the associated characteristic such as nominal current, damage curves or inrush peak current from the time-overcurrent plot.

### 33.4.7.2 Drawing Options

The *Drawing Options* page of the time-overcurrent options dialog shows the following:

**Voltage Reference Axis.** The following options can be set:

- Current unit: The current unit may be set to either primary or secondary (relay) amperes.
- Shown voltages: More than one current axis may be shown, based on different voltage levels. *All* voltage levels found in the path when a time overcurrent plot is constructed are shown by default. A *user defined* voltage level may be added. Optionally, only the user defined voltage level is shown.

**Show relays.** This option is used to display only certain types of relay characteristics. For example, you might want to display only earth-fault relays on the diagram and ignore phase fault characteristics. This could be done by selecting the 'Earth Relays' option.

**Recloser operation.** The different recloser stages can be shown simultaneously or switched off in the diagram.

**Cut Curves at.** This option determines the maximum extent of the displayed characteristics. For the default option -, the displayed curves continue past the calculated short-circuit or load-flow current to the full extent of the defined characteristic. If the option *Tripping current* is selected, only the part of the curve less than the tripping current as determined by the active calculation is displayed. The third option, *Max. Short-Circuit/Rated Breaking Current* means the curves will be displayed to the extent of the maximum current defined within the Max/Min Fault Currents page within the protection device as described in Section 33.3.2 or alternatively to the rated breaking current of the associated circuit breaker if this is lower.

**Display results.** This option is used to select how the calculated load-flow or short-circuit currents will be displayed. Either the lines representing the calculated current values, the grading margins, both or none may be selected.

**Consider Breaker Opening Time.** This option determines whether the relay characteristics will also include the specified breaker (switch) opening time.

**Show 'out of service' units.** The characteristics for units that are out of service are invisible by default. However, a visible colour may be selected.

**Show grading margins while drag&drop.** When dragging curves, the grading margins of the curve will be shown according to the margin entered. Refer to Section 33.4.11.2 for more information on grading margins when dragging the time-overcurrent characteristics.

**Points per curve.** The number of plotted points per curve can be increased to show additional detail, or reduced to speed up the drawing of the diagram.

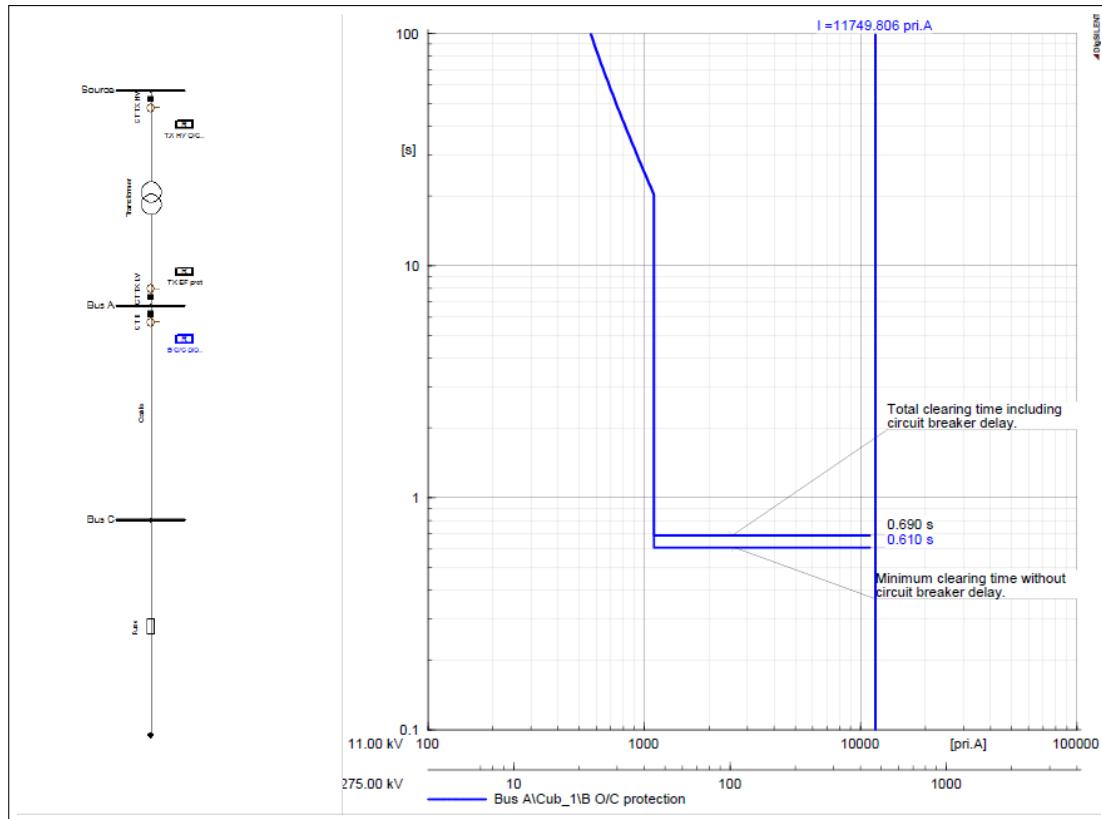


Figure 33.4.3: Time-overcurrent plot showing an overcurrent characteristic including also the breaker delay time.

### 33.4.8 Axes and Gridlines

The axes and gridlines are accessible by right-clicking on an axis and selecting *Edit Axis...*, or by double-clicking on it. These are described in Section 19.8.4.

**Note:** For time overcurrent plots the axis objects additionally include an option to consider or ignore calculated result constants when automatically scaling the axis. This can be enabled via the corresponding checkbox on the scale page of the plot axis dialog itself or via the *Consider for scaling* option which appears in the context menu when clicking in an empty part of the plot area.

### 33.4.9 Plot Legend

The legend is described in Section 19.8.5 and accessible by right-clicking on the legend and selecting *Edit Legend...*, or by double-clicking on it.

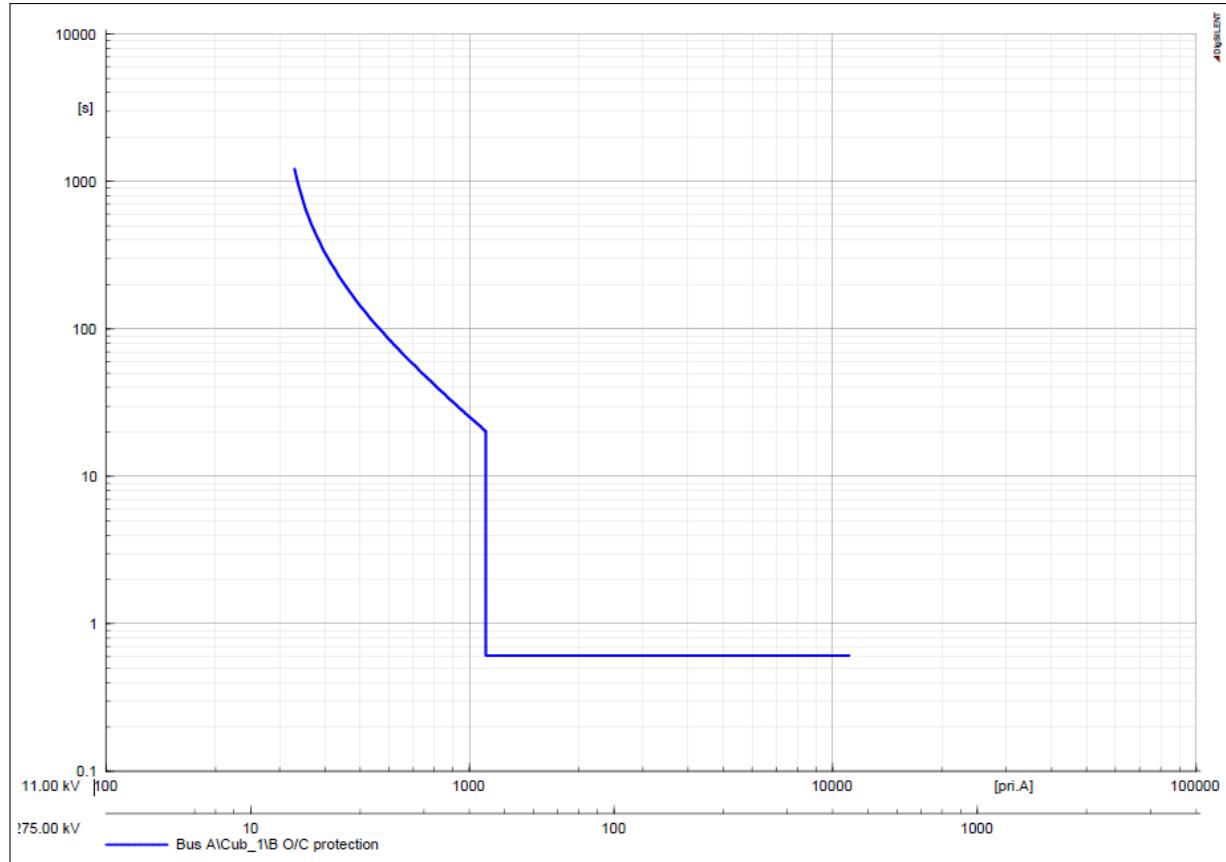
In addition, the legends in the time-overcurrent plots also support the *Display name* project setting as described in Section 9.1.3.5. For example, if the user changes this project setting and selects the *Show element name only* option, then only the element name will be displayed in the time-overcurrent plot legend.

### 33.4.10 Altering protection device characteristic settings from the time-overcurrent plot

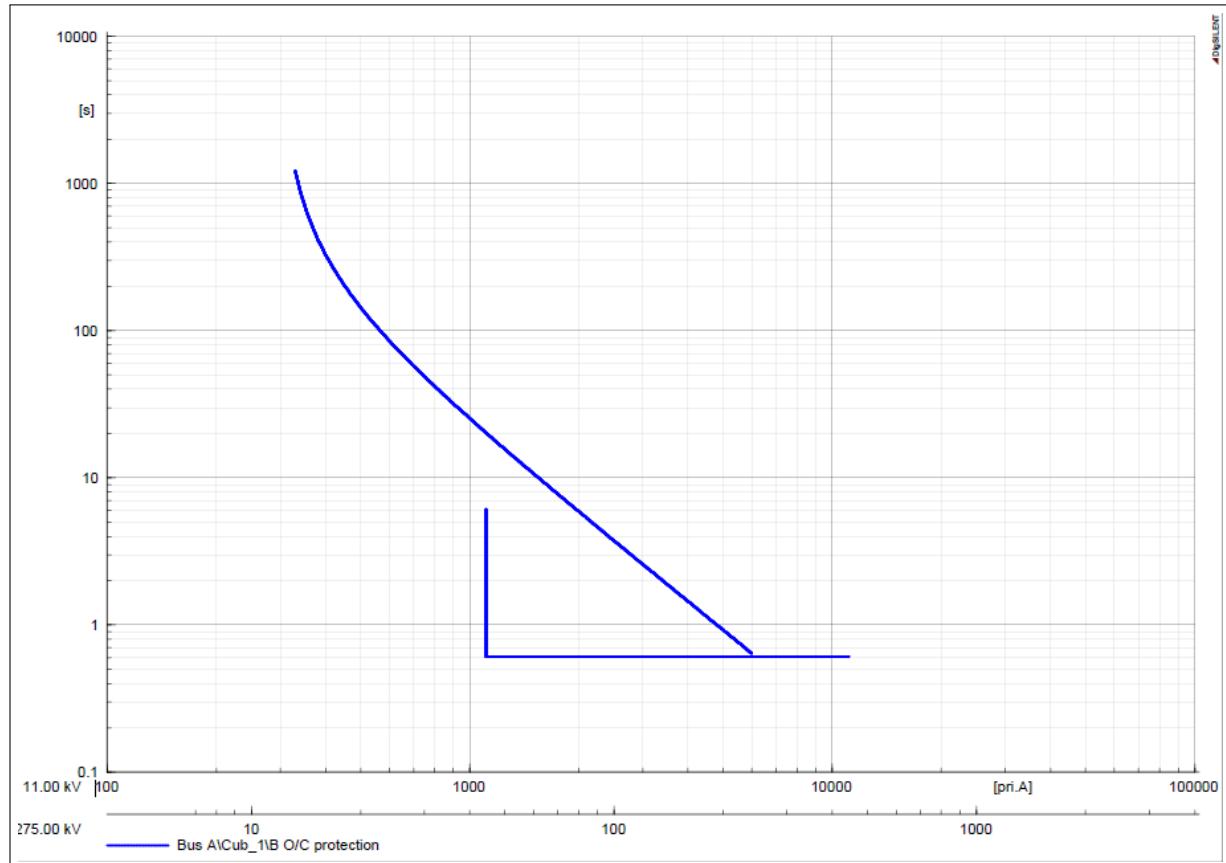
The time-overcurrent plots can be used to alter the relay characteristics graphically. This section describes various procedures used to alter such characteristics.

### 33.4.11 How to split the relay/fuse characteristic

Often a complete relay characteristic is determined from a combination of two or more sub-characteristics. For example, an overcurrent relay often has a time-overcurrent characteristic designed to operate for low fault currents and overloads and a definite time characteristic that is typically set for high fault currents. To alter relay characteristics graphically, every protection device must first be 'split' so that all characteristics are visible on the time-overcurrent plot. Figure 33.4.4 shows an example of such an overcurrent relay before it is split (left plot) and after it is split (right plot).



(a) Unsplit



(b) Split

Figure 33.4.4: Overcurrent relay characteristics in the time-overcurrent plot

There are two methods to split a relay to show the sub-characteristics:

1. Method 1:
    - (a) Right-click the characteristic. The context menu will appear.
    - (b) Select the option *Split Curve*.
  2. Method 2:
    - (a) Double-click the time-overcurrent plot avoiding any shown characteristics.
    - (b) In the table *Elements* of the displayed dialog, there is a column *Split Curve*. Check the *Split Curve* box next to the relays and fuses that need to be split.
    - (c) Click **OK** to close the dialog.
- 

**Note:** Fuses can also be split! When a fuse is split, the fuse characteristic can be dragged with the mouse to automatically change the fuse type to another fuse within the same library level.

---

#### 33.4.11.1 Altering the sub-characteristics

The first step is to *Split* the relay characteristic. See Section 33.4.11. After this there are two different methods to alter the relay sub-characteristics:

1. By left clicking and dragging the characteristic.
    - (a) Drag to the left to reduce the current setting or to the right to increase the current setting.
    - (b) Drag to the top to increase the time setting or to the bottom to decrease the time setting.
  2. By double-clicking a characteristic.
    - (a) Double click the target characteristic. A dialog for that characteristic will appear.
    - (b) Enter time and current numerical settings directly in the available fields.
    - (c) Optional: For time-overcurrent characteristics, the curve type (very inverse, standard inverse, extremely inverse) can also be selected.
- 

**Note:** Relay sub-characteristics cannot be dragged to positions outside the range defined within the relay type, nor can they be dragged diagonally to simultaneously alter the time and current setting.

---

#### 33.4.11.2 Showing grading margins during characteristic adjustment

The time-overcurrent plot option dialog (33.4.7), has an option for showing the grading margins. When this option is enabled, the grading margins will appear whenever a time-overcurrent sub-characteristic is dragged. These are represented as light blue colored characteristics above and below the main sub-characteristic. The upper limit is defined by the characteristic operating time plus the grading margin and the lower limit of the envelope is defined by the characteristic operating time minus the grading margin. An example is illustrated in Figure 33.4.5. The original characteristic is labelled as “1”, the new position as “2”, and the grading margins are labelled as “a”.

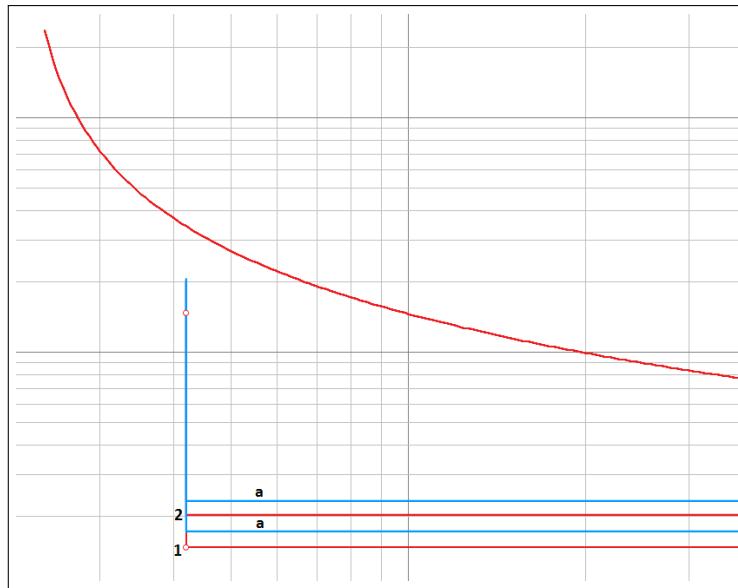


Figure 33.4.5: Grading margins when moving a characteristic

### 33.4.12 Equipment damage curves

Equipment damage curves are used to aid the positioning of relay and fuse time-current characteristics to ensure that thermal damage to equipment is minimised in the event of an overload or short-circuit.

The following types of damage curves exist:

- Conductor damage curve
- Transformer damage curve
- Motor starting curve

#### 33.4.12.1 How to add equipment damage curves to the time-overcurrent plot

There are two methods to add damage curves to an time-overcurrent plot.

1. Method 1:
  - (a) Right-click a transformer, line or asynchronous machine object. A context menu will appear.
  - (b) Select (*Plots → Insert Time-Overcurrent Plot*).
2. Method 2:
  - (a) Right-click on an existing time-overcurrent plot, in an area of the plot which does not already contain a characteristic. A context menu will appear.
  - (b) Select (*Add Curve → Transformer Damage Curve / Conductor/Cable Damage curve / Motor Starting Curve*). A dialog with options for configuring the damage curve will appear. See Sections [33.4.12.2](#), [33.4.12.3](#) and [33.4.12.4](#).

#### 33.4.12.2 Transformer damage curves

In the transformer damage curve dialog the user is able to add a damage curve in accordance with ANSI/IEEE C57.109. This standard differentiates between the damage curve of a transformer which is expected to be subjected to frequent faults and one that is subjected to infrequent faults. In the former

case, mechanical damage at high short circuit levels can be of significant concern. For category II and III transformers in particular, accounting for mechanical damage, significantly alters the damage characteristic of the transformer. An example of a time-overcurrent plot with two relay characteristics and a category II transformer damage curve for a transformer subjected to frequent faults is shown in Figure 33.4.6. The mechanical damage characteristic is ringed in the figure.

If the user wishes to define an alternative damage curve this can be achieved by selecting *User Defined curve* → *New project type*, in the dialog.

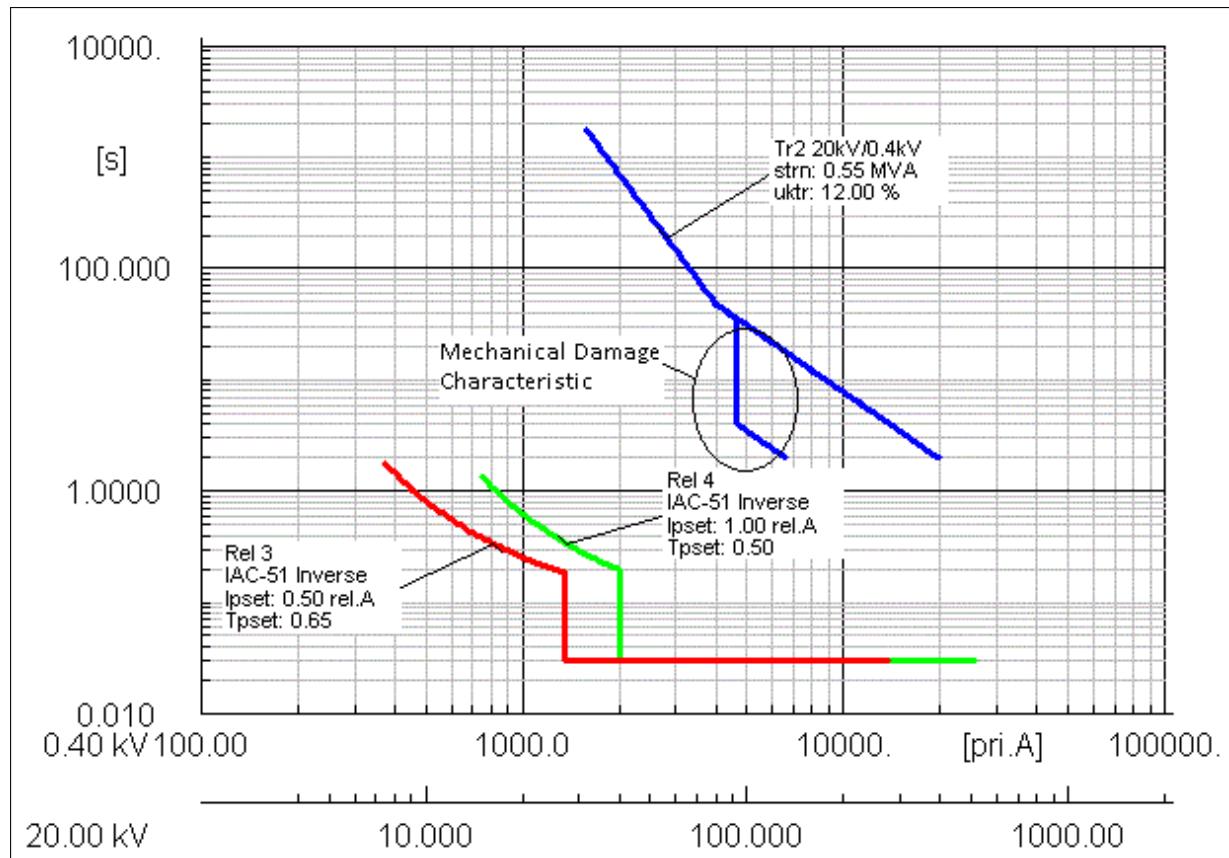


Figure 33.4.6: Transformer damage curve

The transformer damage curve consists of four parts.

#### Rated Current Curve

The rated current curve represents the nominal operation limits of the transformer. For a three phase transformer it can be calculated as:

$$I(t) = I_{rat} = \frac{S_{rat}}{\sqrt{3} \cdot U_{rat}} \quad (33.1)$$

Where:

$I_{rat}$  rated current of the transformer [A]

$S_{rat}$  rated apparent power of the transformer [kVA]

$U_{rat}$  rated voltage of the transformer [kV]

### Thermal and Mechanical Damage Curve

The thermal and mechanical damage curve represents the maximum amount of (short-circuit) current the transformer can withstand for a given amount of time without taking damage. The transformer is classified into one of four possible groups, depending on its rated apparent power and the insulation type (see Table 33.4.1). Dry-type transformers can only be category I or II.

Classification	Three-Phase	Single-Phase
Category I	$S_{rat} \leq 0.5MVA$	$S_{rat} \leq 0.5MVA$
Category II	$S_{rat} \leq 5.0MVA$	$S_{rat} \leq 1.667MVA$
Category III	$S_{rat} \leq 30.0MVA$	$S_{rat} \leq 10.0MVA$
Category IV	$S_{rat} > 30.0MVA$	$S_{rat} > 10.0MVA$

Table 33.4.1: **Categories for Transformers**

The thermal damage part of the curve is identical for all categories of the respective insulation type and is shown in Table 33.4.2. (taken from IEEE Standards Board, IEEE Guide for Liquid-Immersed Transformer Through-Fault-Current Duration, New York: The Institute of Electrical and Electronics Engineers, Inc., 1993. and IEEE Guide for Dry-Type Transformer Through-Fault Current Duration, New York: The Institute of Electrical and Electronics Engineers, Inc., 2002. )

Liquid-Immersed		Dry-Type	
$I/I_{rat}$	$t[s]$	$I/I_{rat}$	$t[s]$
25	2	25	2
11.3	10	3.5	102
6.3	30		
4.75	60		
3	300		
2	1800		

Table 33.4.2: **Thermal Withstand Capabilities**

### ANSI Mechanical Damage Curve

The mechanical part of the ANSI damage curve is only available for transformers of category II and higher. For transformers of categories II and III this part is optional and depends on expected number of fault currents flowing through the transformer over the transformers lifetime. Typically the mechanical part should be considered if the transformer is expected to carry fault current more than 10 (category II) or 5 (category III) times during its lifecycle. For category IV transformers the mechanical part of the curve is always considered. See IEEE Standards Board, IEEE Recommended Practice for Protection and Coordination of Industrial and Commercial Power Systems, New York: The Institute of Electrical and Electronic Engineers, Inc., 1999, Page 426.

The mechanical part of the damage curve is a shifted part of the thermal damage curve. The three points necessary to draw the mechanical damage curve can be calculated as follows:

$$I_1 = I_{rat} \cdot \frac{1}{u_k}; t_1 = 2,0s \quad (33.2)$$

$$I_2 = I_{rat} \cdot \frac{c_f}{u_k}; t_2 = \frac{K}{I_2^2} = \frac{I_1^2 \cdot t_1}{I_2^2} = \frac{2,0s}{c_f^2} \quad (33.3)$$

$I_3 = I_2; t_3 = \text{intersection with thermal damage curve}$

Where:

$I_{rat}$	rated current of the transformer [A]
$u_k$	short-circuit voltage of the transformer [%]
$k$	heating constant with $\frac{I}{I_{rat}} \cdot t = K = const.$
$c_f$	fault current factor [-] – $c_f = 70$ for category II and $c_f = 50$ for categories III and IV

### ANSI Curve Shift

The damage curve is based on a three phase short-circuit on the LV-side of the transformer. In case of unbalanced faults (Ph-Ph, Ph-E, Ph-Ph-E) the phase current on the HV side may be distributed over multiple phases, depending on the vector group of the transformer. The standard (IEEE Standards Board, IEEE Recommended Practice for Protection and Coordination of Industrial and Commercial Power Systems, New York: The Institute of Electrical and Electronic Engineers, Inc., 1999.) therefore suggests to multiply the rated current of the transformer by a shifting factor, thus enabling the engineer to archive proper protection of a transformer for unbalanced faults. While the shift is only applicable for “Dyn” vector-groups (according to the cited standard) and single-phase to ground faults, the same principle of current reduction on the HV side also applies to other vector groups. The resulting shifting factors and the corresponding fault type can be taken from Table 33.4.3.

Vector Group(s)	Shift Factor	Fault Type
Dd	0,87	Ph-Ph
Dyn/Dzn	0,58	Ph-E
Yyn/Zyn/Zzn	0,67	Ph-E

Table 33.4.3: **ANSI Curve Shift Factors**

### IEC Mechanical Damage Curve

The mechanical part of the IEC damage curve is only available for the element specific damage curve and consists of one point only [10]:

$$I(2,0s) = I_{rat} \cdot \frac{1}{u_k} \quad (33.4)$$

Where:

$I_{rat}$	rated current of the transformer [A]
$u_k$	short-circuit to nominal current ratio [%]

### Cold load curve

The cold load curve represents the maximum amount of current a transformer can withstand for a short-time (typically several minutes) before taking damage. The curve is specific for each transformer and the supplied loads and has to be provided by the user as a series of (I/t) pairs.

### Inrush peak current curve

The inrush curve represents the amount of current which flows into the transformer when the transformer is energised. The curve is represented by a straight line between the following two points:

$$I(T_{inrush}^{[1]}) = I_{rat} \cdot \frac{I_{inrush}^{[1]}}{I_{nom}} \quad (33.5)$$

$$I(T_{inrush}^{[2]}) = I_{rat} \cdot \frac{I_{inrush}^{[2]}}{I_{nom}} \quad (33.6)$$

Where:

$I_{rat}$	rated current of the transformer [A]
$\frac{I_{inrush}}{I_{nom}}$	inrush current to nominal current ratio [-]
$T_{inrush}$	inrush duration [s]

Note: If only one of the two points is given, only this point is drawn.

### Three Winding Transformers

The transformer damage curve can be used for 3-winding transformers. On the protection page of the element, a drop-down box is available which allows the user to select which set of values (HV-MV (default), HV-LV, MV-LV) should be used to calculate the curve. The equations remain identical, as there are normally only two windings within a coordination path.

#### 33.4.12.3 Conductor/cable damage curves

The conductor damage curve consists of four parts; a rated current curve, a short-time withstand curve, a long time overload curve and an inrush curve. These components are discussed in the following text.

##### Rated Current Curve

The rated current curve represents the nominal operation limits of the conductor.

$$I(t) = I_{rat} \quad (33.7)$$

Where:

$I_{rat}$	rated current of the line [A]
-----------	-------------------------------

##### Short-Time Withstand Curve

The short-time withstand curve represents the maximum amount of (short-circuit) current the conductor can withstand for short time periods (typically 1s) without taking damage.

There are two separate equations for this curve, both are drawn for  $0.1s \leq t \leq 10s$

Using the rated short-time withstand current:

$$I(t) = I_{thr} \cdot \sqrt{\frac{T_{thr}}{t}} \quad (33.8)$$

Where:

$I_{thr}$  rated short-time current of the line [A]

$T_{thr}$  rated short-time duration of the [s]

Using material data (only available for the generic type):

$$I(t) = \frac{F_{ac} \cdot k \cdot A}{\sqrt{t}} \quad (33.9)$$

Where:

$F_a$  lateral conductivity [-]

$A$  conductor cross-sectional area [ $\text{mm}^2/\text{kcmil}$ ]

$k$  conductor/insulation parameter [ $\frac{A\sqrt{s}}{\text{mm}^2}/\frac{A\sqrt{s}}{\text{mm}^2}\text{kcmil}$ ]

The conductor/insulation parameter can be provided by the user or calculated according to the standards equations as follows:

#### IEC/VDE equations [51]:

$$k = c_1 \cdot \sqrt{\ln\left(1 + \frac{\theta_f - \theta_i}{c_2 + \theta_i}\right)} \quad (33.10)$$

#### ANSI/IEEE equations [3]:

$$k = \sqrt{c_1 \cdot \log \frac{\theta_f + c_2}{\theta_i + c_2}} \quad (33.11)$$

Where:

$c_1$  material constant [-]

$c_2$  material constant [-]

$\theta_f$  max. short-circuit temperature [ $^\circ\text{C}$ ]

$\theta_i$  initial temperature [ $^\circ\text{C}$ ]

Note: Both equations for the conductor/insulation parameter are slightly adapted (from the original form in the standards) to fit into the same form of equation.

The values for the material constants can be taken from the table below.

Standard	IEC/VDE		ANSI/IEEE	
Conductor Material	Copper	Aluminium	Copper	Aluminium
$c_1$	226	148	0.0297	0.0125
$c_2$	234.5	228	234	228

Table 33.4.4: **Material Constants for Short-Term Withstand Calculation**

The initial temperature and final temperature  $\theta_i$  and  $\theta_f$  mainly depend upon the insulation of the conductor. The initial temperature is usually the maximum allowable continuous current temperature, whilst the final temperature is the maximum allowable short circuit temperature. Typical values for  $\theta_i$  and  $\theta_f$  are given in Table 33.4.5.

Cable insulation and type	Initial temperature (°C)	Final temperature (°C)
<b>Paper</b>		
1-6kV: belted	80	160
10-15kV: belted	65	160
10-15kV: screened	70	160
20-30kV: screened	65	160
<b>PVC: 1 and 3kV</b>		
Up to 300mm <sup>2</sup>	70	160
Over 300mm <sup>2</sup>	70	140
<b>XLPE and EPR</b>		
	90	250

Table 33.4.5: Typical cable initial temperature and final temperature values (data from the BICC Electric Cables Handbook 3rd edition)

The option *User Defined* may also be selected in the *Calculate K* field of the dialog, allowing the user to enter a value for K manually. The dialog for doing this is illustrated in Figure 33.4.7.

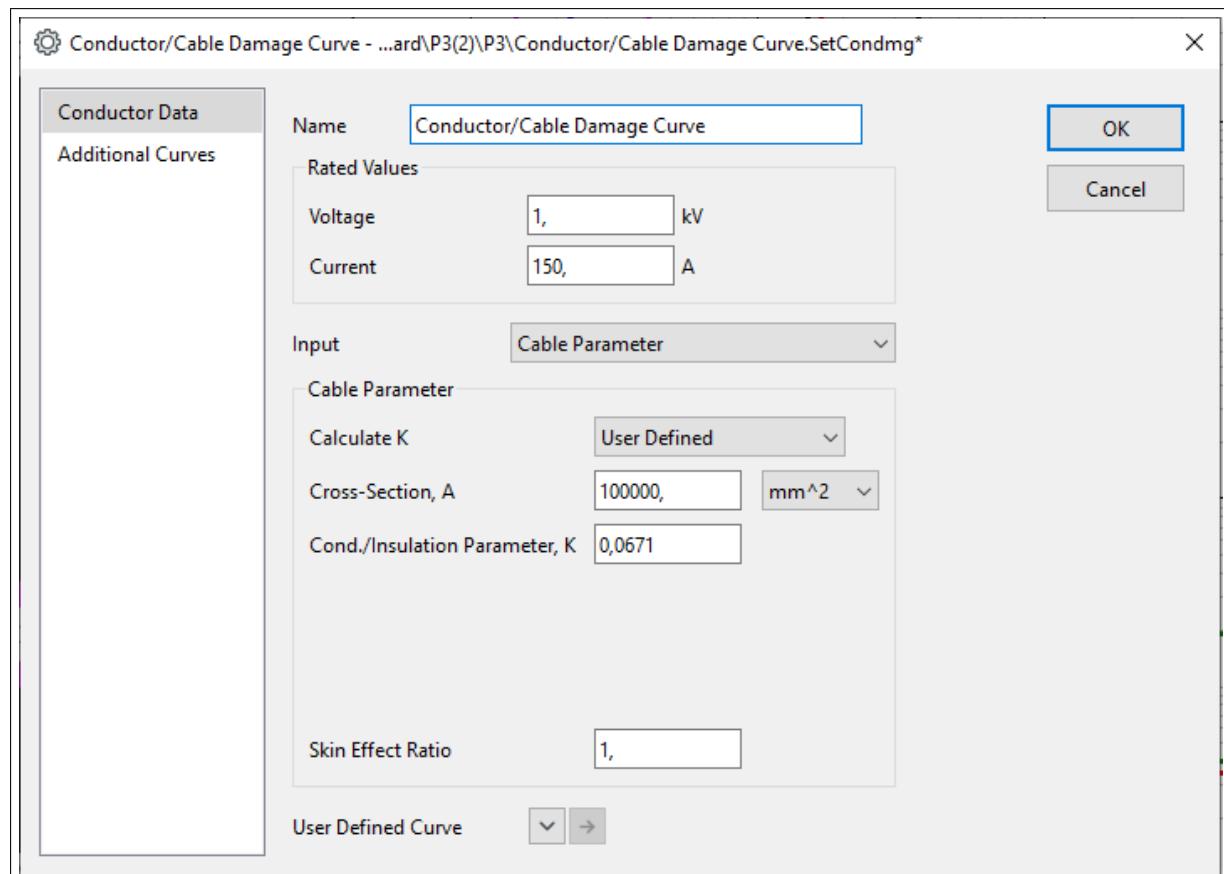


Figure 33.4.7: Conductor/Cable damage curve

Alternatively, rated short-circuit current and time may be entered if *Rated Short-Time Current* is entered as the input method.

If the user wishes to define an alternative conductor/cable damage curve this can be achieved by selecting *User Defined curve* → *New project type*.

Skin effect ratio or ac/dc ratio is a constant as defined in the NEC electrical code. The value is used when carrying out calculations to IEEE/ANSI standards and is not typically referred to by IEC/VDE standards. However, the user is given the option to specify this value when using either set of standards.

### Long time overload curve

The overload page allows the user to define the overload characteristic of the conductor. If an overload characteristic is required, it is necessary to ensure that the *Draw Overload Curve* checkbox is selected.

The user then has the option to define the overload curve according to ANSI/IEEE standards by selecting the relevant checkbox. The equation used is as follows:

$$\frac{I_E}{I_{rat}} = \sqrt{\frac{\frac{T_E - T_0}{T_N - T_0} - \left(\frac{I_0}{I_N}\right)^2 \cdot e^{-\frac{t}{k}}}{1 - e^{-\frac{t}{k}}} \cdot \frac{T_M + T_N}{T_M + T_E}} \quad (33.12)$$

Where,

$I_E$  = Max overload temperature [°C]

$I_N$  = Rated Current [A]

$I_0$  = Preload current [A]

$T_E$  = Max overload temperature [°C]

$T_N$  = Max operating temperature [°C]

$T_0$  = Ambient temperature [°C]

$T_M$  = Zero resistance temperature value [-] (234 for copper, 228 for aluminium)

$k$  = time constant of the conductor dependant on cable size and installation type [s]

Note that the value for TM is derived from the material assigned in the short circuit page which is only visible when the field calculate k is set to ANSI/IEEE or IEC/VDE.

If the checkbox is left unchecked the equation used is as follows:

$$\frac{I_E}{I_N} = \sqrt{\frac{1 - \left(\frac{I_0}{I_N}\right)^2 \cdot e^{-\frac{t}{\tau}}}{1 - e^{-\frac{t}{\tau}}}} \quad (33.13)$$

Where the variables are the same as in the previous equation. A constant designated as tau is requested in the dialog. This is identical to the constant k except k has units of hours, while tau has units of seconds.

### Inrush Curve

The inrush curve represents the amount of current that will flow into the conductor when the conductor is energised. The curve consists of one point only.

$$I(T_{inrush}) = I_{rat} \cdot \frac{I_{inrush}}{I_{nom}} \quad (33.14)$$

Where:

$I_{rat}$  rated current of the line or the damage curve input value [A]

$\frac{I_{inrush}}{I_{nom}}$  inrush current to nominal current ratio [-]

$T_{inrush}$  inrush duration [s]

#### 33.4.12.4 Motor starting curves

A motor starting curve is illustrated consists of two separate components, a starting curve and a damage curve. This section describes the equations and references underpinning the two curves.

The characteristic currents and durations given in the edit dialog result in a step wise motor start current plot, as depicted in Figure 33.4.8.

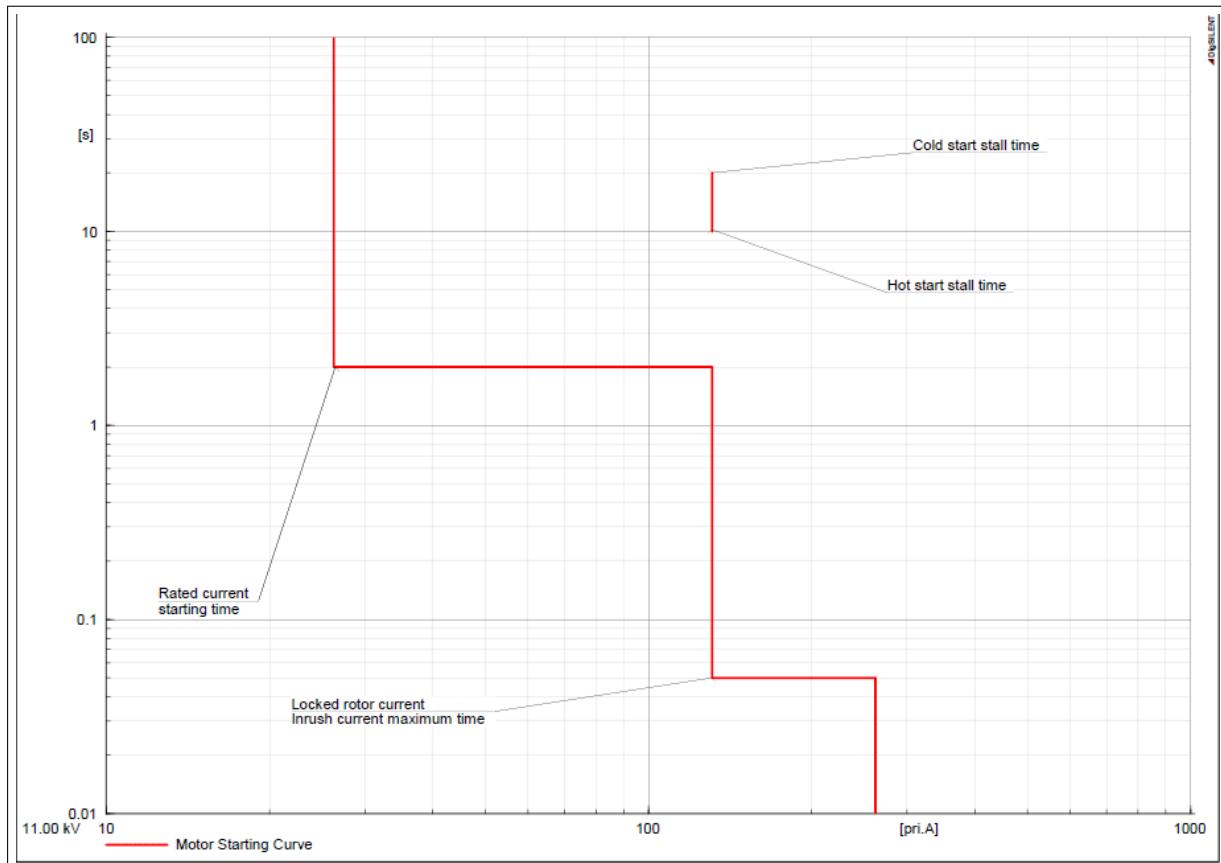


Figure 33.4.8: The motor start curve

### Motor starting curve equations

This section describes the underlying equations and references the respective standards.

---

**Note:** The equations in this section are given with respect to the rated current of the equipment. For the correct drawing in the overcurrent plot, the currents will be rated to the reference voltage of the plot.

$$I = I_{rat} \cdot \frac{U_{rat}}{U_{ref}} \quad (33.15)$$

---

The motor starting curve consists of three parts; a rated current curve, the motor starting curve and the motor inrush curve.

#### Rated Current Curve

The rated current curve represents the nominal operation limits of the motor and is drawn for  $T_{start} < t$ .

$$I(t) = I_{rat} = \frac{S_{rat}}{U_{rat}} \quad (33.16)$$

Where:

$I_{rat}$	rated current time of the motor [A]
$S_{rat}$	rated apparent power (electrical) of the motor [kVA]
$U_{rat}$	rated voltage of the motor [kV]
$T_{start}$	starting time of the motor [s]

#### Motor Starting Curve

The motor starting curve represents the maximum amount of current that will flow into the motor while it accelerates. The curve is drawn for  $T_{inrush} < t \leq T_{start}$ :

$$I(t) = I_{rat} = \frac{I_{lr}}{I_{nom}} \quad (33.17)$$

Where:

$I_{rat}$	rated current of the motor [A]
$\frac{I_{lr}}{I_{nom}}$	ratio of locked rotor current to nominal current of the motor [-]
$T_{start}$	starting time of the motor [s]
$T_{inrush}$	inrush duration [s]

### Motor Inrush Curve

The motor inrush curve represents the amount of current that will flow into the motor when it is energised. The curve is drawn from  $0,01 \text{ s} \leq t \leq T_{inrush}$ :

$$I(t) = I_{rat} = \frac{I_{inrush}}{I_{nom}} \quad (33.18)$$

Where:

$I_{rat}$	rated current of the motor [A]
$\frac{I_{inrush}}{I_{nom}}$	ratio of inrush current to nominal current of the motor [-]
$T_{inrush}$	inrush duration [s]

### Motor Damage Curve

The motor damage curve represents the maximum amount of current the motor can withstand for a given time without taking damage. There are two curves available, one representing the damage characteristic of the cold motor, one representing the damage characteristic of the hot motor. The hot curve must be lower than the cold curve. The curve would actually follow an inverse current-time characteristic but is reduced to a vertical line to indicate the damage region without cluttering the plot. The motor damage curve is drawn from  $T_{hot} \leq t \leq T_{cold}$ :

$$I(t) = I_{rat} \cdot I_{lr} \quad (33.19)$$

Where:

$I_{rat}$	rated current of the motor [A]
$I_{lr}$	ratio of locked rotor current to rated current of the motor [-]
$T_{hot}$	stall time for the hot motor [s]
$T_{cold}$	stall time for the cold motor [s]

### Synchronous Motors

The motor starting curve can be created for synchronous motors. Since synchronous motors are started in asynchronous operation, the curve is identical to the asynchronous motor starting curve. The parameter mapping for the synchronous machine is as follows:

Motor Curve	Starting	Asynchronous Motor	Synchronous Motor
Parameter		Parameter	Parameter
Rated Power	Srat	t:sgn	t:sgn
Rated Voltage	Urat	t:ugn	t:ugn
Locked Rotor Current (Ilr/In)	aiazn	t:aiazn	1 / (t:xdsss)

Table 33.4.6: Synchronous Motor Parameter Mapping

---

**Note:** By default the subtransient reactance (t:xdss) is used. If the flag “Use saturated values” in the machine type is set, the saturated subtransient reactance (t:xdsss) is used.

---

## 33.5 Basics of a distance protection scheme

Section [33.2.2](#), explains the procedure to setup a protection device in *PowerFactory*. When a new device is created within a network model there are a number of parameters to define in the dialog which appears. This section will describe the basic steps that should be completed in order to specify these parameters for distance protection relays. In many cases the setup is similar to overcurrent relay and consequently only the main differences are highlighted in this section.

The following sections, [33.6](#) and [33.8](#) will cover the main graphical tools used for distance protection analysis in *PowerFactory*.

### 33.5.1 Distance relay model setup - basic data page

The basic data page in the relay model (*ElmRelay*) dialog is where the basic configuration of the relay is completed. The procedure is the same as that used for setting up the over-current relay. Refer to Section [33.3.1](#).

### 33.5.2 Primary or secondary Ohm selection for distance relay parameters

It is always possible to enter the reach setting/s of the distance mho (refer Section [33.5.3.3](#)) and distance polygon (refer Section [33.5.3.4](#)) blocks in terms of primary Ohms or secondary Ohms. However, for the purpose of the respective block types, and specifying the valid settings range, one of these quantities must be configured as the default mode. Normally this is secondary Ohms, however some relays may allow this to be primary Ohms and hence in *PowerFactory* it is possible to alter the default option. To do this:

1. Go to the Advanced data page of the relay type.
2. Choose either Secondary Ohm or Primary Ohm.
3. Press **OK** to close the relay type.

There is another feature that is enabled if the Primary Ohm option is selected. This is the overriding of the CT and VT ratio determined from the selected VT and CT automatically with custom settings. To do this:

1. Enable the Primary Ohm option for impedance ranges as described above.
2. Select the *Current/Voltage Transformer* page of the relay element.
3. Click **Set CT/VT ratio**.
4. Enter the updated parameters of the CTs and VTs.

This feature could be used for instance to quickly see the effect of altering the CT or VT ratio without having to modify the *PowerFactory* CT and VT objects.

### 33.5.3 Basic relay blocks used for distance protection

The following sections provide a brief overview of some of the basic protection blocks that can be found within distance relays in *PowerFactory*. Some of the protection blocks such as the measurement block, logic block, directional, and overcurrent blocks that were discussed in Section 33.3.7 are also used within distance relays. Consequently, this section only discusses those blocks that are unique to distance relays. By necessity, this manual only provides a brief high level overview of the blocks. For more information on these blocks please refer to the [Protection Devices Library](#).

#### 33.5.3.1 The polarising block

The purpose of the “Polarising” block is to provide “polarising” current and voltage signals to the distance protection zones (either Mho or Polygonal). The block takes as input the following signals:

- Real and imaginary components of the three phase currents and voltages;
- Real and imaginary components of the zero sequence currents; and
- Optional: Real and imaginary components of the mutual zero sequence currents;

It produces as output:

- Real and imaginary components of the three phase-phase *operating* currents;
- Real and imaginary components of the three phase-ground *operating* currents;
- Real and imaginary components of the *polarising* phase-phase voltages;
- Real and imaginary components of the *polarising* phase-ground voltages;
- Real and imaginary components of the *operating* phase-phase voltages; and
- Real and imaginary components of the *operating* phase-ground voltages;

The calculation of the above components depends on the configuration of the block and the polarisation method selected. The currently supported polarisation methods are:

- Voltage, Self
- Voltage, Cross (Quadrature)
- Voltage, Cross (Quad L-L)
- Positive Sequence
- Self, ground compensated

Further to this, polarising blocks allow for settings of earth fault ( $k_0$ ) and mutual earth fault ( $k_{0m}$ ) compensation parameters to be applied if these features are available in the relay model.

The user can click the **Assume k0** button to automatically set the zero sequence compensation factor of the polarising block to match the calculated factor for the protected zone.

#### 33.5.3.2 The starting block

The starting block is used exclusively in distance relays as a means to detect fault conditions. It can be configured to send a starting signal to protection blocks that accept such a signal. This includes Mho, Polygonal and timer blocks. The fault detection method can be based on overcurrent or impedance. Also, both phase fault and earth fault detection is supported by the block.

### 33.5.3.3 The distance mho block

Distance protection using mho characteristics is the traditional method of impedance based protection and was initially developed in electro-mechanical relays. Today, such characteristics are also supported by numerical protection relays primarily for compatibility with these older units but also because most protection engineers are inherently familiar with mho based protection. *PowerFactory* supports the following types of mho characteristics:

- Impedance
- Impedance (digital)
- Impedance Offset
- Mho
- Mho Offset Mta
- Mho Offset X
- Mho Offset Generic
- Mho Offset 2 X
- Asea RAKZB Mho Offset

From the user perspective, the type of characteristic used by the block is dependent on the type, and the user does not normally need to be concerned with its selection from the *RelDismho* dialog, an example of which is shown in Figure 33.5.1.

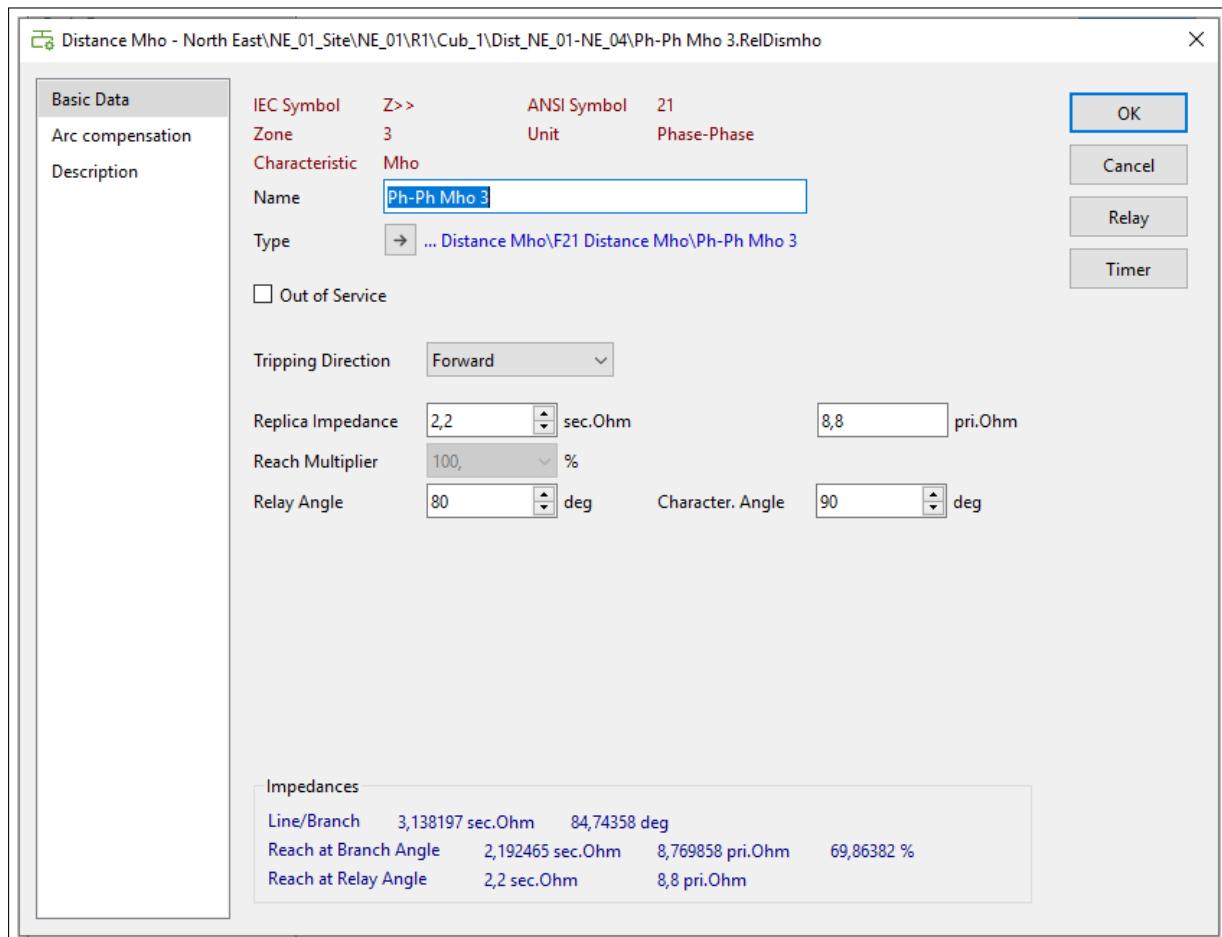


Figure 33.5.1: Distance mho block

The user is required simply to enter the settings for the replica impedance, either in secondary or primary Ohms, and the relay angle.

The block also shows the impedance characteristics of the branch that it is protecting and the effective reach of the relay in the *Impedances* section at the bottom of the dialog.

---

**Note:** The displayed impedance shown in blue text at the bottom of the mho block indicates the impedance of the primary protection zone. This could be a single *PowerFactory* line element or multiple line elements. *PowerFactory* automatically completes a topological search until it finds the next terminal with type “busbar”, or a terminal inside a substation, or another protection device. If the “protected zone” consists of multiple parallel paths, the displayed impedance is the one, out of all branches, with the largest impedance.

---

The distance mho block does not have a time dial internally, instead it is connected to an external timer block (refer Section 33.5.3.5) that controls the tripping time of the zone. However, the timer block associated with the particular mho zone can be conveniently accessed by clicking the **Timer** button.

If the **Timer** button of a zone is greyed out, this means there is no timer block directly connected to the zone. This could be the case if the zone is designed for instantaneous tripping.

#### 33.5.3.4 The distance polygon block

Most modern numerical distance protection relays tend to support a so-called polygonal (also called a quadrilateral) characteristic. The benefit of such characteristics is that they allow the definition of independent resistive and reactive reaches. In particular, the ability to specify a large resistive reach is a benefit for protecting against resistive faults.

Many modern relays also support other sophisticated features such as tilting polygons and double directional elements to constrain the impedance characteristic to a more specific area. In fact, there is not really such a thing as a standard polygonal characteristic with each manufacturer generally using a slightly different, although often similar philosophy. Consequently, the *PowerFactory* polygonal block has been designed to support a range of different characteristics including:

- Quadrilateral
- Quadrilateral Offset
- Polygonal (90°)
- Polygonal (+R, +X)
- Polygonal (Beta)
- Siemens (R, X)
- Quadrilateral (Z)
- ABB (R, X)
- ASEA RAZFE
- Quad (Beta)
- Quad Offset (Siemens 7SL32)
- EPAC Quadrilateral
- GE Quadrilateral (Z)

As for the mho block, the user does not usually need to be concerned with the selection of the correct characteristic as this is specified by the type and would have been defined by the developer of the relay model.

An example of the dialog for the polygonal (beta) characteristic in *PowerFactory* is shown in Figure 33.5.2. In this case, the block is required to set the direction, the X reach, the R resistance, the X angle, the Relay Angle and -R Ratio. Like the mho block, the timer for the zone can be easily accessed through the **Timer** button.

The *Impedance* section at the bottom of the dialog shows the reach of the zone in absolute values, as well as relative to the element directly connected to the cubicle where the relay is defined. The R and X values of this element are also shown as a reference for the setup of the zone.

---

**Note:** One major difference between a polygonal block and a mho block is that the polygonal block always requires a separate directional block. There is a convenient **Directional Unit** button that gives access to the relevant directional unit directly from the polygonal dialog.

---

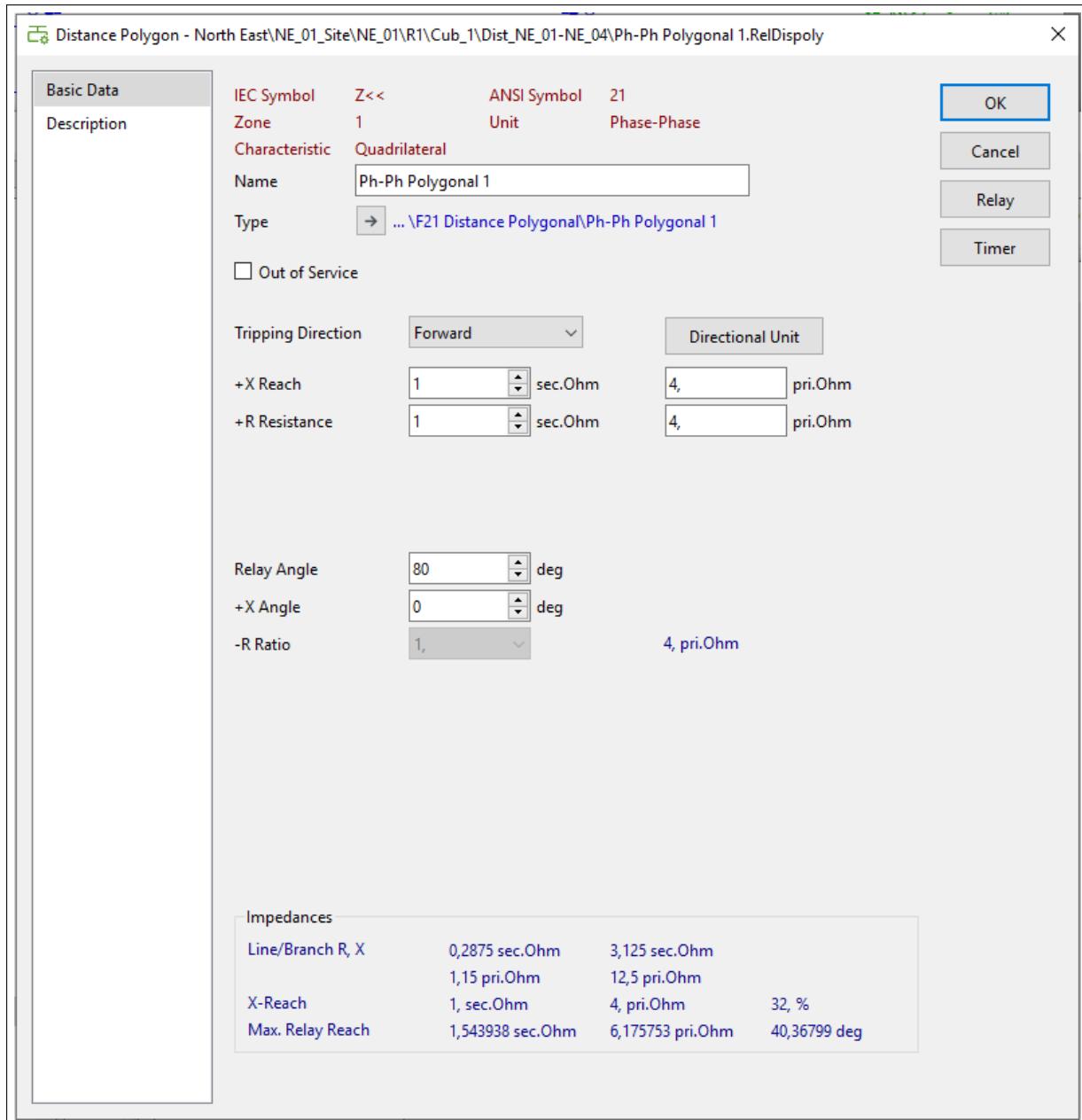


Figure 33.5.2: Distance polygon block (Polygonal (Beta))

### 33.5.3.5 The timer block

In distance relay models, the timer block is used to either control the tripping time of distance polygon blocks or to implement other time delays in the relay that cannot be implemented within a specific block.

The block has relatively simplistic functionality for steady state simulations, but can be configured also as an output hold or a reset delay in time domain simulations. The block settings can be implemented as a time delay in seconds, or as a cycle delay. If the timer block is used to control a distance polygon, the delay can be started with a signal from the starting block.

### 33.5.3.6 The load encroachment block

Many modern numerical distance protection relays include a so-called load encroachment feature. In *PowerFactory* four types of load encroachment characteristics are supported:

- Schweitzer
- Siemens
- ABB
- GE

Most types of load encroachment can be supported by using a block with one of these characteristics.

The user does normally not need to concern themselves with selecting the appropriate characteristic because this will have already been selected by the relay model developer. In this block the user is only required to set reach and angle.

### 33.5.3.7 The power swing and out of step block

In *PowerFactory* the power swing block can be configured to trigger power swing blocking of distance zones and to trip the relay when detecting out of step conditions. One or both of these functions can be enabled in this block.

A power swing blocking condition is detected by starting a timer when the impedance trajectory crosses an outer polygonal characteristic. If a declared time (usually two - three cycles) expires before the trajectory crosses a second inner polygonal characteristic zone, then a power swing is declared and the relay issues a blocking command to distance elements in the relay. The obvious potential downside to this feature is that there is the potential to block tripping of distance zones for real faults. Fortunately, the impedance trajectory for most real faults would cross the outer and inner zones of the power swing characteristic nearly instantaneously and thus the timer would not expire and the zones would remain unblocked.

The second function of the power swing block is the detection of unstable power swings and the issuing of a trip command - this is known as out of step or loss of synchronism protection. Figure 33.5.3 shows a typical power swing blocking characteristic in red, a stable power swing impedance trajectory in green and an unstable power swing trajectory in blue. The difference between these two characteristics is that the stable swing enters and exits the impedance characteristic on the same side, whereas the unstable swing exits on the opposite side. Logic can be used to detect these different conditions and thereby issue a trip when the unstable swing is detected.

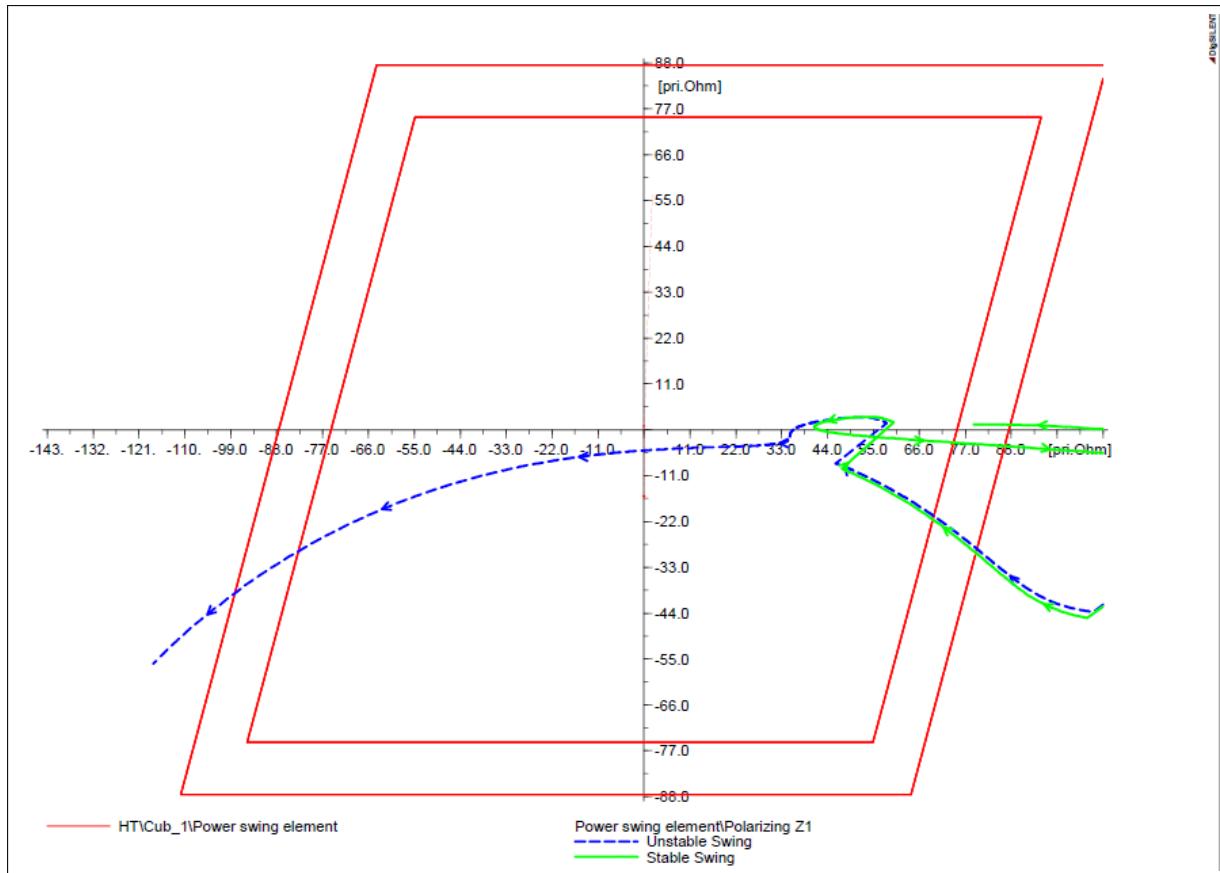


Figure 33.5.3: Stable (green) and unstable (blue) power swings

The power swing area can be configured using internal polygonal characteristics of which the ABB and Siemens types are supported. Or alternatively, it can also be configured with external impedance elements that provide inner zone and outer zone tripping signals to the power swing block.

**Note:** Out of step protection can also be configured with mho elements instead of polygonal elements.

The basic options of the power swing block are as follows:

**PS. No. of Phases.** Typically a power swing requires the impedance trajectories of all three phases to pass through the outer and inner zones to declare an out of step condition. However, in some relays this parameter is configurable.

**Blocking configuration** This parameter has three options:

- Selecting *All Zones* means that a power swing blocking signal will be sent to all distance zones.
- Selecting *Z1* means that a power swing blocking signal will be sent to only Z1 elements.
- Selecting *Z1 & Z2* will send a power swing blocking signal to Z1 and Z2 distance elements only.
- Selecting *>= Z2* will send a blocking signal to all zones except zone 1.

**Out of Step** Checking this box enables the out of step tripping function, unchecking it disables it.

**OOS No. of Crossings** This field configures how many crossings of the impedance characteristic must occur before an out of step trip is issued. For example, the blue trajectory in Figure 33.5.3 is counted as one crossing.

### 33.5.3.8 The distance directional block

The distance directional block is used by the polygonal blocks for determining the direction of the fault and also constraining the characteristic. In *PowerFactory* several types of distance directional blocks are supported:

- Earth
- Phase-Phase
- 3-Phase
- Multifunctional
- Multifunctional (digital)
- Siemens (Multi)
- ABB (Multi)

## 33.6 The impedance plot (R-X diagram)

The impedance or R-X plot shows the impedance characteristics of distance protection relays in the R-X plane. Furthermore, the plot also shows the impedance characteristic of the network near the protection relays displayed on the diagram. The plot is also “interactive” and can be used to alter the settings of the distance zones directly, thus making it a useful tool for checking or determining optimal settings for distance relays.

### 33.6.1 How to create an R-X diagram

There are three different methods to create an R-X diagram in *PowerFactory*. You can create this plot by right-clicking the cubicle, the protection device or the protection path. These methods are explained in further detail in the following sections.

#### 1. From the cubicle:

- (a) Right-click a cubicle where a distance relay is installed. A context menu will appear.
- (b) Select the option *Plots → Insert R-X Plot*. *PowerFactory* will create an R-X diagram on a new page showing the active characteristics for all relays in the selected cubicle.

#### 2. From the relay element in the Data Manager (or other tabular list):

- (a) Right-click the relay icon. A context menu will appear.
- (b) Select the option *Plots → Insert R-X Plot*. *PowerFactory* will create an R-X diagram on a new page showing the active impedance characteristics of this relay.

#### 3. From the protection path:

- (a) Right-click an element which belongs to a protection path. A context menu will appear.
- (b) Select *Network Groupings → Path → Insert R-X Plot...* from the context menu. *PowerFactory* will create an R-X diagram on a new page showing the active characteristics for all relays in the selected path.

For the first two methods, it is also possible to select the option *Plots → Add to R-X Plot* instead of *Plots → Insert R-X Plot*. This will open a list of previously defined R-X Plots. Selecting one of these plots will update the plot to additionally include the new relay device characteristic.

### 33.6.2 Understanding the R-X diagram

An example R-X diagram with two relays facing towards each other at opposite ends of a protected line is shown in Figure 33.6.1. The plot axis originates at the relaying location of the red relay. Shown on the plot are:

- The active zone impedance characteristics for each selected relay.
- The impedance characteristic of the network near to the relay location - shown as a dashed line.
- The locations of other (not selected) distance relays nearby - shown as solid coloured lines perpendicular to the network characteristic.
- The calculated impedances for each fault loop from the polarising blocks in each relay (shown as lightning bolt markers on the plot and also as values within the coloured relay result boxes).
- The detected fault type as determined by the starting elements (shown in the coloured relay result boxes).
- The tripping time of each zone (shown in the coloured relay result boxes). Note this is not enabled by default, see Section 33.7.4.1 for instructions on how to enable this.
- The overall tripping time of each relay (shown in the coloured relay result boxes).

**Note:** Hovering the mouse over the calculated impedances will generate a tooltip providing detailed information about the marker

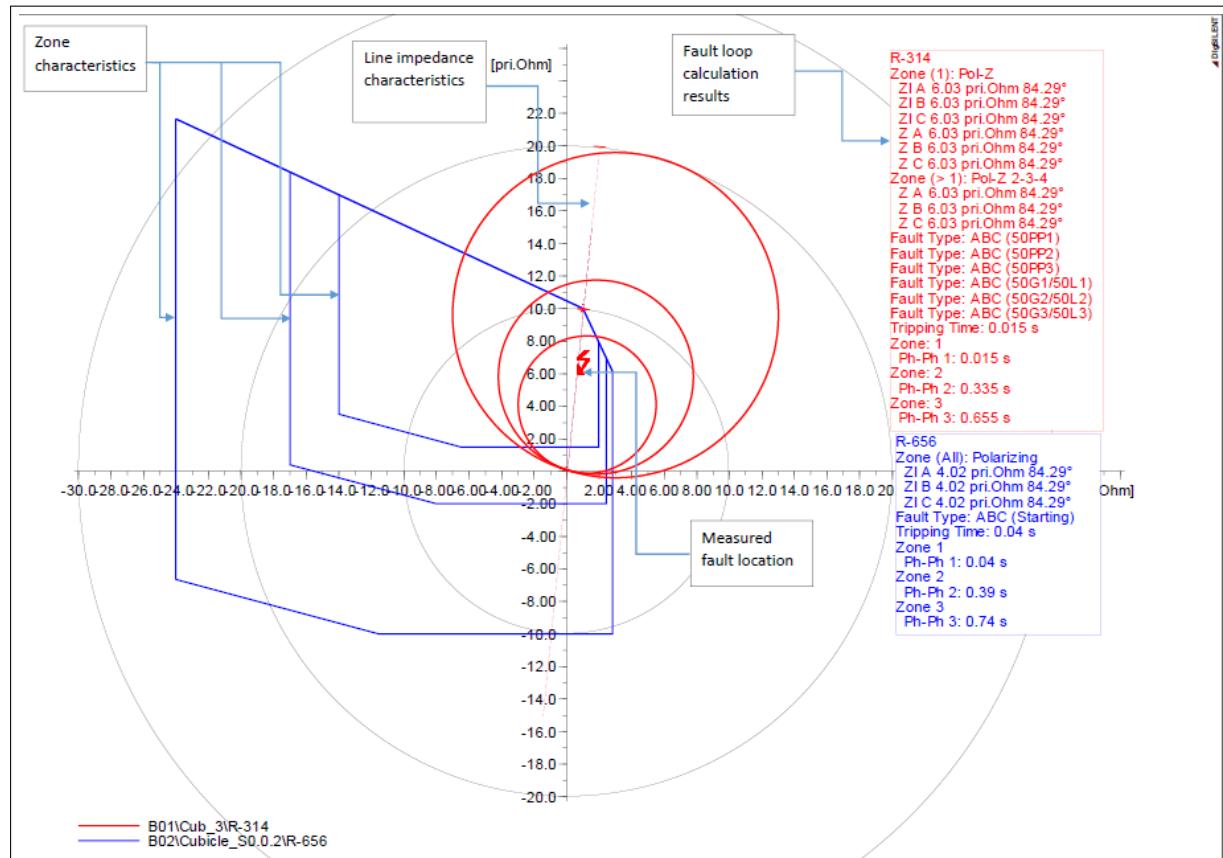


Figure 33.6.1: An R-X plot with short-circuit results and two relays

**Note:** Much of the information shown on the plot can be configured by altering the settings of the R-X plot. Refer to Section [33.6.4](#)).

---

### 33.6.3 Modifying the relay settings and branch elements from the R-X plot

From the R-X plot, the settings of the characteristics shown can be inspected and altered if required.

To do this:

1. Double click the desired characteristic. The dialog for the characteristic will appear.
2. Inspect and/or edit settings as required.
3. Click **OK** to update the characteristic displayed on the R-X diagram.

Also, it is possible to directly edit or inspect the branch elements shown on the diagram. To do so:

1. Double click the desired branch. The dialog for the branch will appear. Note that if you hover your mouse over the element and leave it steady for a few moments the name of element will appear in the balloon help.
2. Inspect or edit the branch parameters as required.
3. Click **OK** to return to the R-X diagram.

### 33.6.4 R-X plot configuration options

The R-X plot is composed of a number of subcomponents which can be independently configured and which are described in the following subsections.

#### 33.6.4.1 RX plot definition

This object dialog can be accessed by double-clicking in the main plot area in an empty region where no characteristics are displayed. Alternatively, right-clicking and choosing *Edit Shown data* will also display the RX plot definition dialog. The options available for each of the pages in the dialog are described in the following paragraphs.

##### Relays page

This page is used to define the relays which will be displayed in the plot.

- **Protection path.** A predefined *Path* object (see section [15.10](#)) may be selected here. Once the path has been assigned the *Capture elements* button may be pressed to conveniently select relays present in the coordination path for addition to the plot.
- **Relays.** The relays which are to be displayed in the plot are listed in this table. The table can be extended or shortened by right clicking in the table and selecting the relevant option. If a new row is appended a relay object should be selected or pasted into the *Element* column of the new row. The visibility, colour, line style, and line widths that are used to represent the relays on the plot are also configured in the columns of this table. Each of these can be adjusted by double-clicking and selecting an alternate option. Refer to Section [19.8](#) for more information on configuring plots in *PowerFactory*.

### Trajectories page

The trajectories page is used to overlay the plot with results. The results would typically be contained in either a result file or a COMTRADE file represented in *PowerFactory* via a *COMTRADE File Info object* containing a link to the relevant COMTRADE file. A result file is generally selected if the results were obtained from a dynamic simulation (or populated by a script). Typically, the result file would contain resistance and reactance values calculated by distance relay polarising blocks during the course of a previously executed dynamic simulation. COMTRADE files would typically be used where network measurements obtained from real network equipment such as relays or fault recorders are to be replayed onto the R-X plot.

An example where this functionality can be useful is when displaying and analysing Power Swing events. When overlaid on distance relay characteristics used to implement power swing blocking or out of step detection the performance of the relays can be analysed too. The configurable parameters of the page are described below:

- **Trajectories.** Each row of this table is used to specify a different trajectory to be overlaid on the plot. The table can be extended or shortened by right clicking in the table and choosing to insert, append or delete rows as required. The trajectory will only be visible if the *Result File* column references an object (ElmRes or IntComtrade) populated with the relevant results and when the corresponding checkbox is selected in the *Visible* column. If an ElmRes object is selected the *Element* column is used to specify the *PowerFactory* object for which results are to be displayed. Typically a polarising block would be specified here. If a COMTRADE file is being replayed using an IntComtrade object then the *Element* column is also populated with a reference to the IntComtrade object. the *x variable* and *y variable* columns are used to specify the respective resistance and reactance results contained within the selected result file. The colour, line style, line width, sample markers and labels that are used to represent the trajectories on the plot are configured in the remaining columns of this table. Each of these can be adjusted by double-clicking and selecting a suitable option. Refer to Section 19.8 for more information on configuring plots in *PowerFactory*. The *Sample Marker* column can be specified with a relevant marker shape in order to highlight on the trajectory the precise sampled values from the result file which are displayed on the plot. This can be helpful in order to provide a sense of how quickly the impedance trajectory is changing.
- **Restrict time range.** A result file may contain information covering a wider time frame than required. When selected, this checkbox can be used to restrict the displayed results to the specific time range which is relevant. Any information that precedes the specified *start time* or exceeds the *End time* is not displayed on the plot. The values can also be manually iterated by the user to help track the progress of a trajectory with time.
- **Draw direction arrows.** To help the user understand the direction of the trajectory this option can be selected in order to add direction arrows to the displayed trajectory. The arrows indicate the direction in which result samples associated with later time steps can be found.

### Drawing Options page

This page is used to control the appearance of a variety of characteristics which are overlaid on the R-X plot, including the relay zone characteristics themselves and the impedances of the network branches (e.g. lines and cables) that they protect.

The *General* tab of this page has the following settings:

- **Unit.** This option changes the reference frame for the plot axes and therefore determines whether the characteristics are displayed in primary or secondary ohms. It is also possible to select *% of line* which will adapt the axes to display all characteristics in terms of a % impedance of the relay's primary protected branch.
- **Relays Units.** This option is used to include or exclude specific types of relay characteristic from display. For example, it is possible to display only earth fault distance characteristics by selecting the option *Ph-E*.

- **Restrict zones.** This setting determines which zones are displayed. The text field can be used to restrict the zone to a specific range e.g. by specifying 1-3 only zones 1,2 and 3 will be displayed. Alternatively, by separating zones using a comma, specific zone combinations can be displayed. e.g. by specifying 1,3 in the text field only zones 1 and 3 will be displayed.
- **Starting.** This checkbox configures whether impedance starting element characteristics will be displayed on the diagram.
- **Overreach zones.** This checkbox configures whether overreach elements will be displayed on the diagram.
- **Power Swing.** This checkbox configures whether power swing elements will be displayed on the diagram.
- **Load Encroachment.** This checkbox configures whether load encroachment elements will be displayed on the diagram.
- **Complete shape.** This checkbox enables the display of the complete polygonal characteristic, when part of it would normally be invisible (and not a valid pickup region) due to the effect of the distance directional element. Enabling it also allows the selection of a line style for the displayed part of the characteristic that is not normally visible.
- **Marker.** This option is used to select how the calculated load-flow or short-circuit current fault loop markers will be displayed in the plot. A variety of options are available including an option to hide the markers completely (do not show).
- **Impedance limit.** This option can be used to exclude high impedance fault loop markers from display within the plot. The following settings are available, all, only smallest,  $Z < 1.5 * Z_{min}$  and  $Z < 5 * Z_{min}$ . If *only smallest* is selected then only the smallest magnitude line to line and line to ground fault loop impedances will be displayed. For the other options  $Z_{min}$  corresponds with the aforementioned smallest magnitude fault loop impedances.
- **Show tolerance.** When this option is selected a percentage tolerance may be specified. The tolerance is displayed as shading around the fault loop markers in the plot in cases where there may be benefit in illustrating potential uncertainty around the calculated values.
- **Calculated Impedances.** Determines whether the impedances calculated for each fault loop by the relay's polarising block(s) will be displayed in the *relay results* box.
- **Detected Fault Type.** Determines whether the fault type calculated by the starting element will be displayed in the relay results box.
- **Tripping Time of Relay.** Determines whether the overall tripping time of the relay will be displayed in the relay results box.
- **Tripped Zones.** Determines whether the tripping time of each zone that trips will be independently displayed in the relay results box.
- **Force Positive Sequence Representation.** For some relay models the reach of phase to earth zones is defined in terms of loop impedance. This is in contrast to the majority of relays where the reach is defined in terms of positive sequence impedance. This makes it difficult to compare the respective reach of relays employing the different zone definition methodologies when presented side by side. By selecting this check box, the visualisation of zone reaches which are defined in terms of ohms per loop are adjusted for projection onto an R-X plot that shows all measured impedances and zone reaches in the positive sequence representation only. This can make it easier to coordinate between relays of different manufacturers. Note this feature is not available for every relay model where reach is defined in terms of loop impedance.
- **Default Length for Blinder Units.** This option is used to adjust the represented length of blinder units on the plot and is specified in secondary ohms.
- **Colour out of service units.** By default out of service characteristics are invisible. However, Out of service characteristics can be shown in a different colour making them visible on the plot.

The *Branch impedances* tab of this page is used to specify how the branch impedance elements are displayed on the diagram and has the following settings:

- **Number of Relay Locations.** A topological search can be extended out into the network from the location of each of the relays assigned to the plot with the object of finding further relay locations. Once the first level of relay locations is found, the search can be extended to find the next level and so on. These relay locations are connected to the original relay location and to each other by branch elements such as lines. These branch elements will become less and less relevant the further the search extends from the original relay locations. This setting allows to constrain the topological search to a specific number of levels, based on the hierarchical position of the remote relay locations relative to the position of the relays selected in the plot. Only branches extending up to the specified relay location level number are displayed. If zero is specified, no branches are shown at all.
- **Branches, max. Depth.** The topological search described above is stopped if no new relay locations have been found since the last relay location was encountered and the search has traversed the number of branch elements specified for this setting. Only traversed branches are displayed in the plot. If set to zero, no branches are shown at all.
- **Ignore Transformers.** The topological search described above will not extend in to transformer elements when this option is selected and therefore the impedance of transformers will not be displayed in the plot.
- **Impedances.** There are two methods for determining the branch impedance. The first, *Input Data*, uses the entered impedance data of the branches specified in their respective types. The second method, *Calculated Impedance*, carries out a short circuit sweep calculation similar to that described in Section 33.8.3 except that the measured impedances are calculated rather than tripping times. One scenario where this method is more accurate is when modelling the protection of a section of network with multiple infeeds. Greater accuracy is achieved at the expense of calculation time.
- **Line style.** The line style for branch impedance representation can be selected.
- **Line width.** The line width for branch impedance representation can be selected.

### Text Format page

This page is used to configure the text formatting for the plot. The options are identical to those used for the time overcurrent plot as described in Section 33.7.4.1

### Style and Layout page

On the *Style and Layout* page, the additional display options can be selected as described in Section 19.8.2.3.

#### 33.6.4.2 Axes and Gridlines

The axes and gridlines are accessible by right-clicking on an axis and selecting *Edit Axis...*, or by double-clicking on it. Many of the available configuration parameters are described in Section 19.8.4.

---

**Note:** for R-X plots specifically it is possible to specify whether a polar or cartesian coordinate representation is used using the setting on the *Scale* page of this object. Enabling and disabling of autoscaling, centering of the origin and scaling of the plot to either characteristics, calculated results or both can also be done from the context sensitive menu by right-clicking in the main plot area in an empty region where no characteristics are displayed and by selecting the corresponding options.

---

### 33.6.4.3 Plot Legend

The plot legend configuration is described in Section 19.8.5 and is accessible by right-clicking on the legend and selecting Edit Legend..., or by double-clicking on it.

In addition, the legends in the R-X plots also support the *Display name* project setting as described in Section 9.1.3.5. For example, if the user changes this project setting and selects the *Show element name only* option, then only the element name will be displayed in the R-X plot legend.

### 33.6.5 Adding user defined constants to the R-X plot

User defined constants can be added to the R-X plot by doing the following

1. Right-click on the R-X plot avoiding clicking on any existing characteristic.
2. Choose the option *Add Constant* → *then select the required type of constant...*. A dialog will appear that allows you to configure the properties of the constant.
3. Optional: Adjust the constant properties of the constant and set a user defined label. More information on the properties can be found in Section 19.8.7.
4. Press **OK** to add the constant to the diagram.

### 33.6.6 Converting fault loop impedance markers generated from a calculation into permanently displayed impedance markers

When a calculation is active, fault loop impedance markers are automatically displayed in the R-X plot as described in Section 33.6.2. When the calculation is reset these markers will usually disappear. However, in some cases it may be helpful to continue to display the markers. For example, for comparison purposes once a different calculation is executed. To facilitate this comparison, it is possible to convert any fault loop impedance marker generated from an active calculation into a permanently displayed impedance marker, by doing one of the following:

1. By right clicking:
  - (a) With a calculation active, Right-click on the impedance marker which is to be converted.
  - (b) Choose the option *Set User Defined* from the context sensitive menu.
2. By double clicking:
  - (a) With a calculation active, double-click on the impedance marker which is to be converted.
  - (b) Press the *Set User Defined* button in the dialog which appears.

Once the action above has been completed for each relevant impedance marker, reset the calculation. The converted marker(s) should remain active, with a label attached.

Once the converted marker has been generated its properties and the properties/visibility of its label can be modified by doing one of the following:

1. By right clicking:
  - (a) Right-click on the converted impedance marker or its label.
  - (b) Choose the option *Edit Curve Label* from the context sensitive menu.
  - (c) Configure the properties of the dialog which appears as required.
2. By double clicking:
  - (a) Double-click on the converted impedance marker or its label.
  - (b) Configure the properties of the dialog which appears as required.

More information on configuring the properties of the marker object (VisValue) can be found by visiting the manual section linked [here](#).

## 33.7 The relay operational limits plot (P-Q diagram)

The relay operational limits or P-Q diagram plot shows the starting characteristics of distance protection relays in the P-Q plane. Typically a network operator will deal with power quantities rather than impedance, current or angle quantities when characterising the power flows and load in their network. By representing the starting characteristic in the P-Q plane the network operator is better able to compare their actual load data with the starting settings in the network's distance relays so as to assess whether particular load configurations might cause undesirable operation of network protection relays. A load configuration resulting in a power flow which lies outside of the starting characteristic of a distance relay in the P-Q plane will cause the relay to start and may therefore result in tripping of the relay. The plot is intended to assist in avoiding such circumstances.

The plot is “interactive” and can be used to alter the settings of the relay starting element directly, thus making it a useful tool for checking or determining optimal starting settings for distance relays.

### 33.7.1 How to create a P-Q diagram

#### 1. Using the *Insert Plot icon*:

- (a) This general method for creating any plot is described in Section [19.8](#).
- (b) The relay for which the starting is to be plotted should be entered as the *Element* in the *Relays* table of the plot dialog.

### 33.7.2 Understanding the P-Q diagram

Two example P-Q diagram plots for two separate relays are shown in Figure [33.7.1](#). The plots show the following:

- In both diagrams the horizontal axis represents active power whilst the vertical axis represents reactive power.
- The diagram on the left relates to a distance relay utilising underimpedance starting.
- The diagram on the right relates to a distance relay utilising overcurrent starting.
- The regions bounded by the two characteristics represent operating points which will not cause the starting to pick up. Operating points outside of these bounded regions will cause the starting elements to pick up.
- Note that the operating characteristics displayed are scaled according to a voltage factor selected in the plot settings. In the illustrated cases the voltage factor is selected to be 0.9 p.u. for both locations.
- The lightning bolt markers indicate the operating points of the system as the relaying locations as determined by the load flow calculation. Note that the load flow results are scaled according to the voltage factor applied to the plot.
- In the legend, the operating point of the system as observed by the relays is quantified. Both the voltage factor scaled and measured operating points are quantified.
- Note that for the left relay the operating point is quantified on a polar coordinate system, whilst for the right relay the operating point is quantified on a Cartesian coordinate system. This is due to a user selectable parameter in the plot settings.

---

**Note:** Hovering the mouse over the calculated P-Q marker will generate a tooltip providing more detailed information about the marker

---

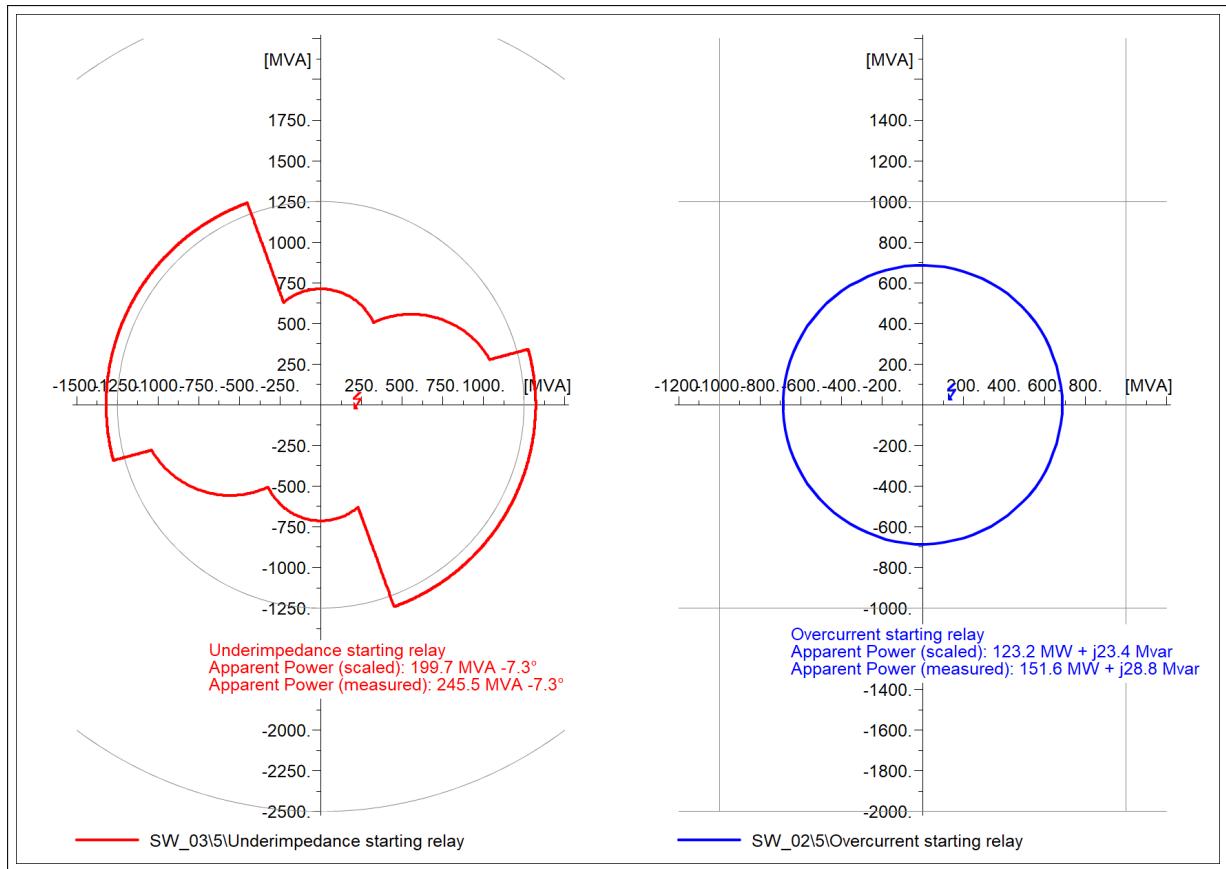


Figure 33.7.1: A P-Q diagram plot with load flow results and starting characteristics for two separate relays

### 33.7.3 Modifying the starting element settings from the P-Q plot

From the P-Q plot, the settings of the starting element shown can be inspected and altered if required.

To do this:

1. Double click the desired characteristic. The dialog for the characteristic will appear.
2. Inspect and/or edit settings as required.
3. Click **OK** to update the characteristic displayed on the P-Q diagram.

### 33.7.4 Configuring the P-Q plot

There are several ways to alter the appearance of the P-Q diagram. Many configuration parameters can be adjusted by right-clicking in the main plot area in an empty region where no characteristics are displayed and by then selecting the relevant options from the context sensitive menu. Alternatively, double-clicking in certain areas of the plot will open the dialogs of components associated with the plot. The P-Q plot is composed of a number of subcomponents which can be independently configured and which are described in the following subsections.

### 33.7.4.1 P-Q plot definition

This object dialog can be accessed by double-clicking in the main plot area in an empty region where no characteristics are displayed. Alternatively, right-clicking and choosing *Edit Shown data* will also display the P-Q plot definition dialog. The options available for each of the pages in the dialog are described in the following paragraphs.

#### Relays page

This page is used to define the relays which will be displayed in the plot.

- **Relays.** The relays which are to be displayed in the plot are listed in this table. The table can be extended or shortened by right clicking in the table and selecting the relevant option. If a new row is appended a relay object should be selected or pasted into the *Element* column of the new row. The visibility, colour, line style, and line widths that are used to represent the relays on the plot are also configured in the columns of this table. Each of these can be adjusted by double-clicking and selecting an alternate option. Refer to Section 19.8 for more information on configuring plots in *PowerFactory*.

#### Drawing Options page

This page is used to control the appearance of the starting characteristics which are overlaid on the P-Q plot and the calculated PQ values.

- **Voltage factor.** This option controls the voltage factor used in determining the scaled operating characteristics and the scaled apparent power measurement displayed in the relay results box as also described and illustrated in Section 33.7.2
- **Marker.** This option is used to select how the calculated load-flow or short-circuit current fault loop markers will be displayed in the plot. A variety of options are available including an option to hide the markers completely (do not show).
- **Show tolerance.** When this option is selected a percentage tolerance may be specified. The tolerance is displayed as shading around the PQ measurement markers in the plot in cases where there may be benefit in illustrating potential uncertainty around the calculated values.
- **Show in relay results box.** Determines whether the Calculated PQ values for the active calculation be displayed in the relay results box.

#### Text Format page

This page is used to configure the text formatting for the plot. The options are identical to those used for the time overcurrent plot as described in Section 33.7.4.1

#### Style and Layout page

On the *Style and Layout* page, the additional display options can be selected as described in Section 19.8.2.3.

### 33.7.4.2 Axes and Gridlines

The axes and gridlines are accessible by right-clicking on an axis and selecting *Edit Axis...*, or by double-clicking on it. Many of the available configuration parameters are described in Section 19.8.4. Additionally, for P-Q plots specifically it is possible to specify whether a polar or cartesian coordinate representation is used using the setting on the *Scale* page of this object. Enabling and disabling of autoscaling, Centring of the origin and scaling of the plot to either characteristics, calculated results or both can also be done from the context sensitive menu by right-clicking in the main plot area in an empty region where no characteristics are displayed and by selecting the corresponding options.

### 33.7.4.3 Plot Legend

The plot legend configuration is described in Section 19.8.5 and is accessible by right-clicking on the legend and selecting Edit Legend..., or by double-clicking on it.

In addition, the legends in the R-X plots also support the *Display name* project setting as described in Section 9.1.3.5. For example, if the user changes this project setting and selects the *Show element name only* option, then only the element name will be displayed in the R-X plot legend.

### 33.7.5 Adding user defined constants to the P-Q plot

User defined constants can be added to the P-Q plot by doing the following

1. Right-click on the P-Q plot avoiding clicking on any existing characteristic.
2. Choose the option *Add Constant* → *then select the required type of constant...*. A dialog will appear that allows you to configure the properties of the constant.
3. Optional: Adjust the constant properties of the constant and set a user defined label. More information on the properties can be found in Section 19.8.7.
4. Press **OK** to add the constant to the diagram.

### 33.7.6 Converting P-Q markers generated from a load flow calculation into permanently displayed P-Q markers

When a load flow calculation is active, P-Q markers are automatically displayed in the P-Q plot as described in Section 33.7.2. When the calculation is reset these markers will usually disappear. However, in some cases it may be helpful to continue to display the markers. For example, for comparison purposes once a different calculation is executed. To facilitate this comparison, it is possible to convert any P-Q marker generated from an active calculation into a permanently displayed P-Q marker, by doing one of the following:

1. By right clicking:
  - (a) With a calculation active, Right-click on the P-Q marker which is to be converted.
  - (b) Choose the option *Set User Defined* from the context sensitive menu.
2. By double clicking:
  - (a) With a calculation active, double-click on the P-Q marker which is to be converted.
  - (b) Press the *Set User Defined* button in the dialog which appears.

Once the action above has been completed for each relevant P-Q marker, reset the calculation. The converted marker(s) should remain active, with a label attached.

Once the converted marker has been generated its properties and the properties/visibility of its label can be modified by doing one of the following:

1. By right clicking:
  - (a) Right-click on the converted P-Q marker or its label.
  - (b) Choose the option *Edit Curve Label* from the context sensitive menu.
  - (c) Configure the properties of the dialog which appears as required.
2. By double clicking:
  - (a) Double-click on the converted P-Q marker or its label.
  - (b) Configure the properties of the dialog which appears as required.

More information on configuring the properties of the marker object (VisValue) can be found by visiting the manual section linked [here](#).

### 33.7.7 How to split a P-Q characteristic

Some P-Q starting characteristics may be composed of two or more sub-characteristics. For example, a relay might combine an overcurrent starting element with an Impedance starting element. By default the P-Q diagram will show the overall characteristic resulting from the combination of the sub-characteristics, but in some cases it may be beneficial to show the shape of each sub characteristic in full. To do this it is possible to *split* the P-Q characteristic.

There are two methods to split a P-Q characteristic to show any sub-characteristics which are present:

1. Method 1:
  - (a) Right-click the characteristic. The context menu will appear.
  - (b) Select the option *Split Curve*.
2. Method 2:
  - (a) Double-click the relay operational limits plot avoiding any shown characteristics.
  - (b) In the table *Relays* of the displayed dialog, there is a column *Split Curve*. Check the *Split Curve* box next to the relevant relay that needs to be split.
  - (c) Click **OK** to close the dialog.

## 33.8 The time-distance plot

The time-distance plot *VisPlottz* shows the tripping times of the relays as a function of the short-circuit location. It is directly connected to a path definition so it can only be created if a path is already defined. A path in a single line diagram is defined by selecting a chain of two or more busbars or terminals and inter-connecting objects. The pop-up menu which opens when the selection is right-clicked will show a *Path ...* option. This menu option has the following sub-options:

- **New:** this option will create a new path definition
- **Edit:** this option is enabled when an existing path is right-clicked. It opens a dialog to alter the colour and direction of the path
- **Add To:** this option will add the selected objects to a path definition. The end or start of the selected path must include the end or start of an existing path.
- **Remove Partly:** This will remove the selected objects from a path definition, as long as the remaining path is not broken in pieces
- **Remove:** This will remove the firstly found path definition of which at least one of the selected objects is a member

There are a number of ways to create a time-distance plot but it should be noted that in each case a path must first be defined. The elements which belong to a particular path can be highlighted by setting the colour representation of the single-line diagram to *Other → Groupings → Paths*. To create the plot choose one of the following methods:

- In the single line diagram right-click on an element which is already added to a path definition. From the context menu select the option *Plots → Insert Time-Distance Plot*. *PowerFactory* will then create a new object *VisPlottz* showing the time-distance plot for all distance relays in the path.
- In the data manager or the network model manager right-click on an element which already belongs to a path and select *Plots → Insert Time-Distance Plot* from the context menu. As above, this will create a new object *VisPlottz*.
- Select the elements from the network graphic to be included in the protection path. For more information regarding the graphic selection, please refer to the Section 15.10. Right-click on any

element from the selection. A context menu will appear. Select *Plots* → *Insert Time-Distance Plot*. As a result, a path will automatically be defined for the selected elements and the time-distance plot will be displayed.

- A Path object *SetPath* can be located in the Data Manager in the *path* folder in the project's *Network Data folder*. Select the "Paths" folder and right-click the path object on the right side of the Data Manager. Then select *Plots* → *Insert Time-Distance Plot* from the context menu.

Additionally, multiple paths can be selected in the network graphic or in the Data Manager by using the left mouse button in combination with the CTRL Key. As a result, multiple time-distance plots will be displayed on separate pages. Moreover, the time-distance plot does not automatically display the single line graphic of the selected path on the left side of the plot. In order to activate this option, double click on the plot, navigate to the Scale page and select "Show Section of Single Line Graphic".

### 33.8.1 Forward and reverse plots

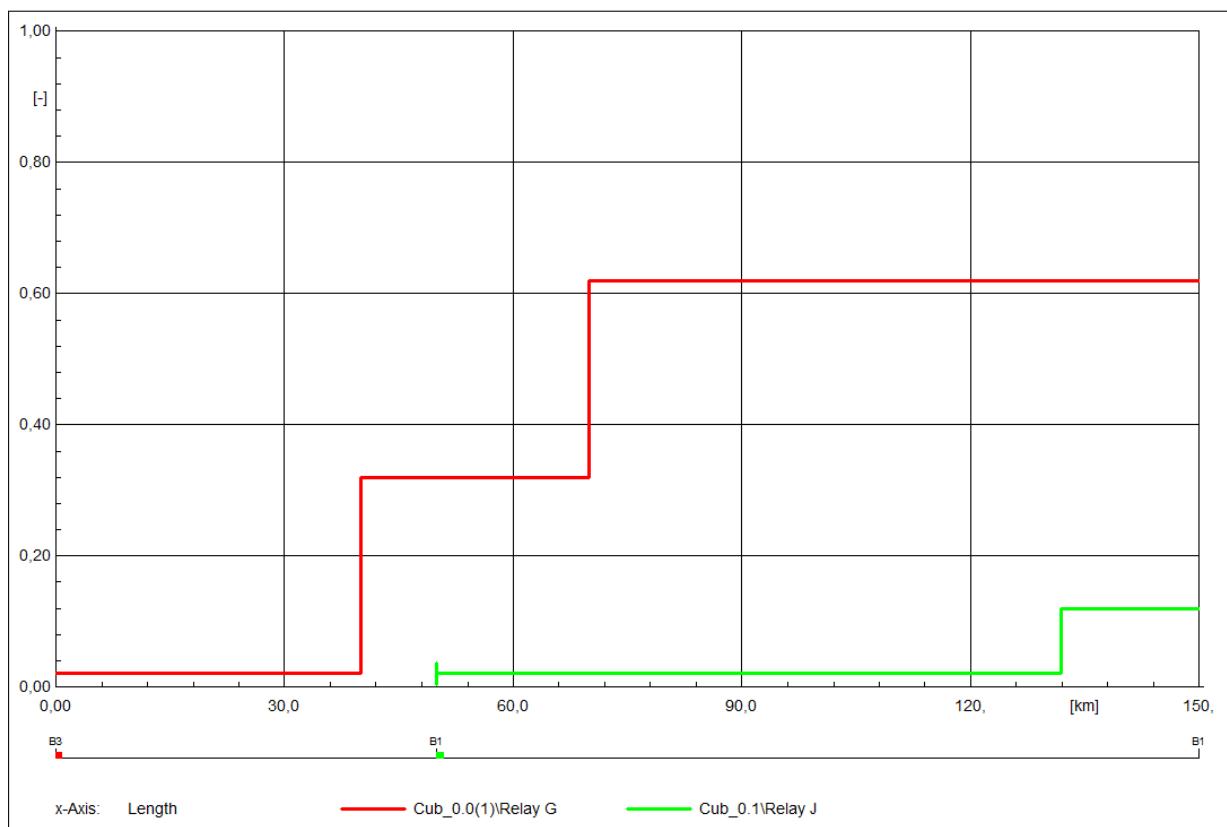


Figure 33.8.1: A forward time-distance plot

Figure 33.8.1 illustrates a forward direction time-distance plot. The diagram shows all relay tripping times for power flows in the forward (left to right) direction of the path. It is also possible to display diagrams which show the tripping times of the relays for power flows in the reverse direction (right to left). Three display options are available for the diagrams parameter specified on the Relays page of the Time-distance plot dialog:

**Forward/Reverse.** Both diagrams are shown in the plot one below the other.

**Forward.** Only the forward direction diagram is shown.

**Reverse.** Only the reverse direction diagram is shown.

### 33.8.2 The path axis

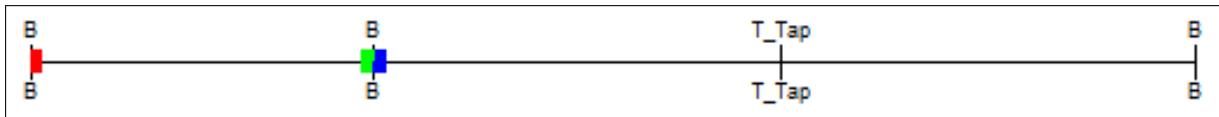


Figure 33.8.2: A path axis

The path axis in Figure 33.8.2 shows the complete path with busbar and relay locations. Busbars/Terminals are marked with a tick and the name. The coloured boxes represent relays and the left or right alignment represents their direction. The path axis can represent many different units according to the user's needs see Section [33.8.5](#)

### 33.8.3 Methods of calculating tripping times

There are two alternative methods for the calculation of the tripping times shown in the plot. To change the method, select the *Method* option in the context menu or double-click the plot to access the time-distance plot dialog and edit the Method parameter on the Relays page.

**Note:** A third option (Coordination Results) associated with the distance protection coordination assistant is also available as a selection option for this parameter. Please see Section [33.13.10.3](#) for more information.

The main difference between the calculation is in relation to their relative accuracy and speed.

**Short-circuit sweep method** The short-circuit sweep method is the more accurate method for charting the variation in relay tripping time with fault position. However, the increase in accuracy comes at the cost of performance. A routine is followed whereby a specified type of short circuit is applied to the network model at regular positions between the first and the last busbar in the path. At each short-circuit location the relay tripping times are established. The user can control the step size between short circuit positions in order to speed up the calculation while ensuring adequate accuracy. Furthermore the user has the option to enable an additional built in algorithm to dynamically modify the step size at critical points on the path, to ensure that the relay operation times are adequately assessed while optimising the calculation time performance. See Section [33.11](#) for information on the short circuit sweep command.

One major advantage of this method is that it will illustrate the under-reaching and over-reaching effects caused by infeeds and outfeeds that may be present in the protection coordination path. It will also take into account all the settings of the relay model including starting settings, polarising settings and the application of any signalling schemes. Further the response of the protection in response to different types of faults can be analysed (e.g. 3ph, LL, LE, LLE etc with selectable fault impedances). Following calculation of the short circuit sweep it may be necessary to adjust the network model (e.g. modify protection settings) and then rerun the sweep.

**Kilometrical method** This method is faster but is far less accurate than the short-circuit sweep method. It is faster because no short circuit calculations are actually carried out and so detailed performance of the relays is not examined. Only a very limited amount of information is extracted from the relay models, the vast majority of the relay configuration is ignored. For distance relays the impedance of the path is compared directly against the reach settings of the relays only. So for example if zone 1 of a distance relay reaches 50km along a path and zone 2 reaches 80km along a path, the plot will display a zone 1 tripping time taken from the relay for the first 50km followed by a zone 2 tripping time for the next 30km.

This method will not account for the starting characteristic of a distance relay or any of the other more detailed features a relay may have. For an overcurrent relay the plot will provide information

limited to the direction of operation and the operation time of definite time elements. The plot is therefore unlikely to reflect the true performance of the protection scheme. However for distance protection in particular, it does serve as an excellent visualisation of the base settings that have been applied to a distance relay.

Ideally, the kilometrical method based plots and the short circuit method based plots complement each other. For a particular study it may be worthwhile presenting kilometrical method based plots as well as short circuit sweep method based plots. For example if settings have been adjusted to take account of infeeds, the short circuit method based plot might illustrate a well coordinated system with the expected reaches applying throughout the system. The corresponding kilometrical plot however, might be grossly distorted illustrating that in order to achieve the good coordination represented by the short circuit sweep, it was necessary to vastly increase the reach settings of the relay. This then raises the question as to whether such a large increase is acceptable.

Conversely, a kilometrical method based plot might indicate that all reaches in the system are entirely as expected but when the results are plotted using a short circuit sweep based plot it suddenly becomes clear that the performance of the relay settings are completely inadequate. This could be due to many reasons which should then be further examined. For example, effects of infeeds, poor selection of starting settings etc.

### 33.8.4 Short-circuit calculation settings

If the method for the calculation of the time-distance plot is set to Short-Circuit sweep, the short-circuit sweep command object *ComShcsweep* is used along with the short circuit command *ComShc*. The commands can be accessed via the corresponding buttons on the Basic Options tab of the Relays page of the Time Distance Plot dialog.

The Short-Circuit command is used to configure the type of short circuit to be carried out by the short circuit sweep command. Here the type of fault can be selected as well as the calculation standard and the fault impedance. Other typical short circuit command parameters are also available for configuration.

The Short-Circuit sweep command is used to define the step size between short circuit applications as well as whether to apply an algorithm to dynamically vary the step size. Selection of the *Iterative* option will enable the algorithm. The algorithm can be adjusted by setting the *Precision* and *step size* parameters. The algorithm initially calculates short circuits along the path at locations separated by the *Step size* parameter value. If a variation in the tripping time is detected between any two adjacent short circuit calculations, the algorithm will sweep the region between the two short circuits using the *Precision* (steps) parameter value.

---

**Note:** The easiest way to recalculate the short-circuit sweep for the time-distance plot is by simply pressing the button . This is only needed when using the Short-Circuit Sweep method.

---

### 33.8.5 The distance axis units

There are a number of possible distance axis units available:

- **Length.** Distance axis is shown in km depending on the line/cable length from the reference relay.
- **Impedance (pri.Ohm).** Distance axis shows the primary system impedance from the reference relay to the remote end of the path.
- **Reactance (pri.Ohm).** Distance axis shows the primary system reactance from the reference relay to the remote end of the path.
- **1/Conductance (pri.Ohm).** Distance axis shows the primary system 1/Conductance from the reference relay to the remote end of the path.

- **Impedance (sec.Ohm).** Distance axis shows the secondary impedance from the reference relay to the remote end of the path.
- **Reactance (sec.Ohm).** Here the secondary reactance from the reference relay is measured on the secondary side.
- **1/Conductance (sec.Ohm).** Distance axis shows the secondary system 1/Conductance from the reference relay to the remote end of the path.

### 33.8.6 The reference relay

The short-circuit sweep method needs to know which relay should be used as the origin of the distance axis. This relay is named the reference relay. If no reference relay is selected, the distance is measured from the beginning of the path. In cases where the reference relay is set, the busbar connected to the reference relay is marked with an arrow.

The reference relay is set using either the graphic or by editing the Time-distance plot dialog. Changing the reference relay graphically is done by clicking with the right mouse button on the relay symbol and selecting Set reference relay in the context menu. If there is more than one relay connected to the selected busbar, *PowerFactory* offer a list of relays which can be used. In the dialog of the time-distance plot the Reference Relay frame is located at the bottom.

### 33.8.7 Capture relays

The **Capture Relays** button enables the user to easily add relays in the selected path to the time-distance plot. In order to delete a relay from the diagram, the respective line in the relay list has to be deleted.

### 33.8.8 Double-click positions

The following positions can be double-clicked for a default action:

- **Axis.** Edit scale
- **Curve.** Edit step of relay
- **Relay box.** Edit relay(s)
- **Path axis.** Edit Line
- **Any other.** Open the “Time Distance” edit dialog

### 33.8.9 The context menu

If the diagram is right-clicked at any position, the context menu will pop up similar to the plot menu described in Chapter 19: Reporting and Visualising Results, Section 19.8 (Plots).

There are some additional functions available in addition to the basic plots-methods for the time-distance plot.

- **Grid.** Shows the dialog to modify the grid-lines.
- **Edit Path.** Opens the dialog of the displayed path definition (*SetPath*).
- **Method.** Sets the method used for calculation of tripping times.

- **x-Unit.** Sets the unit for the distance axis, km impedances,...
- **Diagrams.** Select whether diagrams show forward, reverse or both.
- **Consider Breaker Opening Time.**
- **Report.** This option prints out a report for the position of the relays, their tripping time as well as all calculated impedances in the output window.

## 33.9 Basics of a differential protection scheme

Section 33.2.2 explained how to setup a protection device in *PowerFactory*. After a relay has been created within the network model there are a number of parameters to define in the dialog which appears. This section will describe the basic steps that should be completed in order to specify these parameters for differential protection relays. In many cases the setup is similar to overcurrent relay and consequently only the main differences are described in this section.

Section 33.10 will cover the main graphical tools used for differential protection analysis in *PowerFactory*.

### 33.9.1 Differential relay model setup-basic data page

The basic data page in the relay model (*ElmRelay*) dialog is where the basic configuration of the relay is completed. The procedure is the same as for setting up an over-current relay. Refer to Section 33.3.7.

### 33.9.2 Basic relay blocks used for differential protection

This section provides a brief overview of some of the basic protection blocks that can be found within differential relays in *PowerFactory*. This section only describes those blocks that are unique to differential relays. This manual offers only a brief high level overview of the blocks. For more information on these blocks please refer to the [Protection Devices Library](#).

#### 33.9.2.1 Differential block

The differential block is used for defining the device's operate/restraint characteristic. The differential thresholds, the restraint slopes [%], the slope threshold limit, the slope restraint shape, the restraint current calculation logic and the differential trip delay can all be set in the differential block element.

Inside the Differential Block Type (*TypBiasidiff*) the user can also select the type of the block. This can have a large impact on graphical presentation of parameters of the differential protection (see Section 33.10).

A Harmonic blocking feature is available for use during EMT simulation if the relevant harmonic currents inputs are available. If at least one of the harmonic input currents is above the relevant harmonic threshold the differential trip of the block is inhibited.

#### 33.9.2.2 CT Adapter block

A CT Adapter block can be used where the ratio of a CT's supplying a differential relay need to be normalised. e.g. for a transformer differential protection where the CT's are located at different voltage levels and where a vector rotation may be introduced by the winding arrangement.

Two different types of CT adapter blocks are available: a 3 phase CT adapter and a single phase CT adapter.

A CT Adapter block can be connected:

- To the outputs of a measurement block to simulate the behaviour of a microprocessor differential device which internally compensates the currents coming from CTs
- Directly to the CT outputs to simulate the CT adapter for differential devices which do not include a microprocessor.

## 33.10 Differential Plots

In *PowerFactory* two different types of differential diagrams are available:

- Magnitude biased differential diagram *VisMagndiffplt* (see subsection [33.10.1](#))
- Phase comparison differential diagram *VisPcompdiffplt* (see subsection [33.10.2](#))

Differential plots can be created by right-clicking on the cubicle or the protection device according to the following instructions.

### 1. From the cubicle

- Right-click a cubicle containing differential relays. The context menu will appear.
- Select the option *Plots → Insert Current Comparison Differential Plot* or *Plots → Add to Current Comparison Differential Plot*. *PowerFactory* will create a diagram showing the differential protection plot, or it will open a list of existing plots where the selected device is presented so that user can choose which plot he wants to see.

### 2. From the protection device

- Open a tabular view of the protection device either by editing devices within a cubicle, from the list of calculation relevant objects (Network Model Manager) or from the Data Manager.
- Right-click the  icon. A context menu will appear.
- Select *Plots → Insert Current Comparison Differential Plot* or *Plots → Add to Current Comparison Differential Plot*.

### 33.10.1 Magnitude biased differential diagram

This diagram is available for the following types of differential blocks:

- 3ph
- 1ph
- Restricted Earth Fault

The X-axis represents the Restraint current and Y-axis the Differential current.

The differential characteristic depends upon the differential element (class “RelBiasdiff”) settings and is available also when no calculation has been executed. The restraint current RMS values and differential current RMS values are calculated by the differential element (“RelBiasdiff” class) and are available only when a calculation has been executed. The *1ph* and the *Restricted Earth Fault* differential type calculate an unique working point (a Restraint current value and a Differential current value), the *3ph* differential type calculates three working points (three Restraint current values and three Differential current values)

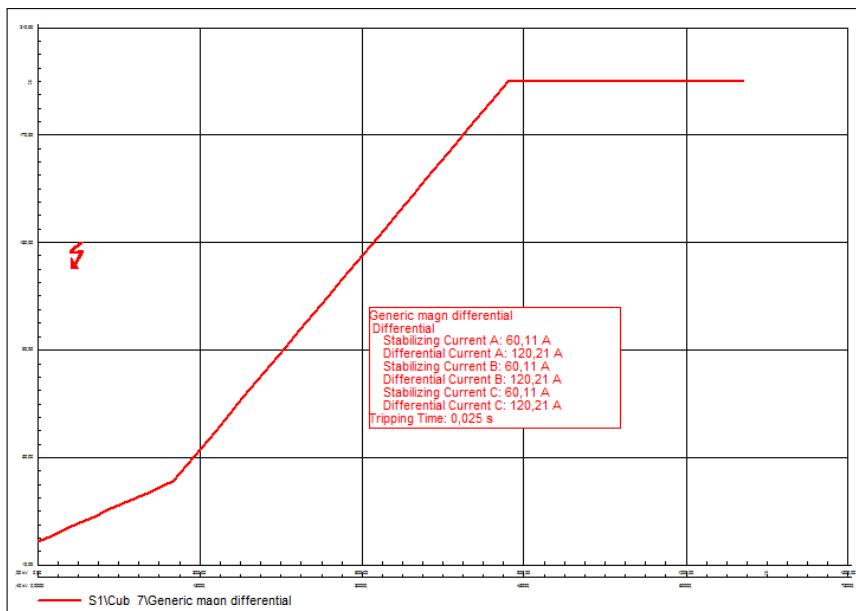


Figure 33.10.1: *PowerFactory* magnitude biased differential diagram for 3ph differential protection element

### 33.10.2 Phase comparison differential diagram

A phase comparison differential diagram (.VisPcompdifflt) is available for the following types of differential blocks:

- 3ph Phase Comparison
- 1ph Phase Comparison

The Restraint Region depends upon the differential element (“RelBiasdiff” class) settings and is also available when no calculation has been executed. The differential current vector is calculated by the differential element and is available only when a calculation has been executed.

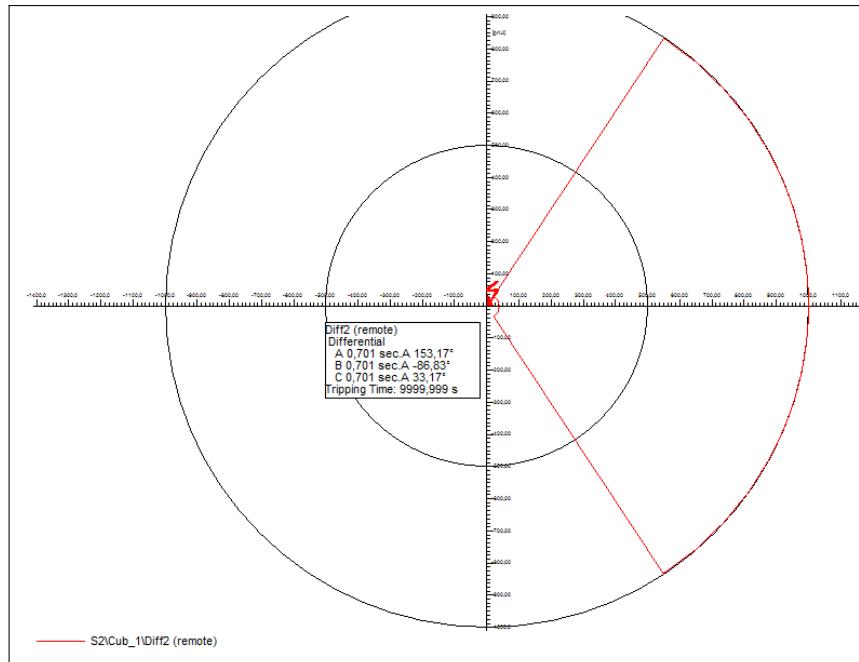


Figure 33.10.2: Example of *PowerFactory* Phase comparison differential diagram for differential protection element of the type 3ph Phase Comparison

### 33.11 The Short-Circuit Sweep command

The Short-Circuit Sweep Command is used to carry out short circuits at intervals along the length of one or more specified paths. The short circuits of the sweep are carried out at constant intervals with the user having control over the size of the interval so as to optimise the resolution of the sweep and thereby improve the calculation time. Furthermore the user has the option to employ an iterative control strategy over the sweep routine, using precision step sizes to ensure that step changes in the monitored variables are not missed. The short-circuit locations can be specified in km, impedance, reactance and 1/conductance in primary ohms or secondary ohms. At each short circuit location, the user can specify the types of fault event to be carried out (e.g. single phase to ground, 3 phase etc.) as well as the real and the reactive parts of any fault impedance. More than one fault event can be carried out at every short circuit location.

The short circuit sweep includes its own results file in which any result variable available in *PowerFactory* can be monitored. For each fault event at each location, results can be stored in the results file and made available for post calculation analysis and presentation in plots. The dialog of the command is illustrated in Figure 33.11.1.

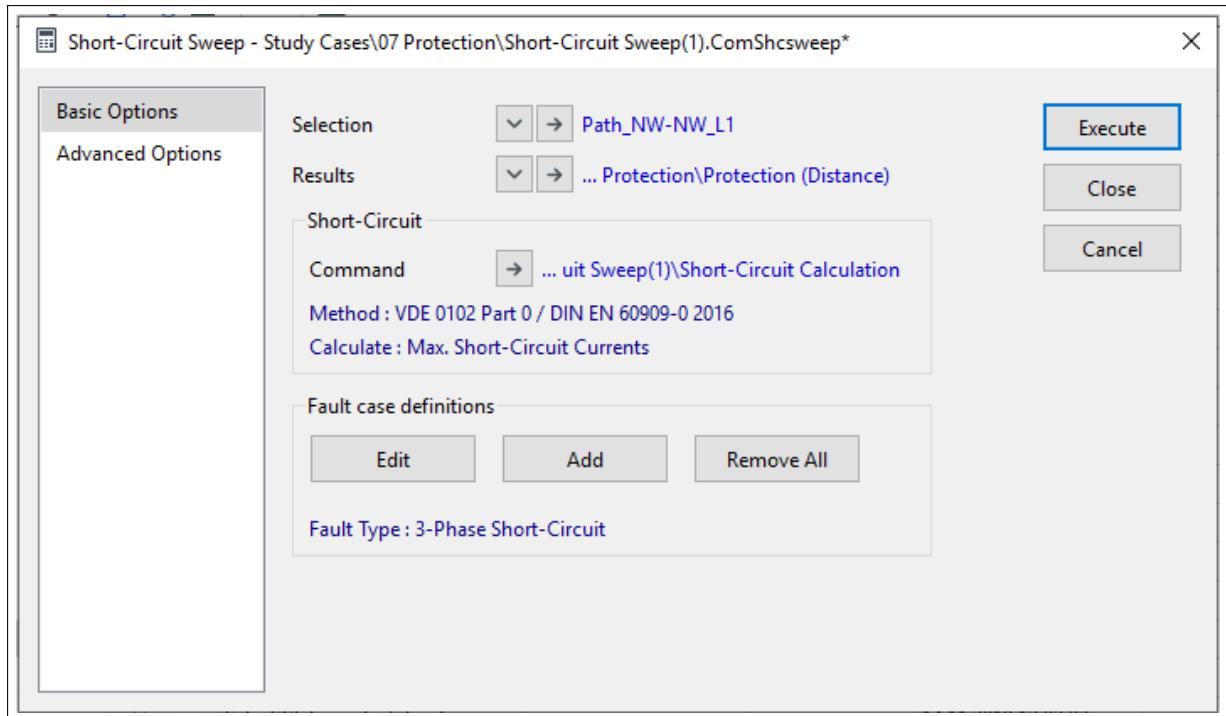


Figure 33.11.1: Short-Circuit Sweep command

In the *Selection* field a path or set of paths may be selected over which the short circuit sweep shall be carried out.

The *Results* field specifies the results file which shall be used to store the results. In this results file it is possible for the user to specify the result variables to be monitored throughout the calculation/

The *Command* field references a short circuit command. This command should be used to specify the short circuit method and other settings e.g. maximum or minimum short circuit current or LV/Momentary, LV/Interrupting or 30 Cycle Currents/Voltages (for ANSI calculations). It is not used to specify the fault impedance or the fault type. This is specified in the fault events.

The fault events are specified using the fault case definitions field. By choosing the Add button a new fault event is created and this can be configured by the user to specify a fault type and impedance. Multiple fault events can be defined and later edited or removed using the Edit button or Remove All button respectively.

On the second page of the Short circuit Sweep command dialog, the user can configure the sweep routine and whether an iterative strategy should be used. In general the iterative approach is significantly faster but can lead to more inaccurate results. The iterative step-size deduction employs interval bisection to find the locations at which the tripping times of relays change. The algorithm starts with the full length of each section of line of greater length than the specified Min. line length. It calculates a fault at either end of the line and looks at the responses of the relays. If the algorithm detects that the relay operating time changes for any relay, then it calculates a short circuit in the middle of the line. Depending on whether the change happened in the first or the second half of the line it selects the corresponding section and repeats the sequence until the location only changes by the specified Precision. This result is recorded and then the process is repeated with the section between the found location and the end of the line, until either no change is detected or the next position is the line end. The specified step size is used, when two subsequent iterations yield positions which are within two times the specified Precision of one another. In that case it is assumed that there is an IDMT characteristic in effect and the algorithm continues using the fixed step size approach for this line.

The short circuit sweep command is used by a number of different features in *PowerFactory* including the time-distance plot (see Section 33.8), the Protection Audit tool (see Section 33.15) and Short-Circuit

Sweep plots (see Section 33.12). In each of these cases its use has been integrated into the feature so that the short circuit sweep does not need to be created, instead only configuration of the Short-Circuit Sweep is actually required.

## 33.12 Short-Circuit Sweep Plots

Like the Time-Distance Plot described in Section 33.8 the Short-Circuit Sweep Plot (*VisSweepplot*) allows the user to tap into powerful capabilities of the short circuit sweep command *ComShcsweep*. Unlike the time distance-plot, which can only display the variation in tripping with respect to fault location across a path, the short-circuit sweep plot can be used to display the variation of *any* network parameter with respect to fault location and fault type, across a particular path. For example, it is possible to easily plot the variation in fault current or active power, flowing in a line in response to different fault locations, and fault types, across a particular path which may or may not incorporate the monitored line. Some typical plots are illustrated in Figure 33.12.1.

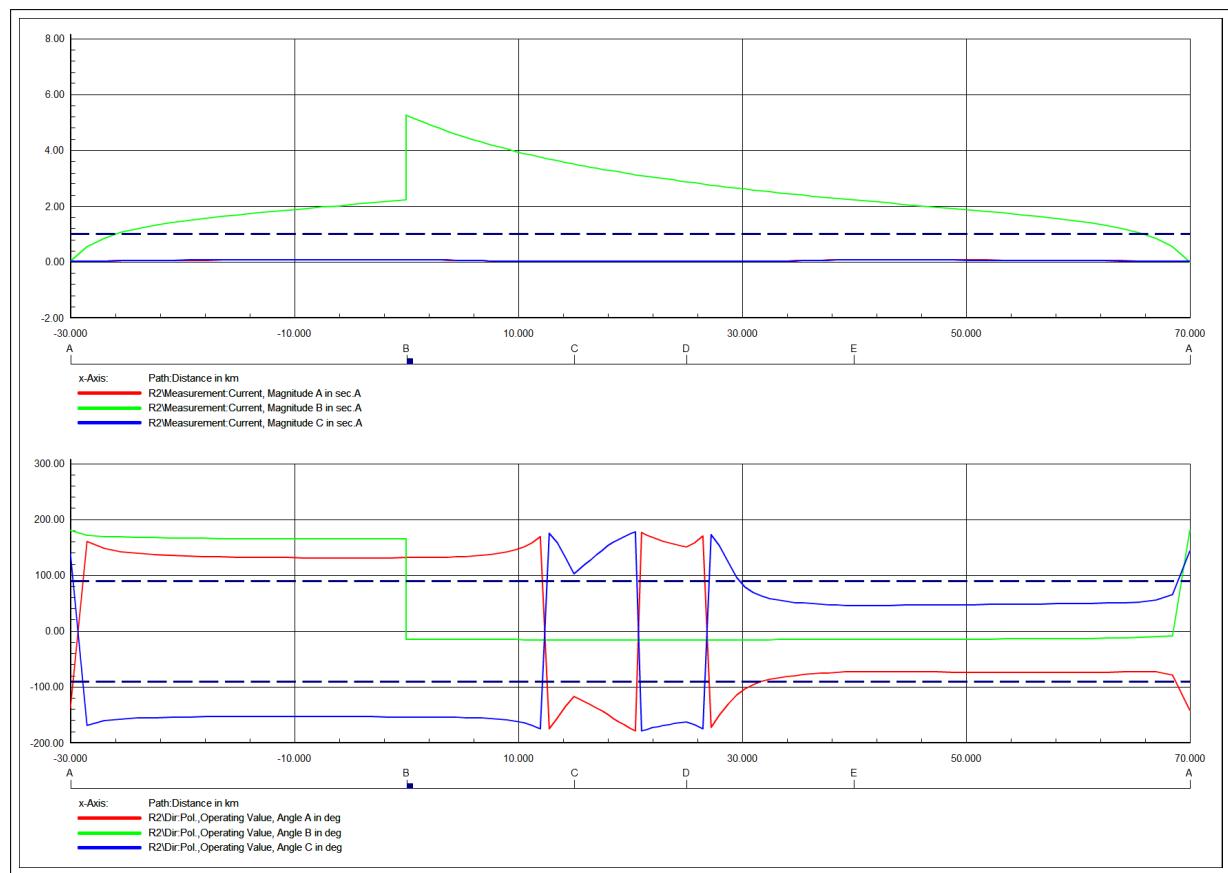


Figure 33.12.1: Typical Short-Circuit Sweep Plots

The plot has clear applications in the protection field but it may also be used for other short circuit applications if appropriate.

The user first defines the path which is to be swept by the short circuit sweep command, the result parameters to be recorded are then defined and finally the short circuit cases to be applied during the sweep are specified. The sweep is then executed and the results collected in a results file. The plot accesses this results file in order to display the final characteristics. The x-axis of the plot can be selected to length, impedance, reactance or 1/conductance, while the units of the y-axis depend on the result parameter being examined. Each plot generated is specific to a particular fault case. For example, one plot may relate to three phase, zero impedance faults, whilst another may be related to

single phase to earth faults, with five ohms of fault resistance. However, it is straightforward to toggle between the different cases using the plot's setting dialog.

For protection purposes, the plot allows the detailed behaviour of a relay to be examined in response to the faults of the sweep. For example, a path defining a closed ring may be swept and the directional element of a relay operating on the ring examined. In this case the plot might be used to gain insight into how the angle between the extracted polarising and operating quantities of the relay change with fault location and ultimately how the detected direction of the fault changes. This information may be useful in explaining the tripping behaviour of the relay model, diagnosing problems with the settings or design of the relay and finally in the selection of appropriate relay settings. The plot dialog also provides an option to display setting thresholds taken from a user defined relay element. These thresholds can be compared against the results of the sweep to help the user identify key transition points or values in the plot.

### 33.12.1 Configuration of Short-Circuit Sweep plots

A feature has been included in the new protection graphic assistant tool (described in more detail in Section 33.17.2) to accelerate the process of generating useful plots specifically for protection purposes. In addition to the method described there it is also possible to create these plots directly from the *Insert Plot* icon in the main icon bar, or by right-clicking a single or multiple path objects in the Data Manager and choosing *Plots* → *Insert Short-Circuit Sweep Plot*, or by right-clicking on an element in the single line diagram belonging to a path and choosing *Plots* → *Insert Short-Circuit Sweep Plot*.

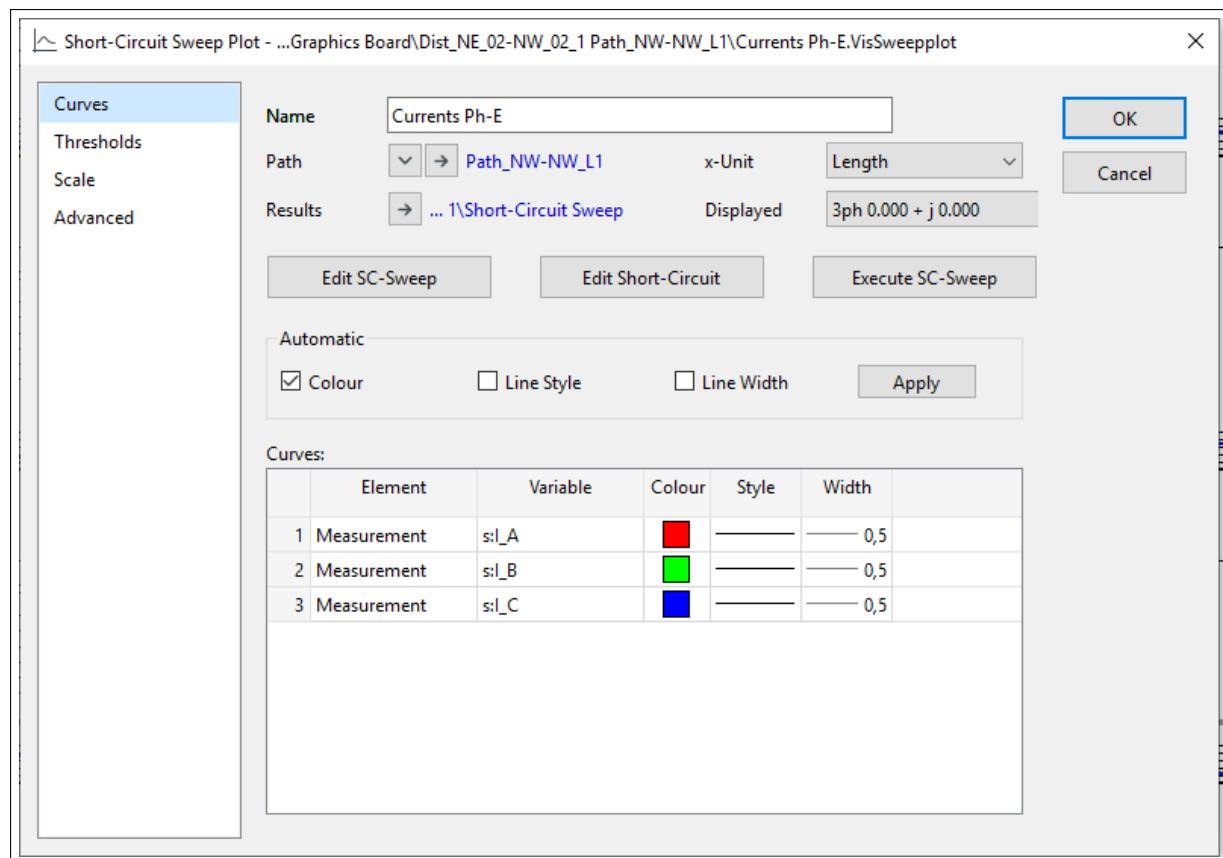


Figure 33.12.2: Short-Circuit Sweep Plot - Curves page

Moreover, the selection of elements from the network graphic can be directly used to create a Short-circuit sweep plot. For more information regarding the graphical selection, please refer to the Section 15.10. Right-click on any element from the selection. Select *Plots* → *Insert Short-Circuit Sweep*

*Plot.* A path will automatically be defined and the short-circuit sweep plot will be displayed. A Path object *SetPath* can be located in the Data Manager in the *path* folder in the project's *Network Data folder*. Select the "Paths" folder and right-click on the path object on the right side of the Data Manager. Then select *Plots* → *Insert Short-Circuit Sweep Plot* from the context menu. Additionally, multiple paths can be selected in the network graphic or in the Data Manager by using left mouse button in combination with the CTRL Key. As a result, multiple short-circuit sweep plots will be displayed on separate pages. In whatever way the plot is created, once the plot has been created it is possible to configure the plot by configuring the plot dialog as shown in Figure 33.12.2.

On the *Curves* page of the dialog the Path can be selected if required. The x-axis unit can be set to either length, impedance, reactance or 1/conductance. A pointer to the results file used by the Short-Circuit sweep is displayed. It should be noted that the results file is stored within the Plot Page object in the data manager, whilst the short circuit-sweep itself is stored within the study case.

The short circuit sweep (see Section 33.11) can be configured and executed using the three buttons named *Edit SC-Sweep*, *Edit Short-Circuit* and *Execute SC-Sweep*. The *Edit SC-Sweep* gives access to the Short-Circuit Sweep command itself (*ComShcsweep*). Here different fault cases (i.e. different types of fault e.g. 3 phase, single phase to ground etc., with different fault impedances) can be added to the analysis. The algorithm which controls the number of short circuit locations examined can be configured on the Advanced options page of the object. The sweep will carry out a series of short circuit calculations at intervals along the length of the defined path and at each location it will record the result variables as specified in the results file. The user can add or remove result variables to/from the results file as required. The sweep will repeat the series of calculations for each fault case defined.

The short circuit calculation method to be used can be configured using the *Edit Short-Circuit* button. Note that the Fault types are not configured here, rather they are configured as described in the previous paragraph. Each Short-Circuit sweep plot relates to a single fault case. The fault case to be displayed is selected using the *Displayed* parameter. The drop down list will contain all fault cases specified in the short circuit sweep command. To execute the short circuit sweep the *Execute SC-Sweep* button may be used. Alternatively a feature has been included in the new protection graphic assistant tool to manage the updating of plots including the execution of Short-Circuit sweep commands. This is described in more detail in Section 33.17.3.

The y-axis parameters to be displayed are specified in the *Curves* table. The colour, line style and line width can all be configured as required.

On the Thresholds page of the plot shown in Figure 33.12.3 it is possible to add additional characteristics to the plot representing the setting thresholds of various relay elements. By plotting the thresholds along with the results it is possible to easily observe where thresholds are exceeded and where they are not. Again the colour, style and width of the threshold characteristic can all be configured as required.

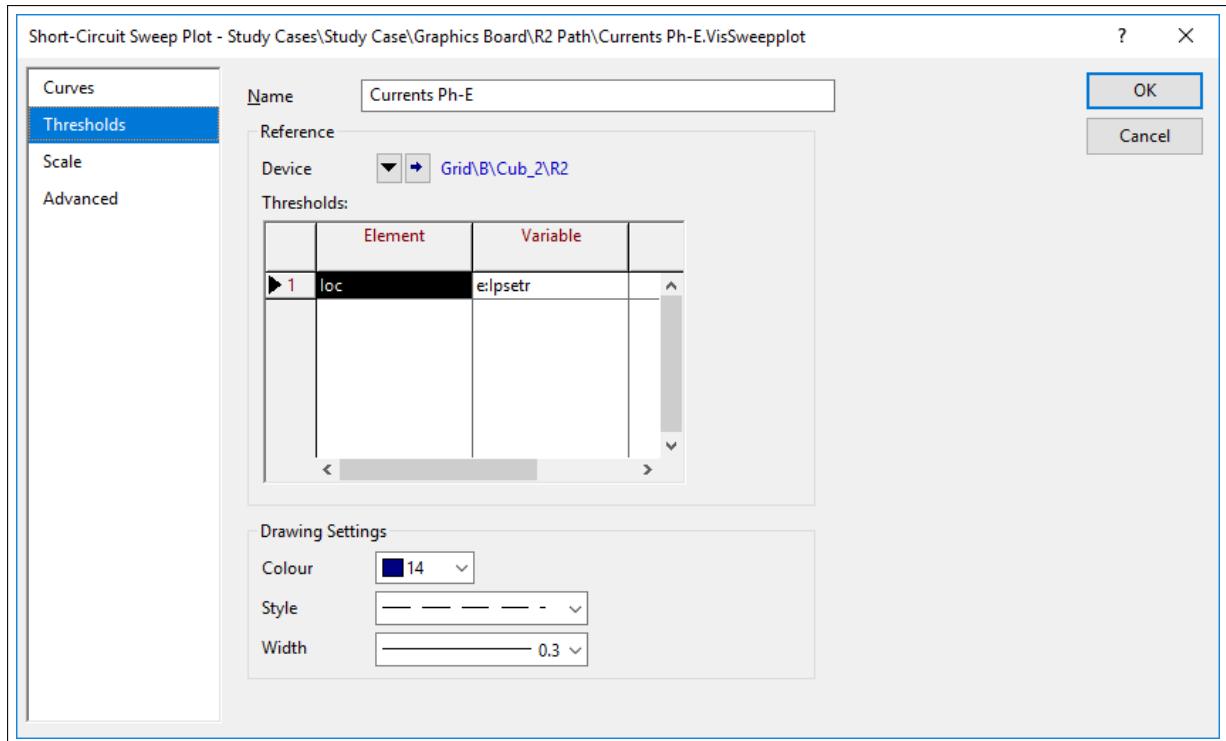


Figure 33.12.3: Short-Circuit Sweep Plot - Thresholds page

## 33.13 Protection coordination assistant

*PowerFactory* includes a protection coordination assistant that can assist with automatically determining settings for overcurrent and distance protection devices. This section explains how to use this tool.

### 33.13.1 Technical background

This section provides a brief overview of overcurrent and distance protection coordination. The user may wish to skip this section and move directly to the sections about configuring the tool if they are already familiar with the basic principles of protection coordination.

#### 33.13.1.1 Overcurrent coordination

Overcurrent protection coordination philosophies vary widely throughout industry so it is difficult to describe a definitive approach. However, in this section some basic principles of overcurrent coordination will be described along with the necessary steps to derive equations for an automatic coordination based on the described philosophy.

Overcurrent protection is most frequently applied in radial networks. For other situations where faults are fed from multiple directions for example where there are separated sources in the system, ring structures or a double circuit feeding two coupled busbar sections then there will be a requirement for overcurrent devices to act on a measured fault direction in addition to the current magnitude.

If we focus on a the simple radial structure supplied from a single equivalent source, the overcurrent philosophy for any overcurrent relay protecting equipment in that structure could be stated as follows:

- An overload stage should protect the connected equipment in case of currents higher than the

permissible equipment current that could flow under normal operating conditions. Normally a security factor of 20% is applied to avoid nuisance tripping. A formalised equation could state that the overload stage should be set to 120% of the nominal equipment current rating.

- The instantaneous definite-time stage is required to detect the minimum short circuit current expected to be seen for faults in the primary protection zone. For this purpose the minimum short circuit current could be calculated and an additional security factor of 20% applied. A formalised equation could state that the instantaneous stage should be set to trip at 80% of the minimum short circuit current observed for faults located in the primary protection zone.
- Whether a third stage, the short-circuit stage, is applied might depend on the protection philosophy. Where it is applied this stage could act as a backup stage for faults located on adjacent equipment (in the backup protection zone). For this purpose as for the instantaneous stage, minimum faults minimum short circuit current could be calculated, but in this case with faults applied in the backup protection zone. Again, a security margin of 20% could be applied. In this case a formalised equation could state that the tripping threshold should be set to 80% of the minimum short circuit current in the backup protection zone, i.e. in the next connected equipment.

For a fault in a radial feeder structure with a relatively large source impedance where the source impedance does not vary much if the fault is applied at different locations along the length of the feeder (e.g. where there are no transformers, or long lines), then the currents also do not differ much when the short circuit location is varied. In these cases, the selectivity is often ensured using time grading. This means, that for a given fault, an upstream device detecting a fault in its backup zone will be set so as to trip later than a downstream device detecting the fault in its primary protection zone. The time difference between the specified tripping time of the upstream device as compared to the downstream device is known as the grading time (and could typically be in the order of 300ms).

The coordination assistant in *PowerFactory* is able to automatically carry out network calculations resulting in the specification of current and time tripping thresholds for overcurrent relays based on formally specified equations and grading times. Where directional elements are specified then this setting is also considered in the coordination.

### 33.13.1.2 Distance coordination

A distance protection scheme works by continuously measuring the voltage and current on a protected circuit. These values can then be used to calculate an equivalent impedance. This impedance can then be compared to a “reach” setting and the basic idea is that the relay should trip if the measured impedance is less than the reach setting.

On an unfaulted circuit the voltage will be high (normally tens to hundreds of thousands of volts) and the current much lower (normally tens to hundreds of Amps). Therefore, according to Ohms law, the normal load impedance might typically be in the order of a few hundred Ohms.

Consider now a three phase bolted fault on a transmission circuit. The voltage falls to zero at the point of fault whilst the current increases in proportion to the source voltage and the impedance between the source and the fault. At the near end of the circuit where the protection relay and measuring CTs and VTs are located, the voltage will drop and the current will increase. The ratio of the voltage at the source to the fault current will be the impedance of the line to the point of the fault. Using this principle, relays can be set to protect a certain ‘zone’ of a line and accurately discriminate between nearby faults and more distant faults.

In practical, distance protection relays use so-called “polarising” voltage and “operating” current quantities to measure the impedance and to determine whether a fault is “in zone” or “out of zone”. A separate directional element is often also employed to determine whether the fault direction is “forward” or “reverse”. This element also uses “polarising” voltage and “operating” current quantities. For faults close to the relay the measured voltage may drop so low that the relay is unable to accurately determine a fault direction. In modern numerical distance protection relays, the polarising voltage quantities used by the directional element often include a memory buffer component that allows the relay to determine direction more accurately for such faults based on the voltage present before the fault occurred. Further

detail on this and other aspects of distance protection can be found in many reference texts and the interested user should refer to these for further information.

For the purpose of coordination, a basic distance protection scheme might consist of three zones of protection configured as follows:

- Zone 1 that covers 80 % of the protected circuit and is set to instantaneous trip.
- Zone 2 that covers 100 % of the primarily protected circuit and a portion of the next adjacent circuit. Zone 2 protection must be time delayed so that discrimination can be achieved with the zone 1 protection on the adjacent circuit. A typical time delay is 400 ms.
- Zone 3 protection provides backup protection for the adjacent circuit and is often set to the impedance of the primarily protected circuit + 100 % of the adjacent circuit. It has a longer time delay than zone 2, typically 800 ms. For older mho relays this zone may be offset so as to provide a degree of reverse reach (in addition to the forward reach) so as to provide backup for bus protection systems.

This is quite a typical configuration but there are many other configurations in use and different parties responsible for protection coordination have different philosophies for determining how the reach and time settings of a distance relay should be determined. In general the various reaches for each zone are normally determined from the network impedance, reactance or resistance of the protected lines. However in each case different factors and offsets are applied according to company preference. These preferences can usually be collated as a collection of setting rules.

If a network model has already been configured in *PowerFactory*, *PowerFactory* usually has access to the network impedance data and topological connection data. If *PowerFactory* is also provided with key facts about the relevant setting rules, then it is able to automatically calculate a set of reach settings for each relaying location in the network in accordance with the relevant philosophy. The protection coordination assistant is a tool which has been provided to facilitate this feature.

The protection coordination assistant automatically determines distance protection settings for the relays or relaying locations it is provided with. The basic functionality of the coordination tool can be illustrated with reference to an example network. Consider the simplified transmission network shown in Figure 33.13.1. This network contains four busbars, three transmission lines along with associated generation and load. The locations where distance protection devices are located are indicated with a blue circle, and the direction in which they are “reaching” is indicated with blue arrows. Line impedance parameters are shown above the centre of each line.

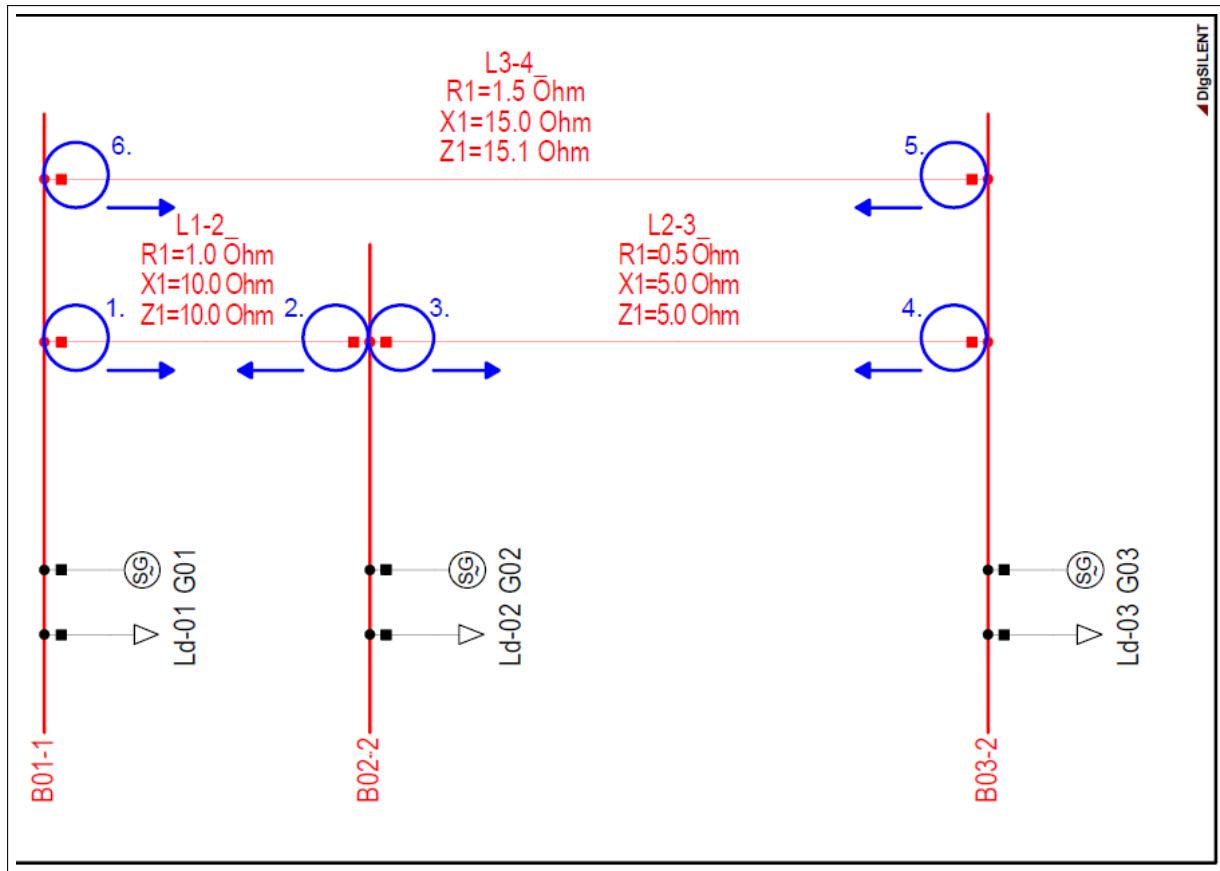


Figure 33.13.1: Example simplified transmission system for the protection coordination example

For the purpose of this illustration, the coordination assistant might be required to determine the settings for three zones and an overreach zone for each relay location shown. In this example, there are six locations where settings would be determined. Therefore the coordination assistant would calculate a minimum of 24 settings and maybe even more (depending on how the reach was specified in the relays). Potentially the coordination assistant could be used to calculate the settings for an entire network. In such a case, the potential benefits of using the tool are immediately apparent.

### 33.13.2 General Handling

Protection coordination is carried out in two steps. In the first step settings are calculated using the protection coordination assistant command itself (*ComProtassist* ). In the second step the results of the coordination are reported and where appropriate, applied to the relays in the network model. For the second step a separate command called the protection coordination results command (*ComCoordreport* ) is used.

In order to carry out an automatic coordination, it is necessary to define the following things:

1. A network model suitable for the accurate calculation of load flow and short circuit results
2. The relay locations
3. The protection philosophy setting rules

Relaying locations are specified by first adding relay (*ElmRelay*) objects to the network model (if they are not already present in the model) and by secondly specifying for which of those relays, settings should be calculated, with the latter specification being made in the protection coordination assistant command (*ComProtassist*) itself.

The distance and overcurrent protection setting rules are also specified in the protection coordination assistant command. Once the Protection coordination assistant has been configured, it is executed and *PowerFactory* will then use the data available to it to calculate settings.

The following sub sections will go through the different options available for the configuration of the protection coordination assistant and the protection coordination results command. They are organised according to the page on which the options are specified.

### 33.13.3 Basic Options

#### Coordinated protection devices

The coordination assistant can be used to carry out both an overcurrent and distance protection coordination at the same time or alternatively each coordination can be carried out separately. The check boxes are used to specify which coordinations are to be considered by the assistant.

---

**Note:** The coordination assistant is available with both licenses *Time-Overcurrent Protection* and *Distance Protection*. However, please note, that the coordination of distance protection devices requires the *Distance Protection* license.

---

#### Protection devices

The protection relays for which the coordination will be carried out can be specified via this option. The devices are selected by clicking on the icon. Devices can either be selected directly or a grouping element which contains devices, such as a set or a path can be selected.

#### Calculation commands

This field is only visible if the check box *Overcurrent Protection* is active. This is because load flow and short circuit calculations must be carried out in order to carry out an overcurrent coordination whereas for the distance protection coordination only the network data is required. Within the *PowerFactory* database the commands are located in the study case folder and can be accessed by clicking on the icon to change the calculation settings. For the overcurrent coordination, the coordination assistant will determine maximum and minimum short circuit currents for each protection device automatically based on the fault types provided in the assistant and the locations of the devices. The following buttons are used to handle the fault events:

- *Edit Faults*: Opens an overview windows with the existing fault events.
- *Add Fault*: Creates a fault event (*EvtShc*) where the fault type and fault impedance are specified.
- *Clear All Faults*: Deletes all existing fault events.

If no fault events are defined, the coordination assistant will use the selected fault type in the short circuit command.

#### Results

The result file in which the results of the coordination(s) will be stored can be specified with the parameter *Result File*. Under many circumstances it may not be necessary to adjust this parameter but if necessary a specific result file can be selected by using the icon. The result file dialog itself can be accessed by clicking the icon.

## Reporting

The *Command* parameter is used to reference an associated protection coordination results command. As mentioned in Section 33.13.2, the coordination process is a two step process where first results are calculated and secondly results are reported. The checkbox option here is used to specify whether the two steps are executed in one action (by execution of the Protection Coordination Assistant only) or whether the two steps are executed separately. When the *Reporting* check box is selected, the two steps are executed in one action, with the protection coordination results command as specified in the *Command* parameter used for the second step. By clicking on the → button it is possible to access and configure the referenced command. If the *Reporting* check box is not selected, the coordination results command must be executed manually as a second step once the first step is complete.

### 33.13.4 Overcurrent Protection

The assistant is able to coordinate overcurrent devices based on the phase currents observed by the device under specific fault conditions. Different current equations can be defined for different equipment types. The assistant automatically selects the relevant equation based on the relay location, direction and connected equipment.

The equation sets can be split into two categories:

- **Branch elements:** Line, Transformer (HV-side), Transformer (LV-side), Busbar
- **Single port elements:** Load, Motor, Generator

Equations can be activated or deactivated according to the network layout and protection philosophy. As a minimum, the equations for lines and loads should be active in order to calculate settings for all equipment types. If for example only equations for lines are given, then relay settings for all elements classed as branch elements are calculated according to the line equations. The same logic applies for single port elements such as motors or generators. By default the equations for loads are applied to all single port elements if alternative equations are not available.

Figure 33.13.2 shows an example network including equipment types and overcurrent protection devices considered by the coordination assistant. The network structure is radial with a uni-directional power flow because only one source is connected, at Station A. Based on the connected equipment types, for each protection device the assistant selects a corresponding equation set. In this example the following equations are used:

- Line equations: *R AB* and *Fuse*
- Transformer HV equations: *R BC HV*
- Transformer LV equations: *R BC LV*
- Motor equations: *R ASM*
- Busbar equations: *R A Bus* and *R AB Bus*

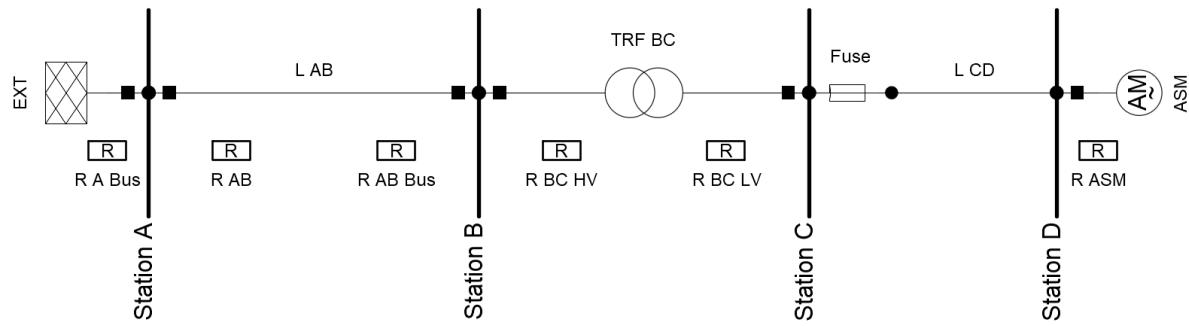


Figure 33.13.2: Example network showing equipment types and overcurrent devices considered by the coordination assistant.

If the network has more than one source, depending on the location of a fault, the power that flows through a relay to the fault can flow in one of two directions. In other words the relay can be subjected to bi-directional power flows. In such cases the user has little choice but to utilise directional discrimination. Figure 33.13.3 shows an example network where two sources have the potential to produce bi-directional power flows in the protection devices. Depending on the fault position every device in this network is potentially able to see a bi-directional power flow. In order for the assistant to coordinate the correct devices, the option *Directional* needs to be activated on the *Protection* page of each relay element *ElmRelay*.

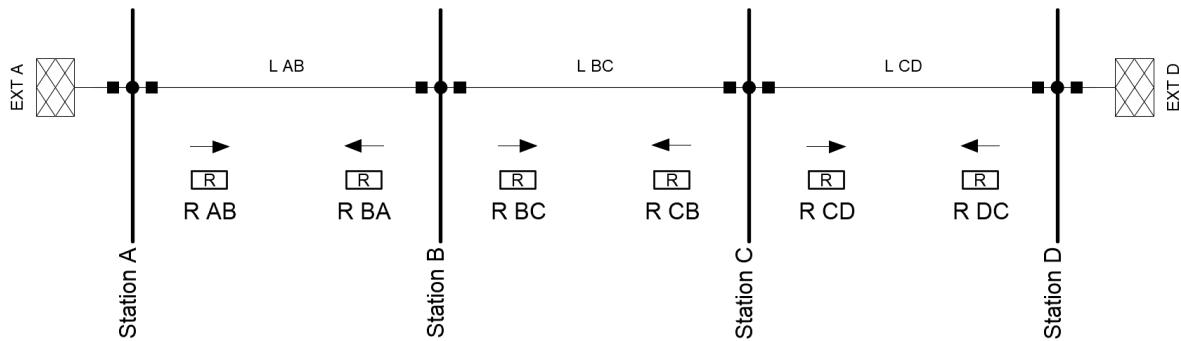


Figure 33.13.3: Example network with two sources for a directional overcurrent scheme.

Based on the given current equations the assistant determines the current settings for each overcurrent protection device. Depending on the power flow direction and the directional setting of each relay, relay grading pairs are identified and the specified time grading applied.

### 33.13.4.1 Definition of current equations

Three overcurrent stage equations and one fuse equation can be defined for each equipment type. For each equipment class the following input fields and options are available:

- *Direction*: An individual direction setting is specified for all overcurrent stages used for the equipment class. The available directional options are non-directional, forward or reverse.
- *Overload*: For the overload stage a current equation can be defined using the available keywords. Different characteristics can be selected in the *Overload characteristic* drop-down menu. It is possible to select a definite-time characteristic or an inverse definite minimum time characteristic such as inverse, very inverse, extremely inverse or long-time inverse.
- *Short-circuit and Instantaneous*: The equations for two definite-time overcurrent stages.
- *Fuse*: If the equipment type is protected by a fuse element, the equation defined in this field will be considered to determine the fuse rating.
- *Max. current*: The maximum current which is seen by the overcurrent protection device is required for every set of equations. If this current is not specified, the assistant will not calculate settings for that equipment type.

The assistant offers a set of pre-defined keywords which are in fact current variables to determine the current setting of each overcurrent stage. Current results from both load flow and short circuit calculations are available. For short circuit currents the assistant calculates minimum and maximum currents considering all defined fault types and the short circuit command specified in the *Basic Options* page.

---

**Note:** The assistant determines minimum and maximum currents based on the application of faults at key locations. These locations are associated with protection areas identified automatically from the locations of the relays and the surrounding network topology. The location setting specified in the short circuit command is not used in these cases.

---

The following list explains each available keyword in detail:

- **I\_nom**: nominal current of the protected element. Please note, that loads do not offer a nominal current which can be utilised by the coordination assistant. For busbars this keyword is only meaningful when a nominal current is specified in the busbar's type.
- **I\_Idf**: load flow current measured at the relay location as a result of the load flow specified in the page *Basic Options*.
- **I\_start**: starting current of asynchronous machines. This keyword is only available when specifying *Motor* equations.
- **I\_minAB**: minimum short circuit current for a fault located at the remote limit of the first protection area (AB) and measured at the relay location. For the example network in Figure 33.13.2 the current I\_minAB for device R AB would be for a fault at 100% of L AB seen from station A i.e a fault effectively applied at Station B.
- **I\_maxAB**: maximum short circuit current for a fault located at the local limit of the first protection area (AB) measured at the relay location. For the example network in Figure 33.13.2 the current I\_maxAB for device R AB would be for a fault at 0% of L AB seen from station A. i.e. a fault applied just on the line side of the CT at Station A.

The remaining available keywords are: **I\_minBC**, **I\_maxBC**, **I\_minCD**, **I\_maxCD**. These currents are determined in a similar way to I\_minAB and I\_maxAB but for the second (BC) and third (CD) protection areas as determined from a topological search of the equipment and relays located in the vicinity of the protection device. For the example network in Figure 33.13.2 the second protection area for device R AB is between stations B and C and the third protection area between stations C and D.

The equations are entered using the described keywords and standard mathematical operators (+,-, /,\* ,max() and min()). When using the max() and min() operators, a maximum of two expressions can be compared with each expression separated by a ";" character.

An example of typical usage for line equations is given below:

- Overload  $I_{setting} = 1,2 * I_{nom}$
- Short-circuit  $I_{setting} = 0,8 * I_{minBC}$
- Instantaneous  $I_{setting} = 0,8 * I_{minAB}$

#### 33.13.4.2 Definition of minimum time delays

For each equipment class for which equations are defined, one minimum time delay can be defined on the *Timers* tab. The delay specifies a fixed minimum timer setting for the fastest overcurrent stage. Normally such time delays are set to zero. But for example if the overcurrent protection is only the backup for a differential protection it might be required to apply a minimum delay for each overcurrent device to grade with the differential protection.

When defining the equations used for *Busbars* an additional checkbox *Assume reverse interlock* is available. The intention of a reverse interlock is to allow a very fast time setting to be employed by busbar protection that would not normally coordinate with downstream protection. The fast acting stage should only be released by the interlock for busbar faults. If this check box is selected activated, the option prevents time grading from being applied to the instantaneous stage. Instead the assistant sets the time delay of the instantaneous stage to the specified minimum time delay.

#### 33.13.4.3 Fuse Characteristic definition

Under the tab *Fuse Characteristic* a generic fuse tripping single-line characteristic (e.g. an average melt curve) can be defined on a per unit basis. The table *Nominal currents* is used to specify a range of available fuse ratings which the coordination assistant can use to scale the generic characteristic deriving a separate characteristic for each fuse rating. The assistant then selects an appropriate fuse characteristic according to the specified current equations and time grading.

### 33.13.5 Distance Protection

The majority of settings rules are defined across the various tabs located on this page of the command dialog. Most setting rules are defined on a per zone basis.

The general tab of this page is illustrated in Figure 33.13.4.

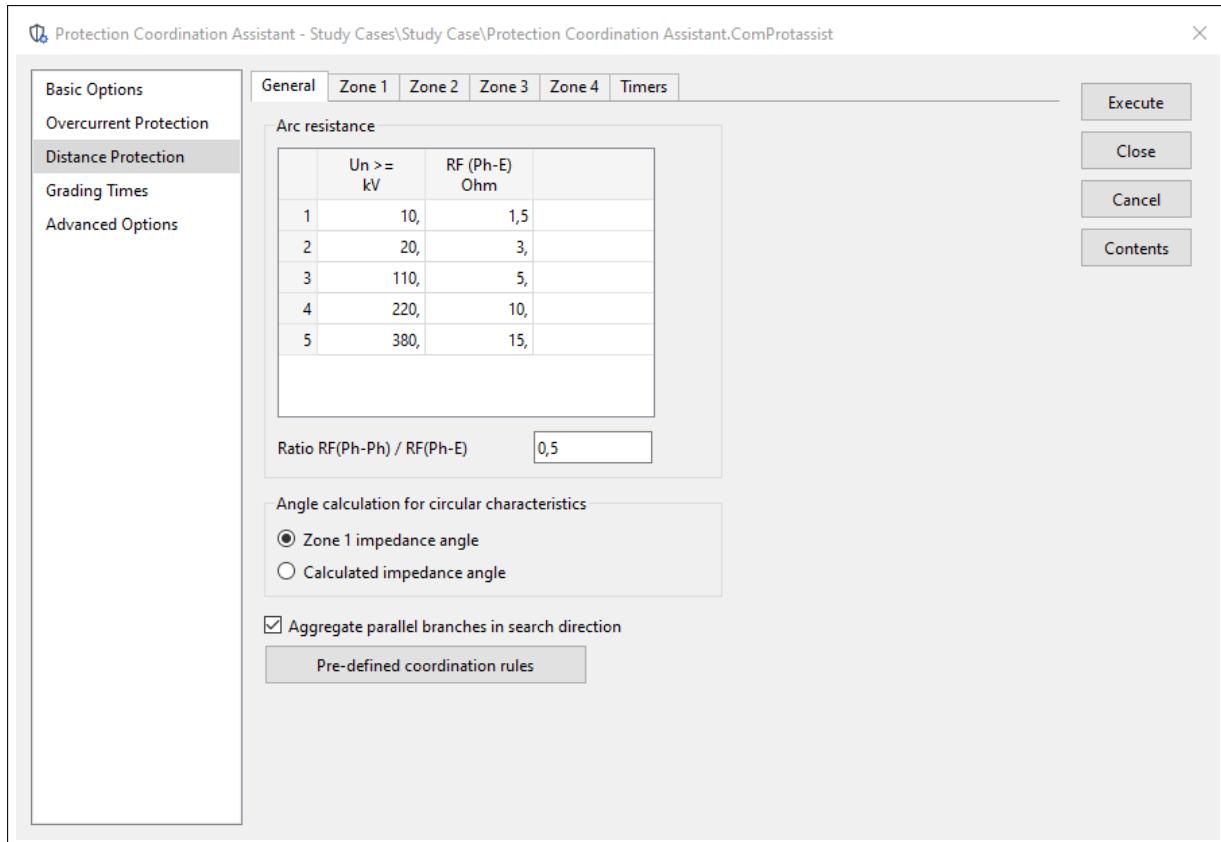


Figure 33.13.4: Protection coordination assistant command - General tab of Distance Protection page

Most distance relays have a resistive reach which exceeds the resistance of the line being protected. This is to ensure that faults are detected even when fault resistance is present. In order to try to anticipate probable values of fault resistance an arc resistance is usually built into the specification of setting rules for the resistive reach. The arc resistance is dependent on various factors but voltage level can be used as the main differentiator.

The table in the *Arc resistance* field allows a user to specify the arc resistance to be considered at different voltage levels. This means that if settings are being calculated for relays located in different voltage networks, the assistant can automatically consider a different arc resistance for each of those relays. The equations used for calculation of the arc resistance are defined on the tabs associated with each zone and will be described in subsequent subsections.

Different arc resistances are sometimes considered for phase distance protection as compared to ground distance protection since the factors which influence arc resistance potentially also vary for such faults.

The *Ratio RF(Ph-Ph) / RF(Ph-E)* parameter in the *Arc resistance* field can be used to capture this.

Circular impedance characteristics are usually specified in terms of an impedance magnitude and an angle. The impedance for each zone is usually specified by way of equations defined in the setting rules. The calculated impedance for each zone is complex and will therefore have a magnitude and an angle. The magnitude of the circular characteristic is usually specifically determined for each zone. However, the angle is not always zone specific. Using The *Angle calculation for circular characteristics* field the user can specify if this is the case. If the *Zone 1 impedance angle* option is selected then the calculated impedance angle for Zone 1 is used for all zones. If the *calculated impedance angle* option is selected then each zone will use the specific impedance angle calculated for that zone.

When the tool carries out the topological search of the network in the vicinity of the considered relays it is possible that the search can encounter multiple parallel paths each of which satisfy the reach equation

criteria. For example there could be identical parallel lines which meet the criteria for  $X_{minBC}$  (see Section 33.13.5.3). The *Aggregate parallel branches in search direction* setting controls how those parallel paths are handled. In cases where the option is checked, the parallel lines will be aggregated and an equivalent impedance selected for the parameter in question. In cases where the option is unchecked the impedances will not be aggregated and the impedance of a single line shall be used.

### 33.13.5.1 Pre-defined coordination rules

There are a number of typical setting rule philosophies which are very widely used. Even in cases where these typical rules aren't adhered to exactly, the actual rules which are used often use the basic structure of one of these typical rules as a basis. As such, these typical setting rules have been pre-programmed in the tool for convenience and can be directly used or subsequently customised according to user need. This section provides some background on the predefined coordination rules.

If the *Pre-defined coordination rules* button is pressed then the dialog as shown in figure 33.13.5 is displayed.

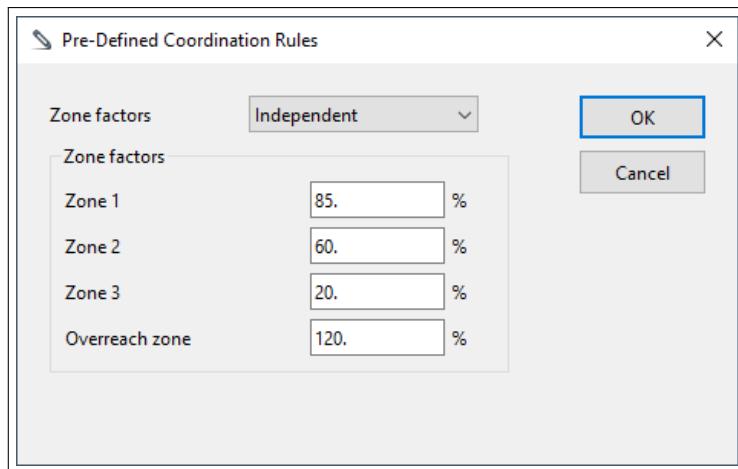


Figure 33.13.5: Pre-Defined Coordination Rules dialog

There are three pre-defined groups of setting rules which can be selected using the *Zone factors* drop down menu:

1. Independent
2. Cumulative
3. Referred to line 1

Depending on which option is selected a different set of Zone factor parameters are available for configuration in the *Zone factors* field.

Once this dialog is configured as required and the OK button pressed, a set of equations corresponding with the selected setting rules is populated in the tabs corresponding with each zone on the Distance Protection page of the main dialog. The equations are self-evident on inspection of the tabs, but an outline of the equations of the three predefined setting rules is provided here for reference.

In the equations listed below  $ZFac_1$ ,  $ZFac_2$ ,  $ZFac_3$ ,  $ZFac_{1b}$  and  $ZFac_{all}$  are the zone factors specified in the dialog illustrated in Figure 33.13.5. Reach equations are defined on a per zone basis with zone 1b representing the overreach zone. Section 33.13.5.3 provides definitions of the variables not defined above. Note that all impedances are complex.

***Independent method***

The zone reactive reaches are determined as follows:

$$X_1 = X_{minAB} \times ZFac_1 \quad (33.20)$$

$$X_2 = X_{maxAB} + X_{minBC} \times ZFac_2 \quad (33.21)$$

$$X_3 = X_{maxAB} + X_{minBC} + X_{minCD} \times ZFac_3 \quad (33.22)$$

$$X_{1b} = X_{minAB} \times ZFac_{1b} \quad (33.23)$$

The zone resistive reaches are determined as follows:

$$R_1 = R_{minAB} \times ZFac_1 + RF \quad (33.24)$$

$$R_2 = R_{maxAB} + R_{minBC} \times ZFac_2 + RF \quad (33.25)$$

$$R_3 = R_{maxAB} + R_{minBC} + R_{minCD} \times ZFac_3 + RF \quad (33.26)$$

$$R_{1b} = R_{minAB} \times ZFac_{1b} + RF \quad (33.27)$$

The zone impedance reaches are determined as follows:

$$Z_1 = Z_{minAB} \times ZFac_1 \quad (33.28)$$

$$Z_2 = Z_{maxAB} + Z_{minBC} \times ZFac_2 \quad (33.29)$$

$$Z_3 = Z_{maxAB} + Z_{minBC} + Z_{minCD} \times ZFac_3 \quad (33.30)$$

$$Z_{1b} = Z_{minAB} \times ZFac_{1b} \quad (33.31)$$

If the primarily protected line consists of parallel lines the equations above for zone 2 and zone 3 are modified as follows:

$$X_2 = X_{maxAB} + X_{minBC} \times K_{par} \times ZFac_2 \quad (33.32)$$

$$R_2 = R_{maxAB} + R_{minBC} \times K_{par} \times ZFac_2 + RF \quad (33.33)$$

$$Z_2 = Z_{maxAB} + Z_{minBC} \times K_{par} \times ZFac_2 \quad (33.34)$$

$$X_3 = X_{maxAB} + X_{minBC} \times ZFac_3 \quad (33.35)$$

$$R_3 = R_{maxAB} + R_{minBC} \times ZFac_3 + RF \quad (33.36)$$

$$Z_3 = Z_{maxAB} + Z_{minBC} \times ZFac_3 \quad (33.37)$$

***Cumulative method***

The zone reactive reaches are determined as follows:

$$X_1 = ZFac_{all} \times X_{minAB} \quad (33.38)$$

$$X_2 = ZFac_{all} \times (X_{maxAB} + ZFac_{all} \times X_{minBC}) \quad (33.39)$$

$$X_3 = ZFac_{all} \times (X_{maxAB} + ZFac_{all} \times (X_{minBC} + ZFac_{all} \times X_{minCD})) \quad (33.40)$$

$$X_{1b} = ZFac_{1b} \times X_{minAB} \quad (33.41)$$

The zone resistive reaches are determined as follows:

$$R_1 = ZFac_{all} \times R_{minAB} + RF \quad (33.42)$$

$$R_2 = ZFac_{all} \times (R_{maxAB} + ZFac_{all} \times R_{minBC}) + RF \quad (33.43)$$

$$R_3 = ZFac_{all} \times (R_{maxAB} + ZFac_{all} \times (R_{minBC} + ZFac_{all} \times R_{minCD})) + RF \quad (33.44)$$

$$R_{1b} = ZFac_{1b} \times R_{minAB} + RF \quad (33.45)$$

The zone impedance reaches are determined as follows:

$$Z_1 = ZFac_{all} \times Z_{minAB} \quad (33.46)$$

$$Z_2 = ZFac_{all} \times (Z_{maxAB} + ZFac_{all} \times Z_{minBC}) \quad (33.47)$$

$$Z_3 = ZFac_{all} \times (Z_{maxAB} + ZFac_{all} \times (Z_{minBC} + ZFac_{all} \times Z_{minCD})) \quad (33.48)$$

$$Z_{1b} = ZFac_{1b} \times Z_{minAB} \quad (33.49)$$

If the primarily protected line consists of parallel lines the equations above for zone 2 and zone 3 are modified as follows:

$$X_2 = ZFac_{all} * X_{maxAB} + K_{par} * X_{minBC} \quad (33.50)$$

$$R_2 = ZFac_{all} * R_{maxAB} + K_{par} * R_{minBC} + RF \quad (33.51)$$

$$Z_2 = ZFac_{all} * Z_{maxAB} + K_{par} * Z_{minBC} \quad (33.52)$$

$$X_3 = 1.10 * (X_{maxAB} + X_{minBC}) \quad (33.53)$$

$$R_3 = 1.10 * (R_{maxAB} + R_{minBC}) + RF \quad (33.54)$$

$$Z_3 = 1.10 * (Z_{maxAB} + Z_{minBC}) \quad (33.55)$$

### **Referred to line 1 method**

In this method, all the calculated zone impedances are based on the impedance of the first protected line and the entered zone factors. The zone impedance settings are calculated as follows:

In general for this method, the zone factors entered should be ascending. *PowerFactory* will print a warning to the output window when it detects this is not the case.

The zone reactive reaches are determined as follows:

$$X_1 = ZFac_1 \times X_{minAB} \quad (33.56)$$

$$X_2 = ZFac_2 \times X_{maxAB} \quad (33.57)$$

$$X_3 = ZFac_3 \times X_{maxAB} \quad (33.58)$$

$$X_{1b} = ZFac_{1b} \times X_{minAB} \quad (33.59)$$

The zone resistive reaches are determined as follows:

$$R_1 = ZFac_1 \times R_{minAB} + RF \quad (33.60)$$

$$R_2 = ZFac_2 \times R_{maxAB} + RF \quad (33.61)$$

$$R_3 = ZFac_3 \times R_{maxAB} + RF \quad (33.62)$$

$$R_{1b} = ZFac_{1b} \times R_{minAB} + RF \quad (33.63)$$

The zone impedance reaches are determined as follows:

$$Z_1 = ZFac_1 \times Z_{minAB} \quad (33.64)$$

$$Z_2 = ZFac_2 \times Z_{maxAB} \quad (33.65)$$

$$Z_3 = ZFac_3 \times Z_{maxAB} \quad (33.66)$$

$$Z_{1b} = ZFac_{1b} \times Z_{minAB} \quad (33.67)$$

If the primarily protected line consists of parallel lines the equations above for zone 2 and zone 3 are modified as follows:

$$X_2 = ZFac_1 * X_{maxAB} + K_{par} * X_{minBC} \quad (33.68)$$

$$R_2 = ZFac_1 * R_{maxAB} + K_{par} * R_{minBC} + RF \quad (33.69)$$

$$Z_2 = ZFac_1 * Z_{maxAB} + K_{par} * Z_{minBC} \quad (33.70)$$

$$X_3 = 1.10 * (X_{maxAB} + X_{minBC}) \quad (33.71)$$

$$R_3 = 1.10 * (R_{maxAB} + R_{minBC}) + RF \quad (33.72)$$

$$Z_3 = 1.10 * (Z_{maxAB} + Z_{minBC}) \quad (33.73)$$

### 33.13.5.2 Definition of reach equations

The core task when configuring the protection coordination assistant to replicate the required setting rules is to define the reach equations. The reach equations are defined on the *Zone 1*, *Zone 2*, *Zone 3* and *Zone 4* tabs of the *Distance protection* page of the protection coordination assistant command. The Zone 1 tab is illustrated in Figure 33.13.6 and the Zone 2, Zone 3 and Zone 4 tabs are almost identical to the Zone 1 tab. As mentioned in the previous section predefined coordination rules can be selected on the General tab and if one is selected the equations will be populated on the relevant zone tabs. These equations can then be modified according to user need.

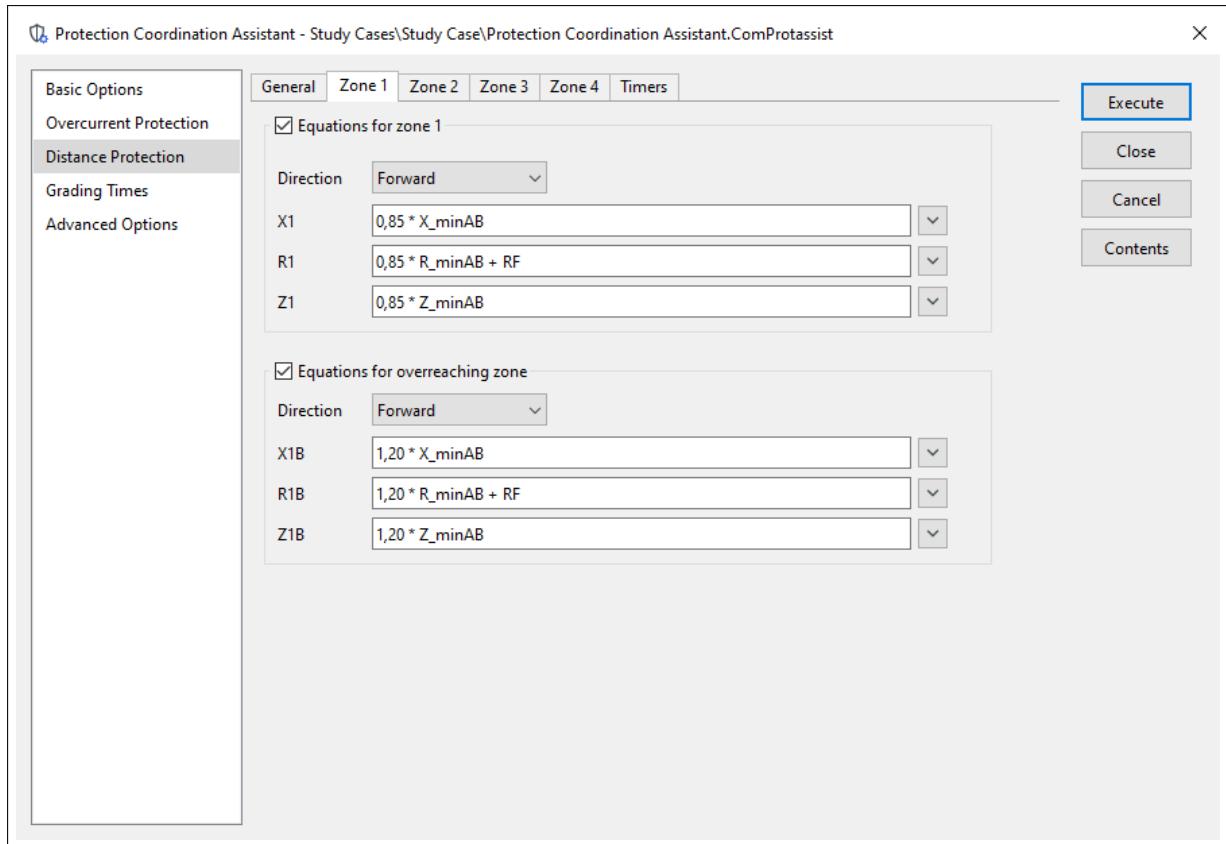


Figure 33.13.6: Zone 1 tab of the distance protection page of the protection coordination assistant command

It can be seen that on the zone 1 tab resistance, reactance, and impedance reach equations can be entered as well as equivalent overreach zone parameters. Both zones can be specified to be forward looking, reverse looking or non directional using *Direction* drop down fields. The calculation of the reaches can also be disabled if necessary by selecting the corresponding equations check box at the start of the relevant field.

The equations are entered using standard mathematical operators (+,-, /, \*, max() and min()) as well as some special variables. These variables, also known as keywords are described in Section 33.13.5.3.

When using the max() and min() operators, a maximum of two expressions can be compared with each expression separated by a ";" character.

An example of typical usage for zone 2 is given below:

```
X2 = max( X_maxAB+0,6*X_minBC ; 1,2*X_maxAB )
R2 = max( R_maxAB+0,6*R_minBC ; 1,2*R_maxAB ) + RF
Z2 = max( Z_maxAB+0,6*Z_minBC ; 1,2*Z_maxAB )
```

For the zone 2, 3 and 4 tabs, the overreach zone equations are replaced with equations to be considered if there are parallel lines in zone 1 but otherwise the handling is the same.

Note that different relay characteristics require different parameters to be specified. For example circular characteristics are usually defined using an impedance, whilst polygonal relay characteristics require a reactance and resistance setting. The coordination assistant will calculate all quantities providing it is provided with corresponding reach setting rules and will report all results.

### 33.13.5.3 Variables (Keywords) used in the definition of the reach equations

When defining setting rules a specific set of variables also known as Keywords must be used to define the rules. These variables are described in this section. In general the variables which are available for each equation are accessible by clicking the button at the end of the equation box.

Consider a portion of network as illustrated in Figure 33.13.7 with a distance relay located at one end as shown which has a CT located in the vicinity of Terminal A oriented towards Terminal B.

---

**Note:** All CT's have an orientation specified in the CT element (*StaCt*) as either towards the associated branch or towards the associated busbar. The orientation of a CT is used by *PowerFactory*'s topological search algorithm to determine which relays are expected to coordinate with one another. Distance relays are generally directional and where the CT's of distance relays which are oriented in the same direction, the distance relays are expected to coordinate

---

From the perspective of the relay, Terminal A is the busbar at which the relay is located. Terminal B could be any busbar located at the end of a line in the direction in which the relay's CT is oriented. In the illustrated case there are two Terminal B's since there is a tee off approximately half way along the first line. Terminal C could be any terminal one line away from any Terminal B, likewise a Terminal D could be any terminal one line away from a Terminal C and a Terminal E any terminal one line away from a Terminal D. In this network there are certainly multiple Terminal C's, D's, and E's but in this illustration only one of each is explicitly represented. Note that a line can be composed of multiple sections. At the end of a line section *PowerFactory* will only consider the line to be terminated if there is a current transformer located at the end of the section which is orientated in the same direction as the original relay's CT or if there are no further lines to traverse. For the illustrated case a CT located in the vicinity of the upper Terminal B oriented towards a Terminal C would terminate the line between Terminal A and Terminal B. Whilst for the tee off circuit, there are no further lines to traverse at Terminal B, so the circuit is terminated here.

Between each of these classified terminals there will be lines of differing impedances. Some will be long and will likely have higher impedances some will be short and will likely have lower impedances. From a coordination point of view the highest and lowest impedance connected between any two terminals is considered to be of interest.

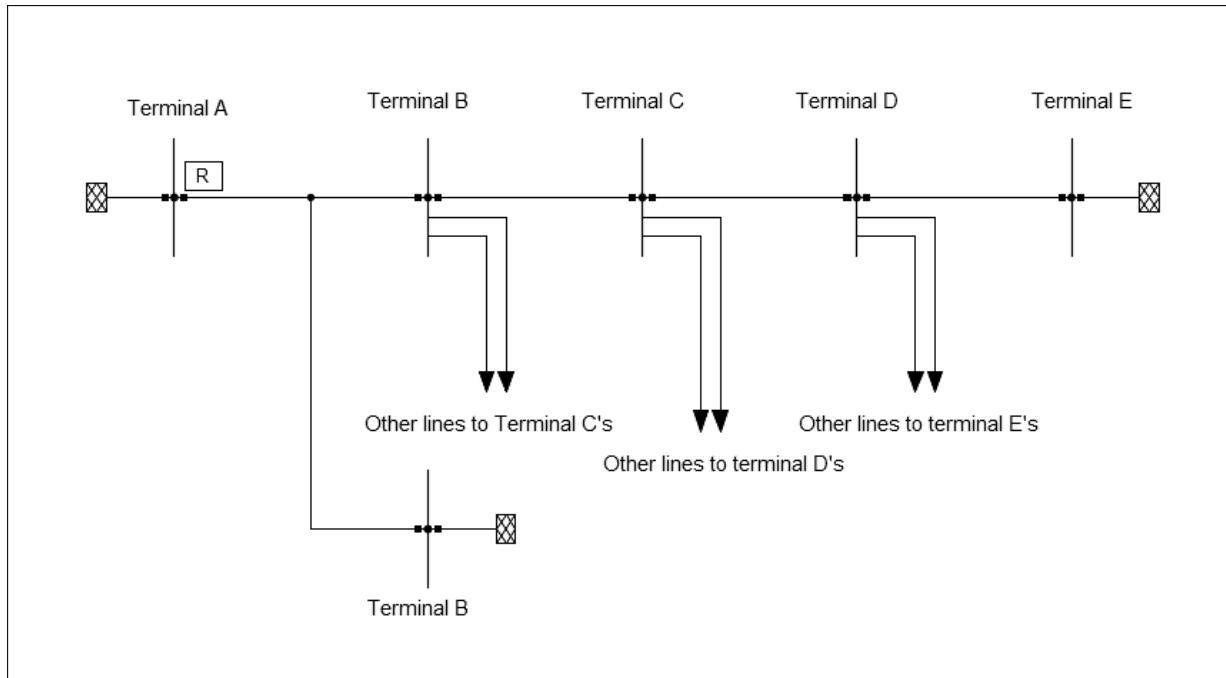


Figure 33.13.7: Classification of terminals in a portion of a network

The classification of the terminals as A,B,C,D or E is important for the definition of setting rules because these classifications are used in the variable names which can be used in the equations defining the setting rules. Each available variable name is listed below:

**Variables specifiable in Zone 1 equations onwards**

X\_minAB: The reactance of the minimum impedance line located between Terminal A and any Terminal B.

X\_maxAB: The reactance of the maximum impedance line located between Terminal A and any Terminal B.

R\_minAB: The resistance of the minimum impedance line located between Terminal A and any Terminal B.

R\_maxAB: The resistance of the maximum impedance line located between Terminal A and any Terminal B.

Z\_minAB: The impedance of the minimum impedance line located between Terminal A and any Terminal B.

Z\_maxAB: The impedance of the maximum impedance line located between Terminal A and any Terminal B.

RF: The arc resistance, specifiable for resistive reach expressions only and adjusted according to the voltage level at which the relay is installed and according to whether phase to phase or phase to ground settings are being specified.

RLoad: Specifiable for resistive reach expressions only, this parameter is calculated based on the ratings of the limiting component in the coordination path in terms of nominal current rating. The nominal voltage and current of this limiting component is used to calculate a prospective load resistance using the equation below:

$$R_{Ld} = \left( \frac{U_{nom}}{\sqrt{3} \times I_{nom}} \right) \quad (33.74)$$

The searching algorithm considers only lines as limiting component of the primary plant. For different zones RLoad could take different values since different zones could cover paths containing different items of plant.

***Additional variables specifiable in Zone 2 equations onwards:***

X\_minBC: The reactance of the minimum impedance line located between Terminal B and any adjacent Terminal C, where Terminal B is the same Terminal that was selected for Terminal B in the section AB part of the equation.

X\_maxBC: The reactance of the maximum impedance line located between Terminal B and any adjacent Terminal C, where Terminal B is the same Terminal that was selected for Terminal B in the section AB part of the equation.

R\_minBC: The resistance of the minimum impedance line located between Terminal B and any adjacent Terminal C, where Terminal B is the same Terminal that was selected for Terminal B in the section AB part of the equation.

R\_maxBC: The resistance of the maximum impedance line located between Terminal B and any adjacent Terminal C, where Terminal B is the same Terminal that was selected for Terminal B in the section AB part of the equation.

Z\_minBC: The impedance of the minimum impedance line located between Terminal B and any adjacent Terminal C, where Terminal B is the same Terminal that was selected for Terminal B in the section AB part of the equation.

Z\_maxBC: The impedance of the maximum impedance line located between Terminal B and any adjacent Terminal C, where Terminal B is the same Terminal that was selected for Terminal B in the section AB part of the equation.

K\_par: This variable is only available for specification in the special equations to be used where the primarily protected lines covered by zone 1 are parallel lines. It is used to represent the impedance reduction factor which can be calculated from the ratio of the impedances of the parallel lines making up the circuit.

***Additional variables specifiable in Zone 3 equations onwards:***

X\_minCD: The reactance of the minimum impedance line located between Terminal C and any adjacent Terminal D, where Terminal C is the same Terminal that was selected for Terminal C in the section BC part of the equation.

X\_maxCD: The reactance of the maximum impedance line located between Terminal C and any adjacent Terminal D, where Terminal C is the same Terminal that was selected for Terminal C in the section BC part of the equation.

R\_minCD: The resistance of the minimum impedance line located between Terminal C and any adjacent Terminal D, where Terminal C is the same Terminal that was selected for Terminal C in the section BC part of the equation.

R\_maxCD: The resistance of the maximum impedance line located between Terminal C and any adjacent Terminal D, where Terminal C is the same Terminal that was selected for Terminal C in the section BC part of the equation.

Z\_minCD: The impedance of the minimum impedance line located between Terminal C and any adjacent Terminal D, where Terminal C is the same Terminal that was selected for Terminal C in the section BC part of the equation..

$Z_{\text{maxCD}}$ : The impedance of the maximum impedance line located between Terminal C and any adjacent Terminal D, where Terminal C is the same Terminal that was selected for Terminal C in the section BC part of the equation.

**Additional variables specifiable in Zone 4 equations:**

$X_{\text{minDE}}$ : The reactance of the minimum impedance line located between Terminal D and any adjacent Terminal E, where Terminal D is the same Terminal that was selected for Terminal D in the section CD part of the equation.

$X_{\text{maxDE}}$ : The reactance of the maximum impedance line located between Terminal D and any adjacent Terminal E, where Terminal D is the same Terminal that was selected for Terminal D in the section CD part of the equation.

$R_{\text{minDE}}$ : The resistance of the minimum impedance line located between Terminal D and any adjacent Terminal E, where Terminal D is the same Terminal that was selected for Terminal D in the section CD part of the equation.

$R_{\text{maxDE}}$ : The resistance of the maximum impedance line located between Terminal D and any adjacent Terminal E, where Terminal D is the same Terminal that was selected for Terminal D in the section CD part of the equation.

$Z_{\text{minDE}}$ : The impedance of the minimum impedance line located between Terminal D and any adjacent Terminal E, where Terminal D is the same Terminal that was selected for Terminal D in the section CD part of the equation.

$Z_{\text{maxDE}}$ : The impedance of the maximum impedance line located between Terminal D and any adjacent Terminal E, where Terminal D is the same Terminal that was selected for Terminal D in the section CD part of the equation.

From the above it should be clear that reach equations must always be built up following a path composed of the impedances of adjacent lines. It would not for example be possible to create a reach equation composed of the impedances of the combination of elements highlighted in Figure 33.13.8.

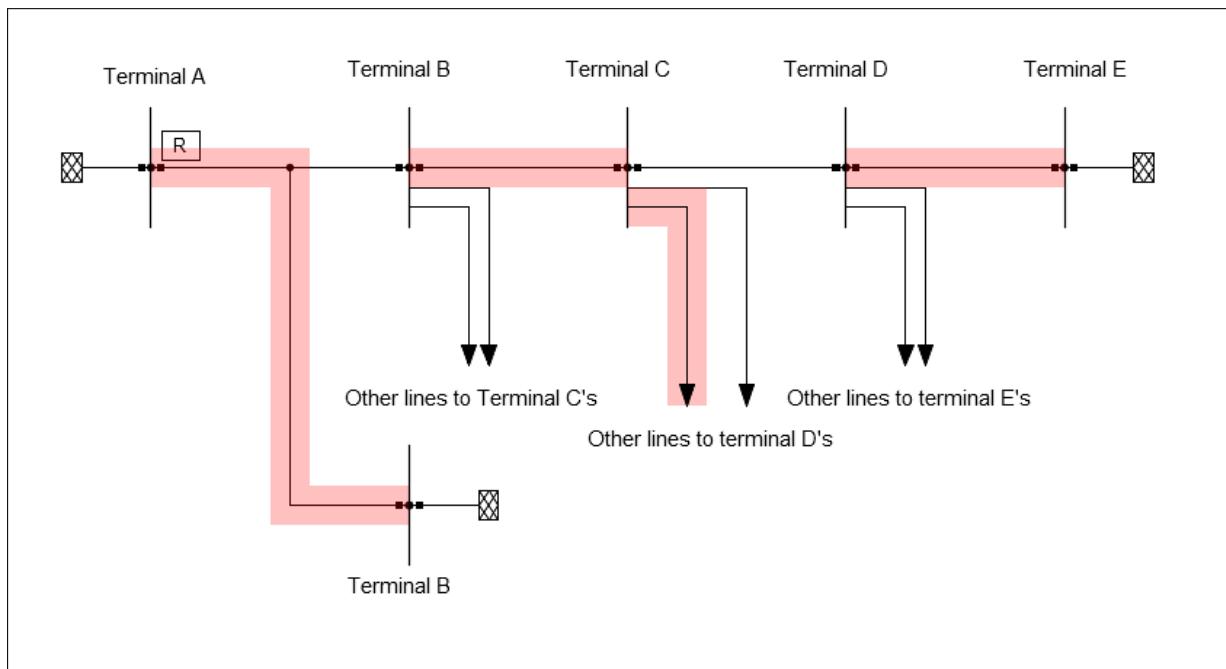


Figure 33.13.8: Example of an impossible combination of lines for a reach equation

### ***Additional variables specifiable in all zones***

In addition to the variables documented above there is one additional set of variables available for use in reach equations.

In some cases it might be desirable to specify reach equations in the forward direction based on the impedances of lines extending behind the relaying location or reach equations in reverse direction based on the impedance of lines extending in the forward direction. One example is in the implementation of a reverse blocking scheme. For such cases the reverse impedances can be specified in the equations simply by reversing the bus letters. So for example, the lowest impedance line connected at Terminal A in the reverse direction from a forward looking zone would be specified as  $Z_{minBA}$ .

These variables are not accessible by clicking the button at the end of the equation box and must be manually entered. This is simply because the variables are less widely used and would otherwise clutter and potentially introduce confusion to the dialog.

Note this inversion in the variable name is not required for reverse looking zones derived from impedances in the reverse direction. For such zones the normal variables are used but of course those variables will be derived from a path extending in the reverse direction.

#### **33.13.5.4 Definition of timer settings**

The final tab on the *Distance Protection* page of the protection coordination assistant command is the *Timers* tab. This is illustrated in Figure 33.13.9

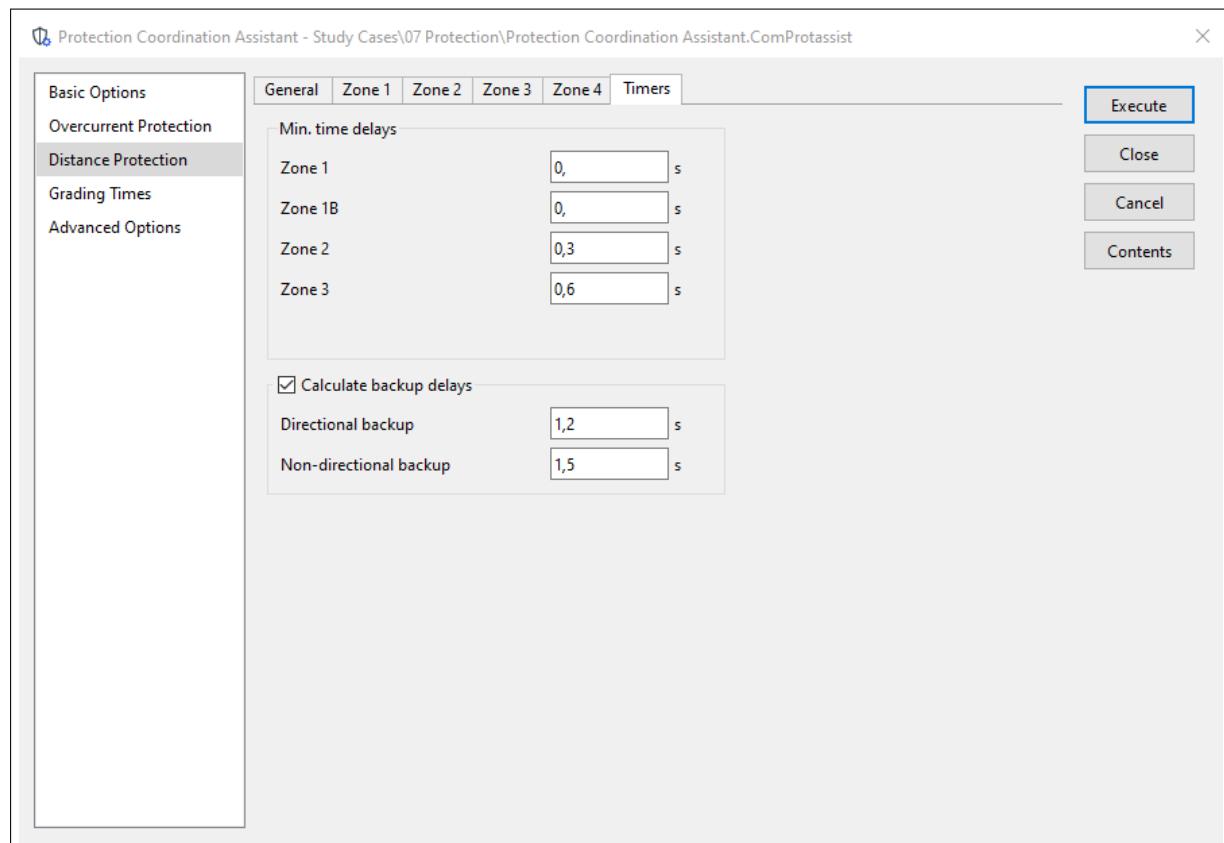


Figure 33.13.9: Timers tab of the distance protection page of the protection coordination assistant command

The *Min. time delays* field contains timer settings for each zone which can be adjusted according to

need.

For some relays directional elements and starting elements provide backup protection for distance elements. The *Calculate backup delays* check box can be used to specify whether the tool will additionally log these settings in the result file with the values specified according to the user requirement.

### 33.13.6 Grading Times

#### Grading Time

Different times can be specified depending on voltage level and protection device. Typically for the distance coordination, only the *Relay-to-Relay* grading time for voltage levels above 1kV needs to be considered. For the overcurrent coordination it is important if relays and fuses are graded in LV or HV networks and if relays are coordinated with itself or with fuses and viceversa. Low voltage circuit breakers (LVCBs) are considered as relays. *Fuse-to-Fuse* grading depends on the nominal current. The grading time is always applied from the downstream protection device. For example if two protection devices are located on the HV and LV side of a transformer, the HV protection device is graded to the LV protection device based on the LV grading time.

#### Distance- to Overcurrent-protection

- *No grading*: The distance coordination algorithm ignores overcurrent devices.
- *Fastest / Slowest definite time characteristic*: The distance coordination algorithm detects overcurrent devices in the search path and grades the zone according to the selected definite time characteristic and the grading time. This option could be useful in distribution networks where a distance zone covers multiple tee off transformers with overcurrent protection. To ensure selectivity between the overcurrent device and the upstream distance device the coordination assistant offers this option.

#### Overcurrent- to Distance-protection

- *No grading*: The overcurrent coordination algorithm ignores distance devices.
- *Fastest / Slowest zone*: The overcurrent coordination algorithm detects distance devices in the search path and grades according to the specified zone timer.

### 33.13.7 Advanced Options

The advanced options page is used to tune the behaviour of the topological search algorithm used to identify the relevant impedances of the network extending from each relay location.

The *Stop criteria for topological search* option can be used to limit the extent of the topological search in order to improve performance of the tool. This is usually only relevant in large networks with many relays where the topological search may continue for some time if it does not encounter a current transformer from another relay or a fuse (provided the *Consider fuses for coordination* option is selected) indicating the termination of a protected line.

### 33.13.8 Prerequisites for using the protection coordination tool

Before starting the protection coordination assistant, ensure the following:

1. A network model of the area has been completed in *PowerFactory*.

2. The relays to be coordinated should be added to the model and instrument transformers, configured with the appropriate ratios, assigned.

### 33.13.9 How to run the protection coordination calculation

To run the protection coordination follow these steps:

1. Click the  icon on the main toolbar.
2. Select *Protection and Arc-Flash Analysis*.
3. Click the  icon. A dialog for the *Protection Coordination Assistant* will appear.
4. Click the  icon and choose the coordination or protection devices to coordinate.
5. Optional: Adjust options for the coordination on the relevant pages.
6. Click **OK** to run the coordination.
7. To analyse results of the coordination see Section [33.13.10](#).

### 33.13.10 How to output results from the protection coordination assistant

This section explains how the results from the protection coordination assistant can be analysed.

The graphical methods of analysis using plots and the tabular method using the built-in report are discussed. Furthermore, there is an option to write the coordination results back to the protection devices via the created tabular report.

Results are generated following execution of the protection coordination results command. Usually this is done automatically on execution of the protection coordination assistant command thanks to a link between the two objects specified in the protection coordination assistant command (as mentioned in Section [33.13.3](#)). However, if this option is not selected the protection coordination results command can be executed independently using the procedure below:

To output results from the protection coordination assistant follow these steps:

1. Execute the protection coordination tool. See Section [33.13.9](#) for instructions how to do this.
2. Click  the icon from the protection toolbar. A dialog for choosing the output options will appear. In the field *Result selection* the result file that the output is based on can be selected. If the user wishes to output results from a different calculation, perhaps completed in another study case, then this is where those results are selected.
3. Check the boxes for the reports that you would like *PowerFactory* to produce. The types of reports are:
  - *Overcurrent protection settings*: This option produces a tabular report for the overcurrent coordination results if the option *Create report* is activated. Further information about the overcurrent report is presented in Section [33.13.10.1](#).
  - *Distance protection settings*: This option produces a tabular report for the distance coordination results if the option *Create report* is activated. Further information about the distance report is presented in Section [33.13.10.2](#). The option *Create Time-Distance Plot* combined with a valid *Path Selection* creates a time-distance diagram with the coordination results for the selected path. In order to select the path click on the icon. General information about this diagram is presented in Section [33.8](#).

### 33.13.10.1 Tabular report of the overcurrent protection coordination results

The report for overcurrent protection settings shows the coordination results for each overcurrent device.

#### Report header

- *Show devices in:* If the devices are part of a grouping element or substation this filter is enabled and the devices can be filtered according to the selected element.
- *Specific Device:* This drop-down menu allows filtering of the report according to a specific device.
- *Transfer settings to all devices:* By clicking on this button, the calculated settings of the devices shown in the report are written to the relevant overcurrent stages of the protection devices. If a filter is enabled only the filtered devices are considered. Please note, that the existing settings are overwritten. The assistant prints the overcurrent devices for which the settings are transferred into the output window.

#### Interactive options

The cells of the report are interactive, meaning that the presented objects are linked to network objects and that the basis of calculated results can be verified by right-clicking on the cell and selecting the appropriate option. The following options are available for current and time delay results:

- *Mark considered network elements in open graphic:* For calculated current results the short circuit location is marked in the open graphic. For calculated time delay results the coordinated devices are marked in the open graphic.
- *Show considered network elements:* For calculated current setting results this option shows the short circuit location in a browser window. For calculated time delay results the coordinated devices are shown in a browser window.

#### Column description

- *Protection device:* lists the considered protection devices. Right-click and select *Transfer settings to device* to write only the settings of one protection device back. By double clicking on the cell the object dialog opens.
- *Max. current:* shows the results of the maximum current equation. Right-click and select one of the previously described options to verify the short circuit location.
- *Stage:* lists the stages for which results are calculated.
- *Direction:* shows the directional setting of each protection device.
- *Characteristic:* lists the selected characteristic for each overload stage. The stages *Short-circuit* and *Instantaneous* will be listed as *Definite time*.
- *Current:* shows the current setting for each stage according to the specified equation. Right-click and select one of the previously described options to verify the short circuit location.
- *Time delay:* shows the selected time delay for each stage. Right-click and select one of the previously described options to verify the coordinated devices.

#### Cell messages

There might be network configurations or other problems where the assistant cannot calculate the settings. In such a case the report will show three dashes "---" in the relevant cells. By hovering the mouse pointer over such a cell entry, a short message is displayed describing why the calculation failed. The following list describes the messages and how the failure could be solved:

- *Equation not provided:* If none of the stage equations or the fuse equation is provided this message appears. Please check the equations for the relevant equipment type.
- *Calculated value exceeds max. current:* The maximum current equation is required for the time grading. If this message appears, one stage equation leads to a higher current value than that resulting from the maximum current equation, which is forbidden. Please check the maximum current equation of the relevant equipment type.
- *Nominal current could not be determined:* In this case one equation uses the nominal current variable and the protected element cannot provide this value. Please verify that the protected element has a nominal current available.
- *Topological path too short:* This message indicates that the keywords or variables selected in the stage equation reaches beyond the end of the network. Please check the equation of the relevant stage.
- *Orientation of the device is ambiguous:* This is typically the case when a bi-directional power flow is detected and the directional setting in the relay element is not activated. Please activate the option *Directional* in the relay element.

---

**Note:** If you recalculate the protection coordination results, this report is not automatically updated - you must use the option *Refresh* icon to update the report.

---

### 33.13.10.2 Tabular report of the distance protection coordination results

For each relay, the following results are produced:

**Protection device.** This column shows the name of each protection device for which settings have been calculated. By right-clicking relays in this column it is possible to navigate to the relay objects in graphics or in the database and it is also possible to transfer the calculated settings to the specific device.

**Zone.** This column shows the zones for every protection device for which settings have been calculated. By right-clicking zones in this column it is possible to transfer the calculated settings for the specific zone only to the device.

**Direction.** This column indicates for each zone whether it is a forward, reverse or non directional zone.

**Polygonal Reactance.** This column shows the primary Ohm reactance for each zone. By right-clicking on the figure and choosing Show traversed element or mark traversed elements in open graphic it is possible to better visualise the reach of the setting.

**Polygonal Resistance Ph-Ph.** This column shows the primary Ohm phase to phase resistance for each zone. By right-clicking on the figure and choosing Show traversed element or mark traversed elements in open graphic it is possible to better visualise the reach of the setting.

**Polygonal Resistance Ph-E.** This column shows the primary Ohm phase to earth resistance for each zone. By right-clicking on the figure and choosing Show traversed element or mark traversed elements in open graphic it is possible to better visualise the reach of the setting.

**Polygonal time delay.** This column shows the time delay for each polygonal zone. By right-clicking on the figure and choosing Show traversed element or mark traversed elements in open graphic it is possible to visualise the other relay elements with which this time setting has been coordinated.

**Circular Impedance.** This column shows the primary Ohm impedance for each zone. By right-clicking on the figure and choosing Show traversed element or mark traversed elements in open graphic it is possible to better visualise the reach of the setting.

**Circular Angle.** This column shows the primary Ohm angle of the impedance for the relay. Depending on the setting of the *Angle calculation for circular characteristics* setting in the Protection coordination assistant, the angle may be calculated each zone or for zone 1 only.

**Circular Time delay.** This column shows the time delay for each circular zone. By right-clicking on the figure and choosing Show traversed element or mark traversed elements in open graphic it is possible to visualise the other relay elements with which this time setting has been coordinated.

The report can be filtered using the drop down lists at the top:

**Show devices in.** Use this filter to select a grouping object class.

**Selected.** Use this filter to select a specific grouping within the grouping class.

**Show Characteristic.** Use this filter to show circular or polygonal characteristics in isolation.

As mentioned above the calculated settings can be transferred to the devices individually by interacting with the cells in the table. However, if after review of the settings it is determined that all settings should be transferred to the relay devices in the network model then this can be done by pressing the *Transfer settings to all devices* button. It is important to note that this process will overwrite existing settings in the model. If this is a problem it is recommended to create a Variation prior to applying the settings. Subsequently, it is easy to revert to the old settings by disabling the Variation. Refer to Section 17.2 for more information about *PowerFactory* Variations.

To output these results to Excel or to HTML click on the corresponding export icon in the upper right corner of the tabular report.

---

**Note:** If you recalculate the protection coordination results, this report is not automatically updated - you must use the option *Refresh* icon to update the report.

---

### 33.13.10.3 Time distance diagrams from the protection coordination

Enabling the option *Create Report* when outputting the coordination results as described in Section 33.13.10, automatically generates a time distance diagram showing the results from the previously completed protection coordination. One diagram will be produced for each path. An example time distance diagram for a coordination completed using the independent method is shown in Figure 33.13.10.

Note that the plot display can be configured by double-clicking the diagram. For further information about time distance diagrams refer to Section 33.8.

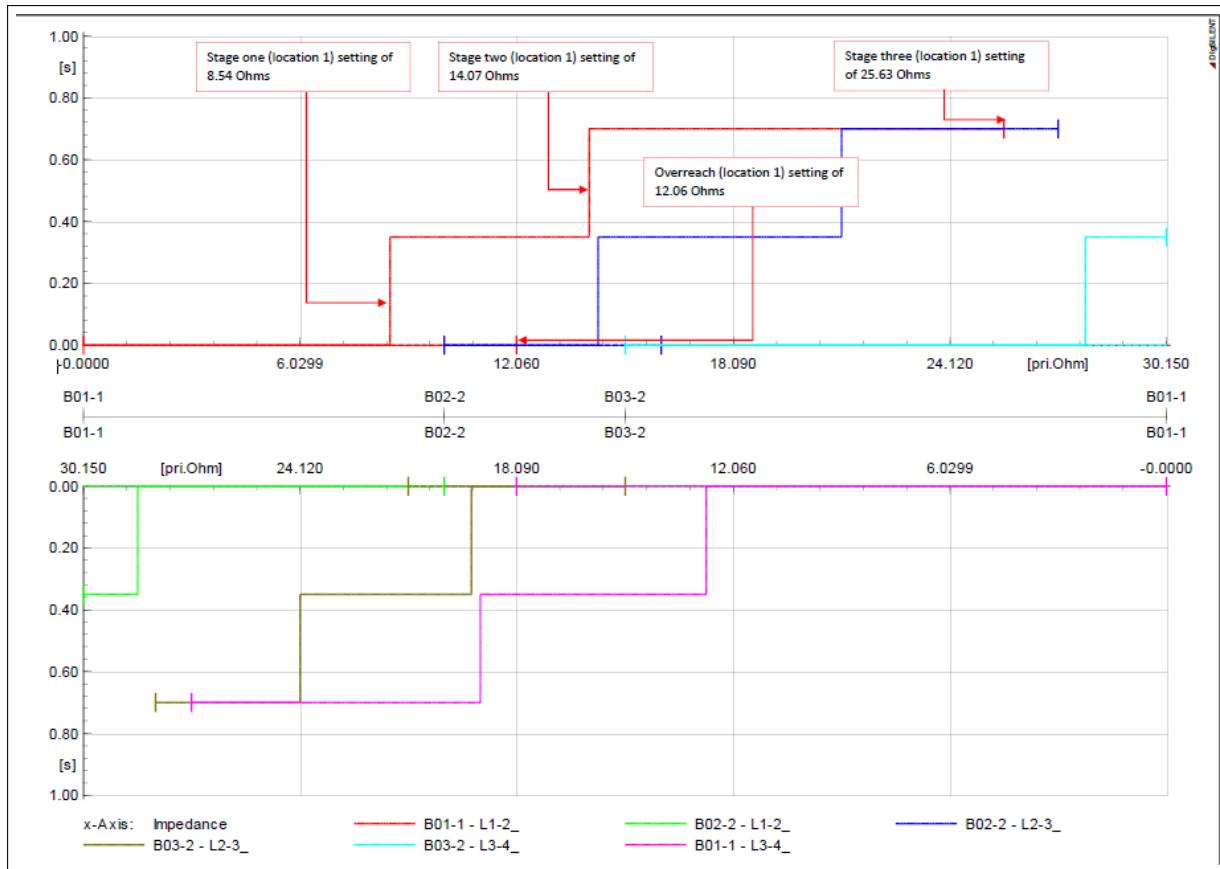


Figure 33.13.10: Time distance diagram showing the result from the protection coordination using the independent method on the network shown in Figure 33.13.1

## 33.14 Accessing results

After all protection devices have been configured and graded, it is often desirable to create reports for future reference. Aside from exporting the time-overcurrent, R-X or time-distance plots as graphical files (see Chapter 19: Reporting and Visualising Results, Section 19.8.6: Tools for Plots), there are several other methods described in subsections 33.14.2 and 33.14.3 to report the relay settings.

Subsection 33.14.1 describes how to access quickly to protection plots of relays.

### 33.14.1 Quick access to protection plots

In most of the protection projects, protection plots are indispensable. That is why some relays can appear in various different protection diagrams or why some projects can posses huge number of plots. To minimise possibility of confusion, there is an option that allows us to directly jump into all existing protection plots that contain the relay of interest.

To quickly access the protection plots for some specific relay, the user should select the option *Plots → Show in Existing Protection Plot* from the context menu. This option can be called from the Network Model Manager (see Figure 33.14.1) or directly out of the network graphic (see Figure 33.14.2).

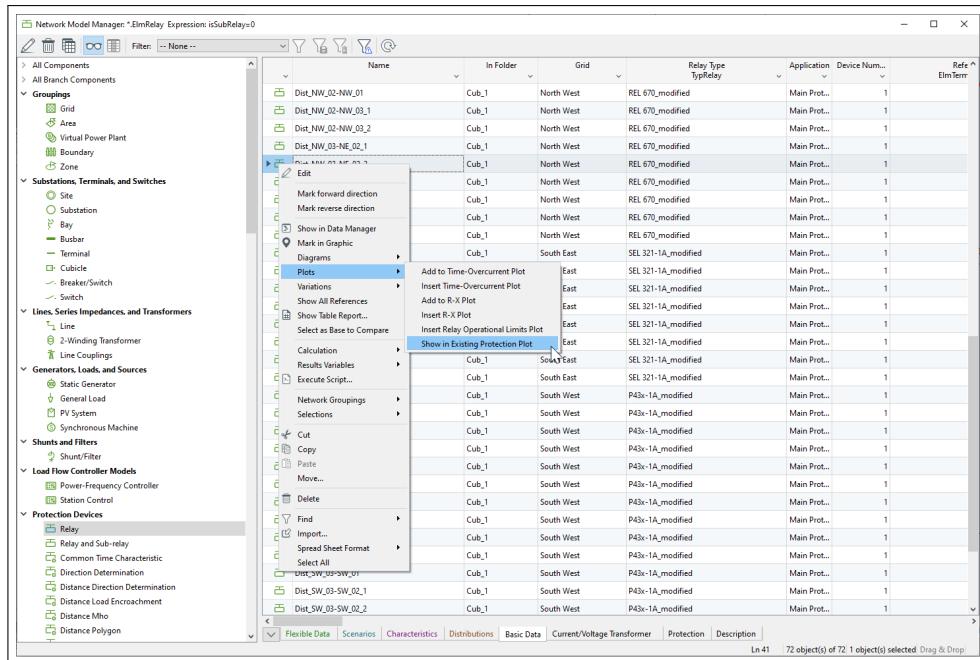


Figure 33.14.1: Access through Object Filter

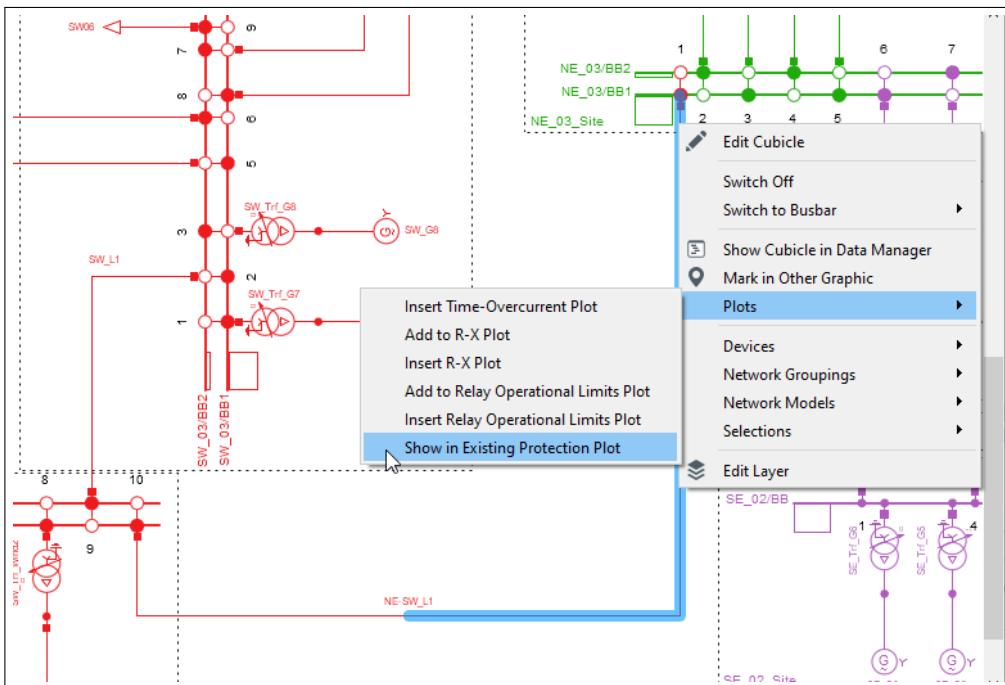


Figure 33.14.2: Access from the Grid

If the Relay appears in more than one diagram, after selection of *In existing Protection Plots*, the user can select which diagram he is interested in (see Figure 33.14.3).

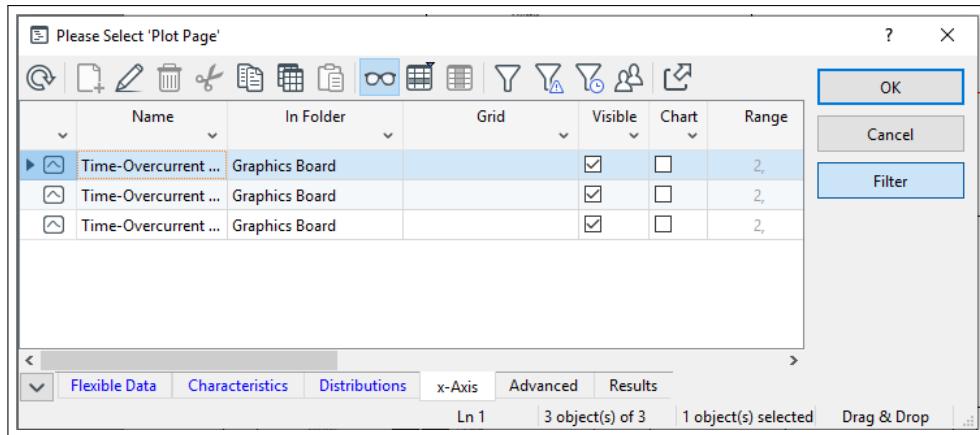


Figure 33.14.3: selection of the plot

### 33.14.2 Tabular protection setting report

A report command specifically for protection can be accessed by either clicking on the *Output of Protection Settings* icon on the *Protection* toolbar or alternatively via the “Output” entry in the main menu.

The Output of protection settings command dialog(*ComProtreport*) has three pages:

1. Basic Options
2. Common Options
3. Specific Options

#### 33.14.2.1 Basic Options

In this page the user chooses which equipment to generate reports for. First the user chooses general classes of equipment from the options below:

- Instrument Transformers
- Overcurrent Protection
- Distance Protection
- Voltage Protection
- Frequency Protection

any combination of the above options may be selected. Each option which is selected will result in the generation of a separate tabular report. I.e. if all five options are selected, five tabular reports will be generated.

In the lower section of the page the user can choose to consider all protection devices in the active grid or only a specific user defined subset. The following objects may be selected as a user defined subset: *SetSelect*, *SetFilt*, *ElmNet*, *ElmArea*, *ElmZone*, *ElmFeeder*, *ElmSubstat* and *ElmTrfstat*. Additionally a single protection device (*ElmRelay*, *RelFuse*) can also be selected.

### 33.14.2.2 Common Options

The decimal precision section can be used to define the number of decimal places to which results are given in the tabular reports. The precision for each unit can be defined individually.

The layout options section is used to configure the layout for each report. Depending on whether they are selected, “Device, Location and Branch” will be the first three columns of the report.

If the *show instrument transformers* option is selected, additional columns will be added to the overcurrent, distance, voltage and frequency protection reports showing details of the instrument transformers.

If the *Report settable blocks only* option is selected, blocks which have no user configurable settings will not be displayed in the report.

If the *Arrange stages vertically* option is selected, additional rows will be added to the report for each protection stage, rather than including additional stages as additional columns.

If the *Show ANSI code* option is selected, each stage column will include the relevant ANSI code as defined by IEEE (ANSI) C37-2.

### 33.14.2.3 Specific Options

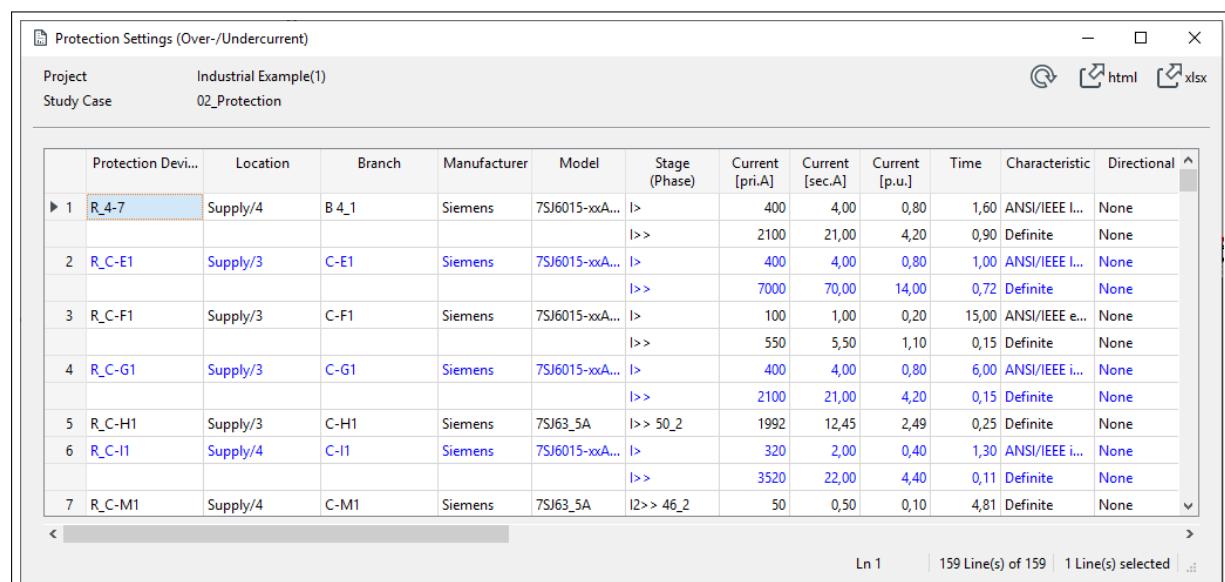
The Over-/Undercurrent and Over-/Undervoltage sections of this page can be used to define whether settings should be displayed in primary units, secondary units, or per unit. Any combination of the 3 options is possible.

This page is also used to limit the report for each type of protection to a specified number of phase and earth fault protection stages.

Once the *ComProtreport* dialog has been configured it can be executed.

### The Tabular Report

An example of a tabular report generated when the *ComProtreport* dialog is executed is illustrated in Figure 33.14.4:



The screenshot shows a software interface titled "Protection Settings (Over-/Undercurrent)". The window includes a header bar with project information ("Project: Industrial Example(1)", "Study Case: 02\_Protection") and file export options ("html", "xlsx"). Below the header is a table with the following data:

Protection Devi...	Location	Branch	Manufacturer	Model	Stage (Phase)	Current [pri.A]	Current [sec.A]	Current [p.u.]	Time	Characteristic	Directional
► 1 R_4-7	Supply/4	B 4_1	Siemens	7SJ6015-xxA...	I>	400	4,00	0,80	1,60	ANSI/IEEE I...	None
					I>>	2100	21,00	4,20	0,90	Definite	None
2 R_C-E1	Supply/3	C-E1	Siemens	7SJ6015-xxA...	I>	400	4,00	0,80	1,00	ANSI/IEEE I...	None
					I>>	7000	70,00	14,00	0,72	Definite	None
3 R_C-F1	Supply/3	C-F1	Siemens	7SJ6015-xxA...	I>	100	1,00	0,20	15,00	ANSI/IEEE e...	None
					I>>	550	5,50	1,10	0,15	Definite	None
4 R_C-G1	Supply/3	C-G1	Siemens	7SJ6015-xxA...	I>	400	4,00	0,80	6,00	ANSI/IEEE i...	None
					I>>	2100	21,00	4,20	0,15	Definite	None
5 R_C-H1	Supply/3	C-H1	Siemens	7SJ63_5A	I>> 50_2	1992	12,45	2,49	0,25	Definite	None
6 R_C-I1	Supply/4	C-I1	Siemens	7SJ6015-xxA...	I>	320	2,00	0,40	1,30	ANSI/IEEE i...	None
					I>>	3520	22,00	4,40	0,11	Definite	None
7 R_C-M1	Supply/4	C-M1	Siemens	7SJ63_5A	I2>> 46_2	50	0,50	0,10	4,81	Definite	None

Figure 33.14.4: *ComProtreport* Tabular report

Relay models (and sometimes stages depending on the setting detailed above) are listed vertically while settings are listed horizontally.

The downward pointing triangular icon at the top of the page can be used to export the report either as HTML format or in excel spreadsheet format.

It is also possible to interact with the data within the report. For instance, if you double click on a particular stage (or right-click and select *Edit*) it is possible to edit the settings dialog for that stage.

Data within this table may also be copied and pasted if required, with or without column headers.

### 33.14.3 Results in single line graphic

The names of the relays or the tripping times may be made visible in the single line graphic by selecting the following options in the main menu.

- *Output - Results for Edge Elements - Relays*
- *Output - Results for Edge Elements - Relay Tripping Times*

The first option (*Relays*), which is always available, will show the names of the relays in all cubicles. The second option will show the tripping times of the relays after a load-flow or short-circuit calculation has been carried out. If a relay does not trip, then a tripping time of 9999.99 s is shown.

It is also possible to colour the single line graphic depending on the tripping time of the protective devices installed. This feature can be activated by clicking the diagram colouring button from the local graphics window icon bar, then selecting: the *protection* tab → 3. *Others*→ *Results*→ *Fault clearing time*.

## 33.15 Protection Audit

The protection audit tool allows a user to examine the performance of a protection system in a highly automated manner where the rigorousness of the examination is fully configurable. The purpose of the analysis is to give confidence to the user that their protection scheme performs in accordance with their particular coordination and operation criteria and to identify where weaknesses or failures in the scheme's performance exist. The automated nature of the tool allows a multitude of fault scenarios to be examined without burdening the user with the responsibility of maintaining high levels of concentration during what would otherwise be a highly repetitive and time consuming task.

As a prerequisite for using this tool it is necessary that the user has a network model including a protection scheme for which settings have already been defined. The verification process involves the application of short-circuit calculations of various fault types throughout the relevant network area. The user specifies the network area to be examined and then configures the fault cases to be applied. The protection devices responsible for each network element are determined automatically by the feature. The fault cases specified by the user are calculated at each network element and at additional relevant locations within the selected area and the reaction of all protection devices to those faults is recorded.

Once the Protection Audit Command has been executed the results must be analysed. In order to carry out this step of the analysis an additional command, the Protection Audit Results command (*ComAuditreport*) is provided. The user configures this command with their particular coordination and operation requirements so that it can determine the degree to which these requirements are met. The output of the command is a set of tabular reports clearly highlighting critical and non-critical cases where the protection scheme has failed to meet the requirements, as well as the cases where no problems were recorded.

### 33.15.1 Protection Audit Command Handling

The Protection Audit Command is initiated using the Protection Audit command (*ComProtaudit*) icon  available from the protection toolbox accessible by clicking the  icon in the main menu.

The first stage of the analysis is to carry out short circuit calculations at locations throughout the network and to examine the resulting tripping times of the defined protection devices. The tripping times are then stored in a results file cross referenced to the relevant fault case. The fault cases to be applied as well as other settings necessary to carry out this analysis are configured in the Protection Audit Command (*ComProtaudit*). Once initiated, the command is stored within the active study case and is automatically provided with a short circuit sweep command (*ComShcsweep*). The short circuit sweep command in turn contains the short circuit command (*ComShc*) to be used throughout the analysis along with a Short Circuits folder (*IntEvtshc*). The Short Circuits folder contains the short circuit events (*EvtShc*) which define the individual fault cases to be applied throughout the analysis. See Section [33.11](#) for information on the short circuit sweep command.

*PowerFactory* automatically determines the circuit breakers and associated protection devices relevant for the calculation using a topological search routine which is adapted according to the type of network element under consideration.

Configuration of the Protection Audit command settings is explained in the sections below.

#### 33.15.1.1 Protection Audit Command. Basic Options

**Network area - Selection.** The network area over which the analysis shall be carried out, should be selected here. The network area can be defined in terms of individual primary network elements or grouping objects or a general set containing references to a custom selection of individual primary network elements or grouping objects. Grouping objects which are valid for the analysis are Path Definition (*SetPath*) objects, Grid (*ElmNet*) objects, Area (*ElmArea*) objects and Zone (*ElmZone*) objects. Substation (*ElmSubstat*), Branch (*ElmBranch*) and switch elements may not be selected.

#### Calculation commands

- **Load Flow.** A load flow command may be configured via this option. The load flow command will be used to examine whether any protection devices have been unintentionally configured such that they will trip during normal load conditions. The load flow command will also be relevant if the fault cases are configured to use the complete short circuit method. Finally the load flow command may be necessary to determine the pre-fault current for any thermal overload type protection or fuse protection used in the network. The load flow command configured via this option is the standard load flow command accessible from the main toolbar.
- **Short-Circuit.** The short circuit command to be used throughout the analysis can be configured via this option. The most important setting here is likely to be the short circuit method to be used for the analysis. e.g. IEC 60909, Complete etc. The short circuit fault types and the fault impedance will be defined separately in the fault case definitions, which are configured via another part of *ComProtaudit* command (see below). The short circuit command used by the calculation is separate from the standard short circuit command accessible from the main toolbar. This means it can have independent settings.

**Fault case definitions.** The fault cases to be examined during the analysis are defined in this section of the dialog. A fault case is defined as a short circuit event (*EvtShc*) and is stored within a Short Circuits folder. The contents of the Short Circuits folder can be accessed at any time by pressing the Edit button. For each fault case a Fault type (e.g. 3 phase, single phase to ground etc) and a fault resistance and reactance should be defined. It is envisaged that a user will ideally define multiple fault cases, and examine different fault impedances, for each fault type, for each analysis so as to comprehensively test their protection scheme.

### Considered network equipment

- **Branches.** If this option is selected, faults will be examined relevant for the protection of branch elements such as lines located within the network area over which the analysis is to be carried out. Fault cases will be applied at the terminals of the branches and at regular intervals along the length of any line objects. Where protection devices are associated with the element, fault cases will be applied on both sides of the relevant current transformers.
- **Busbars.** If this option is selected, faults will be examined relevant for the protection of busbar elements located within the network area over which the analysis is to be carried out. Busbars are considered to be terminal elements where the usage parameter of the terminal is set to “busbar”. Fault cases will be applied at the busbars themselves and where protection is present in cubicles contained within the busbar, fault cases will be applied on both sides of the relevant current transformers.
- **Step size for lines.** Faults will be applied at regular intervals along the length of line elements during the analysis. This setting determines the size of the interval to be applied.

**Results - Results File.** With this option the results file used to store the results of the analysis can be selected and accessed. The results file is structured into multiple sub results files with one sub results file being used per fault case examined. Another sub results file stores the results of the load flow calculation and another contains details of the topological relationships of circuit breakers controlled by protective devices. The default results file location is within the relevant study case.

**Reporting - Command.** This option provides access to the associated Protection Audit Results command which is described in Section 33.15.2. By selecting the Reporting check box the user can choose to additionally execute the Protection Audit Results command, when the Protection Audit command is executed.

#### 33.15.1.2 Protection Audit Command. Advanced Options

##### Stop criteria of topological search

- **Max. Number of busbars.** The command uses a topological search to locate the switches controlled by protective devices, that are used to protect the elements in the network area under examination. In some circumstances, for example where the network as a whole is much larger than the protected area under examination, this search routine could lead to unnecessarily long computation times. This option specifies the maximum number of busbars which can be swept by the search routine before the search routine is halted. A value of 0 means that no limitation is applied.
- **Max. line length.** Similar to the previous option, this option specifies the maximum line length which can be swept by the search routine before the search routine is halted. A value of 0 means that no limitation is applied.

#### 33.15.2 Protection Audit Results Command Handling

The command is initiated using the Protection Audit Results command (*ComAuditreport*) icon  available from the protection toolbox accessible by clicking the  icon in the main menu. It can also be initiated directly from the Protection Audit command by selecting the Reporting checkbox as described above.

Once the Protection Audit Command has recorded the tripping times of the network's protection devices in response to the various fault cases, the next stage is to analyse the results. A results file (*ElmRes*) is made available containing all the data generated by the initial part of the analysis. The quantity of the data generated can be significant and must be further analysed against the user's particular coordination and operation requirements before the results can be displayed in a meaningful way. This

is done using the Protection Audit Results Command. Once initiated the command is stored within the active study case.

The user's particular coordination and operation requirements are entered into this command. Different severities can be assigned to different conditions and locations. For example a coordination issue between main and backup protection can be considered more severe than a coordination issue between the second backup and an upstream device.

The output of the command is a set of tabular reports which can be oriented by network element or by protection device. The tabular reports clearly highlight and differentiate between non-critical and critical cases where the protection scheme has failed to meet the requirements. Cases where no problems with requirements were recorded are also clearly indicated. The level of detail shown by the report can be scaled according to the user's needs.

Results can be assessed against:

- Fixed coordination margin
- Coordination margin based on the downstream circuit breaker delay
- Maximum allowed device tripping time
- Maximum allowed fault clearing time

Configuration of the Protection Audit Results command settings is explained in the sections below.

### 33.15.2.1 Protection Audit Results Command. Basic Options

**Result selection- Results file.** With this option the results file containing the results from the preceding analysis can be selected and accessed. The results file is structured into multiple sub results files with one sub results file being used per fault case examined. Another sub results file stores the results of the load flow calculation and another contains details of the topological relationships of circuit breakers controlled by protective devices. The default results file location is within the relevant study case.

#### Options

- **Verify device coordination.** This option enables the generation of a specific tabular report verifying the coordination performance of the relays against the users requirements. The requirements themselves are defined elsewhere as described in Section [33.15.2.2](#).
- **Verify tripping times.** This option enables the generation of a specific tabular report verifying the performance of the relays against the users requirements with respect to maximum tripping times. The requirements themselves are defined elsewhere as described in Section [33.15.2.3](#).
- **Verify fault clearing time.** This option enables the generation of a specific tabular report verifying the performance of the relays against the users requirements with respect to maximum fault clearing times. The requirements themselves are defined elsewhere as described in Section [33.15.2.4](#).

#### Report style

- **Network element oriented.** If selected, each row of the generated reports will relate to a different network element. Please see Section [33.15.3.1](#) for further information.
- **Protection device oriented.** If selected, each row of the reports will relate to a different relay. Please see Section [33.15.3.2](#) for further information.

**Colours used in report.** Colouring is used throughout the tabular reports to assist in the interpretation of the results. The colouring scheme used can be configured in this settings table. Colouring is according to four levels of severity. The highest level of severity being a critical 'Failure'. Two levels

of non critical failure are also allowed for, namely 'Warning' and 'Notification' and the lowest level of severity is termed 'no issue', where the user's requirements are completely met.

**Network Area - Output for.** These settings allow the user to report results for all elements contained within network area selection or alternatively to limit results to a subset of those elements.

- **All recorded elements:** this option is selected if the user wishes to report results for all elements contained within the network area selection. If this option is selected results for all elements are shown on the same page of the report.
- **User Selection:** this option is selected if the user wishes to restrict result reporting to a subset of the recorded elements. When chosen, a selection table will appear to which rows can be manually added and individual objects selected from the data manager. The buttons on the right hand side can be used to quickly assist with this. Results associated with each item in the selection table will be displayed on a different page of the output report.

### 33.15.2.2 Protection Audit Results Command. Device Coordination

The feature allows coordination of protection devices to either be verified against:

- a fixed, specified, coordination margin, where the duration of the breaker delays is ignored.
- a potentially variable coordination margin, based on the breaker delay of the downstream breaker.

In both cases it is verified that for a particular instance of a fault case the tripping time of the upstream relay in the coordination pair is longer than the tripping time of the downstream relay by a value greater than the coordination margin.

The settings on this page are only relevant if the verify device coordination check box is selected on the basic options page of the command and will influence the results of the associated tabular report.

#### Coordination margin

- **Use breaker delay.** This check box determines whether breaker delays or alternatively a fixed coordination margin should be used to verify coordination.
- **Coordination margin.** The fixed coordination margin which shall be used if breaker delays are not to be considered.
- **Safety margin.** This parameter is only used if the Use breaker delay check box is selected. Under those circumstances, taking the breaker delay of the downstream breaker as the initial coordination margin, this parameter will increase the coordination margin by the specified percentage.
- **Default breaker delay.** This parameter is only used if the Use breaker delay check box is selected. The breaker delay is a parameter which is specified in the type of the circuit breaker. In some cases this data may not be available or may not have been specified in the model. For those cases *PowerFactory* will use the default value specified for this parameter.

**Severity of failures.** This settings table can be configured so as to specify the severity of different kinds of failure. It is assumed that for primary coordination pairs, where one of the devices is topologically located at a level directly above the other device that a coordination failure is a critical failure and so is by default given the highest level of severity and not included in the table. For secondary and tertiary failures, where the upstream device is two or three coordination levels above the downstream device, the user has the option to select the severity level of a coordination failure. The three levels being Failure, Warning, or notification in decreasing order of severity.

### 33.15.2.3 Protection Audit Results Command. Tripping Times

The feature allows the maximum tripping time of the protection devices to be verified against:

- different classes of fault (3 phase, 2 phase and single phase)
- whether the fault being examined is on the primary item of equipment being protected, or a secondary or tertiary item of equipment.

The tripping time of the relevant relays for the examined instance of the fault case will be verified against the specified maximum tripping time.

The settings on this page are only relevant if the verify tripping times check box is selected on the basic options page of the command and will influence the results of the associated tabular report.

**Max. trip time.** This settings matrix is used to specify maximum tripping times according to the different classes of fault in combination with whether the network element on which the fault instance is applied is considered to be primary, secondary or tertiary equipment from the point of view of the relay being considered.

**Severity of failures.** This settings matrix can be configured so as to specify the severity of different kinds of failure. The user has the option to select the severity level for each of the combinations defined by the matrix. The three levels of severity being Failure, Warning, or Notification in decreasing order of severity.

### 33.15.2.4 Protection Audit Results Command. Fault Clearing Times

The feature allows the maximum fault clearing time of the protection devices to be verified against whether the fault being examined is on the primary item of equipment being protected, or additionally a secondary or tertiary item of equipment. The maximum fault clearing time is distinct from the maximum tripping time in that it takes into account the delay introduced by the circuit breaker as well as the delay introduced by the relay, while the maximum tripping time considers the relay delay only.

The fault clearing time of the relevant relays-circuit breaker combinations for the examined instance of the fault case will be verified against the specified maximum fault clearing time.

The settings on this page are only relevant if the verify fault clearing times check box is selected on the basic options page of the command and will influence the results of the associated tabular report.

**Max. fault clearing time.** This settings table is used to specify maximum fault clearing times according to whether the network element on which the fault instance is applied is considered to be primary, secondary or tertiary equipment from the point of view of the relay being considered.

**Severity of failures.** This settings table can be configured so as to specify the severity of different kinds of failure. The user has the option to select the severity level for primary, secondary or tertiary equipment, with the three levels of severity being Failure, Warning, or Notification in decreasing order of severity.

**Default breaker delay.** The breaker delay is a parameter which is specified in the type of the circuit breaker. In some cases this data may not be available or may not have been specified in the model. For those cases *PowerFactory* will use the default value specified for this parameter.

**Critical fault clearing times (3-phase).** Certain items of equipment may require specific fault clearing times distinct from the general verification based on the Max. fault clearing time parameter described above. For example verification may be required to ensure that disconnection of equipment occurs within the critical fault clearing time of a particular generator such that dynamic stability problems do not arise. For such cases it is possible to specify an alternative critical fault clearing time. The element with the special requirement and the associated critical fault clearing time to be verified against three phase fault cases are entered in the table.

### 33.15.3 Report Handling and interpretation

The Protection Audit Results Command (*ComAuditreport*) was described in Section 33.15.2. Once this command has been executed, one or more reports may be generated. This section describes the handling and interpretation of those reports.

The type of report is displayed in the header. There are three possible headers (verify device coordination, verify tripping time and verify fault clearing time). For each report, the Project title is displayed as well as the active study case when the reports were generated. Three buttons are included on the right hand side of the dialog. A refresh button, which may be used to rebuild the report under particular circumstances. An html button which may be used to launch relevant software (e.g. a web browser) and generate an html format version of the report. Finally The xls button will launch relevant software to handle xls files (e.g. spreadsheet software) and will generate an xls format version of the report.

The Network Area section of the report will be selectable if the Network Area - Output for: user selection option has been configured (see Section 33.15.2.1) and can be used to move between pages, where each page is associated with a different network area.

Two complimentary forms of report may be generated by selecting the relevant option on the Basic Options page of the Protection Audit Results dialog (see Section 33.15.2.1). The handling and interpretation of these are described below.

#### 33.15.3.1 Network element oriented

When selected, each row of the generated report(s) will relate to a different network element. Results are presented in terms of a coloured bar. Segments of the coloured bar represent different fault positions associated with the element. The segments are coloured according to the worst violation per fault position on the element (i.e. if there are two warnings and one failure at for example 20% of a line, this segment will be coloured according to the failure).

Figure 33.15.1 shows a typical example of a report. The first column of results, the Total column gives an overview of the results considering all fault cases. In the subsequent columns results are grouped into fault case classes. For instance if phase A to ground, phase B to ground and phase C to ground fault cases are examined, these results will all be grouped together in the single phase to ground column. It is also possible to see the results for the individual fault cases associated with a grouping. For example to see the individual A,B or C phase to ground fault results click on one of the bars in the single phase to ground column and select the Expand selected fault type button. The table will be rebuilt to list the individual fault case results associated with the selected column. See Figure 33.15.2.

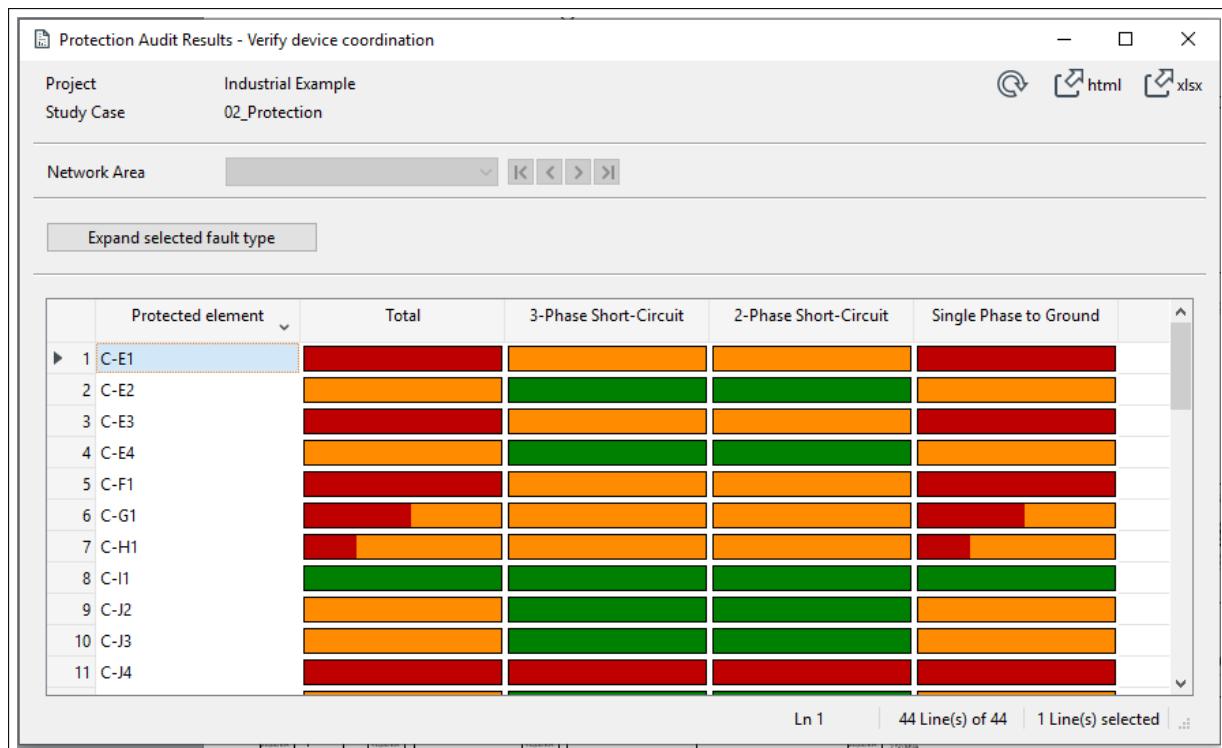


Figure 33.15.1: Network element oriented results example

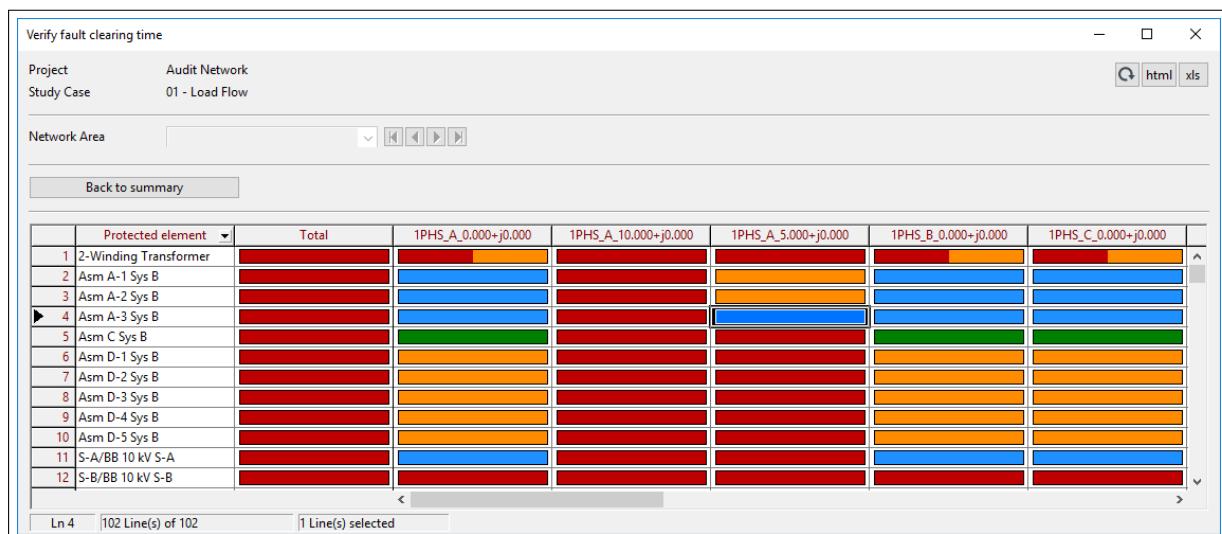


Figure 33.15.2: Network element oriented results example - single phase to ground expanded view

### 33.15.3.2 Protection device oriented

When selected, each row of the generated reports will relate to a different relay model. Results are presented in terms of a coloured bar. Each relay is determined to be responsible for providing either primary, secondary or tertiary protection for certain items of equipment. For each item of equipment the relay response will be examined for a finite number of fault case instances. Each segment of the coloured bar represents a different fault case instance associated with the relay, relevant to the column being examined. The segments are coloured according to the severity of the violation measured for each fault case instance. e.g. a relay might be found to be responsible for providing primary protection to a line, secondary protection to a switchboard and tertiary protection to a second line. A total of 25

fault locations may be examined for these three items of equipment in relation to three different fault cases, giving a total of 75 fault case instances. 10 cases may be found to be failures 15 to be warnings and 50 to have no issue. In this example the bar in the Total column for that relay would be coloured according to those proportions.

Figure 33.15.3 shows a typical example of a report. The first column of results, the Total column, gives an overview of the results considering all fault case instances. In the subsequent columns results are grouped into fault case classes. For instance if phase A to ground, phase B to ground and phase C to ground fault cases are examined, these results will all be grouped together in the single phase to ground column. It is also possible to see the results for the individual fault cases associated with a grouping. For example to see the individual A,B or C phase to ground fault results click on one of the bars in the single phase to ground column and select the Expand selected fault type button. The table will be rebuilt to list the individual fault case results associated with the selected column. See Figure 33.15.4.

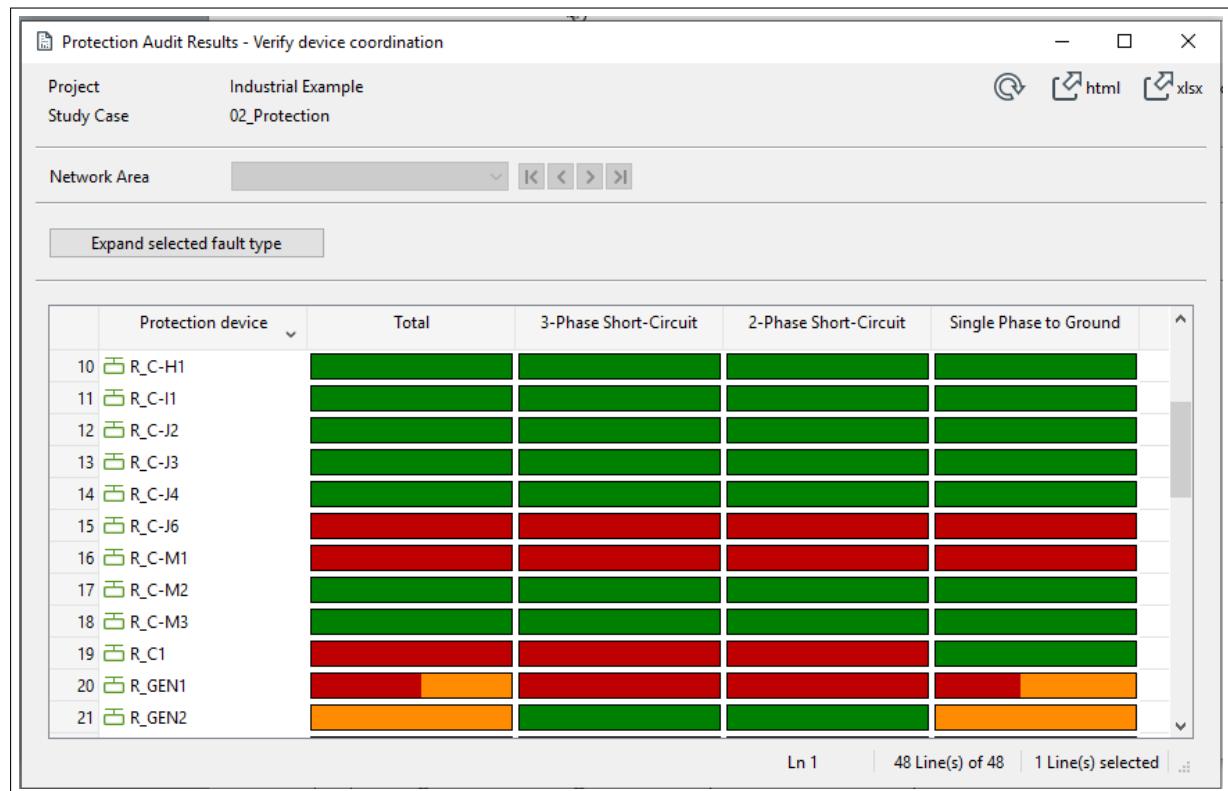


Figure 33.15.3: Protection device oriented results example

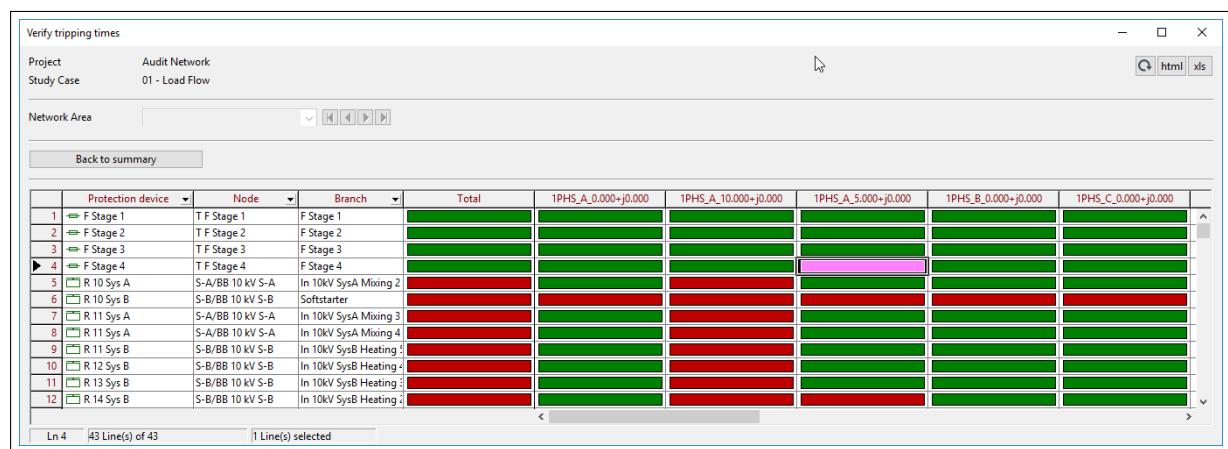


Figure 33.15.4: Protection device oriented results example - single phase to ground expanded view

## 33.16 Short circuit trace

The Short circuit trace is a tool based on the complete short circuit calculation method that allows the user to examine the performance of a protection scheme in response to a fault or combination of faults; where the response is examined in time steps and where at each time step, the switching outcomes of the previous time step and the subsequent effect on the flow of fault current, is taken into consideration.

Consider a network as illustrated in Figure 33.16.1:

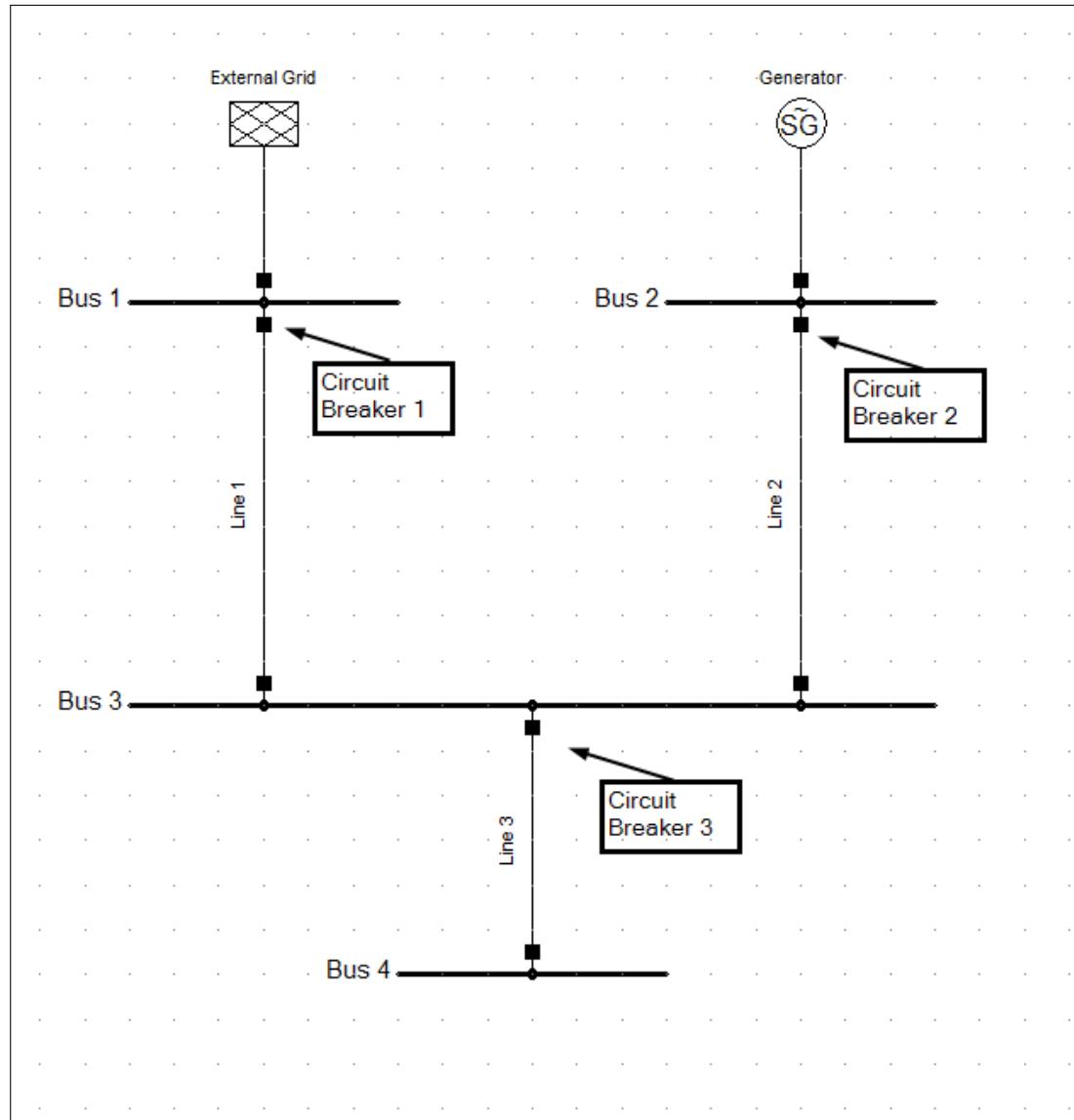


Figure 33.16.1: Short circuit trace example

Suppose that for a particular fault at bus 4, the relay controlling circuit breaker 1 trips significantly faster than the relays controlling circuit breakers 2 and 3. Once circuit breaker 1 trips, the fault is not cleared but the fault current is reduced, since the contribution from the external grid is removed.

With a single conventional static short circuit calculation it is not possible to take account of the fact that the fault current seen by circuit breaker 3 reduces after circuit breaker 1 trips. The short circuit calculation results will show a tripping time for circuit breaker 3 that is too quick. It will be based on the incorrect premise that circuit breaker 3 sees contribution from both the external grid and the synchronous generator for the entire duration of the fault.

It should be possible to more accurately determine the sequence of operation of the protection scheme. To clear the fault completely, circuit breaker 2 or circuit breaker 3 must trip. Due to the dynamic variation in the fault current, the tripping times of the two circuit breakers are not immediately obvious. Ideally, a dynamic simulation method should be used to accurately calculate the respective tripping times of the two circuit breakers. However, a dynamic simulation is not always practicable and where the user is willing to accept a reduced level of accuracy in exchange for a faster, simpler calculation result, then the Short circuit trace should be considered.

Consider again the network illustrated in Figure 33.16.1 with a fault occurring at bus 4. All relays are overcurrent relays with the relay controlling circuit breaker 1 having a significantly faster tripping time than the other two relays. A sequence of events as illustrated in Figure 33.16.2 occurs.

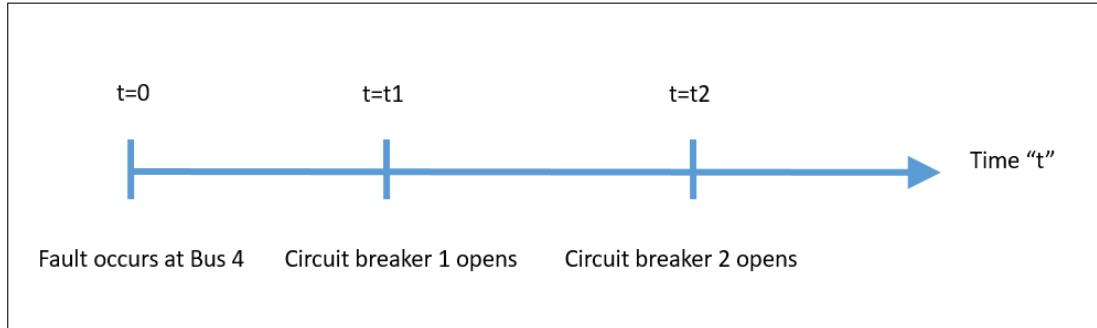


Figure 33.16.2: Short circuit trace sequence

The following describes how the Short Circuit Trace calculation arrives upon that sequence of events.

- Time Step 1 ( $t = 0$ ): The fault occurs at bus 4. Fault current flows from both the synchronous generator and the external grid according to the complete short circuit method of calculation. The relay controlling circuit breaker 3 sees the fault current from both sources. The relays controlling circuit breakers 1 and 2 see only the fault current from the sources present in their particular branch of the network. The tripping time of each of the relays can be evaluated based on the respective magnitudes of the current components seen by the relays and with reference to each of the relay's tripping characteristics.
- Time Step 2 ( $t = 0 + t1$ ): According to the tripping times calculated at Time Step 1 it is established that the relay controlling circuit breaker 1 will trip first in time  $t1$ . Therefore at stage 2 circuit breaker 1 is opened and the complete short circuit method calculation is once again carried out for a fault at bus 4. This time, the current seen by circuit breaker 3 only includes contribution from the generator and not from the external grid.  
That is not to say that the influence of the external grid is erased from the record during this second time step. Where an inverse-time characteristic applies, the time spent in the previous trace calculation step, with both sources supplying current is used to determine the initial state of the relay moving into the second time step in order to better approximate the tripping time. Additionally consideration has been given to accurate representation of cases where the function responsible for the trip, changes between time steps (e.g. from DT to IDMT units).  
For the purposes of this example it is assumed that circuit breaker 2 and circuit breaker 3 are malcoordinated and circuit breaker 2 is established to be the next quickest to operate.
- Time Step 3 ( $t = 0 + t2$ ): According to the tripping times calculated at Time Step 2 it is established that the relay controlling breaker 2 is the next to trip and trips in time  $t2$ . Since the fault is now isolated from all connected sources, fault current no longer flows and the short circuit trace calculation is complete.

From the above, a sequence of operation for the protection scheme is established and specific protection operating times are calculated, taking account of the variation in network topology that occurs during the ongoing response of a protection scheme to a fault situation.

### 33.16.1 Short Circuit Trace Handling

A command specifically for the Short Circuit Trace feature can be accessed by clicking on the *Start Short-Circuit Trace* icon on the *Protection* toolbar. The command can also be executed from the context menu by right-clicking on busbar or line in the data manager or single line diagram and by choosing *Calculation → Short Circuit Trace....*. When executing the command from the context menu a short circuit event referencing the selected item of equipment will automatically be generated and presented to the user in a browser dialog. The user should use this to configure the nature of the short circuit to be applied.

The Short-Circuit Trace command dialog (*ComShctrace*) has only one page which is illustrated in Figure 33.16.3.

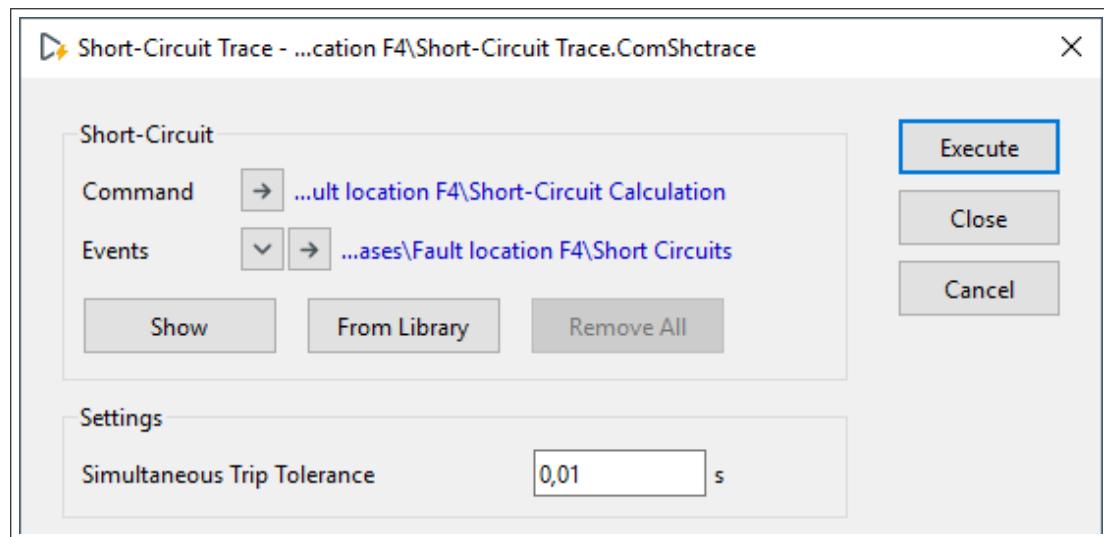


Figure 33.16.3: Short circuit trace command dialog

#### The Short-Circuit Trace Command

A link to the short circuit command (*ComShc*) to be used for the calculation is automatically generated. This command is described in detail in the Chapter 26. Please note that for the Short Circuit Trace function, some options are fixed. For instance, only the complete short circuit method may be selected whilst the type of short circuit is specified in the short circuit event rather than in the command.

The *Events* part of the page is used to define the events to be applied at the beginning of the calculation. The following kinds of events may be specified.

- Intercircuit Fault Events (*EvtShcl*)
- Outage Events (*EvtOutage*)
- Short-Circuit Events (*EvtShc*)
- Switch Events (*EvtSwitch*)

A list of the selected events can be easily accessed by pressing the *Show button*.

Predefined events can be selected from the project's operational library by selecting the *From Library* button.

All events can be removed from the event queue by pressing the *Remove All* button.

The *Simultaneous Trip Tolerance* setting is used to minimise the number of time steps presented for cases where the operation of devices is expected to be practically simultaneous. If the breakers at both

ends of a protected line (for example) operate in times which have a difference between them less than the *Simultaneous Trip Tolerance* then operation of both devices will be presented during a single time step corresponding with the switching time of the fastest operating device.

Once the simulation is ready to begin, press the execute button. At this point the simulation is initialised and the short circuit events specified in the Basic Options page are applied to the network.

The user is presented with a tabular report as illustrated in Figure 33.16.4. The header of the report states the project name and the active study case along with the currently examined *Time Step*. Initially this time step will correspond with the *Switch Time* of the fastest acting circuit breaker / protection device combination. However, as the trace is progressed this value will be updated. The user is given the option to display *Only active devices* (i.e. devices which are instigated to trip in response to the fault) or *All devices*.

The devices themselves are shown in the first column of the table. It is possible to edit the devices by double left clicking on the cells in this column. A context menu can be presented by right-clicking on a cell in this column and from here it is possible to mark the location of the device in a single line diagram. The *Protection device Time [s]* column presents the tripping times of the protection devices in response to the fault during the examined time step. This time does not include any breaker delay introduced by the associated switching device. The associated switching device is presented in the *Switch* column and the total fault clearing time taking account of the breaker delay as well as the protection device operating time is presented in the *Switch Time [s]* column. It is the Switch time which determines the sequence of operation of the relays during the trace and therefore the examined Time steps. The device(s) identified to trip during the examined time step are indicated with an arrow in the row header of the table.

By pressing the *Next Time Step* button the user can advance the trace to the next time step and the table of results is updated (along with the single line diagram). The report also provides buttons which allow the user to jump to the last time step or stop the trace.

The final column in the table *Switch Blocked* can be used to simulate failure of one or more switches to operate during the trace. If the box corresponding with a device is checked in the table, upon pressing the *Next Time Step* button the operation of the blocked switch will be disabled. The blocking will be considered to occur before the results of the next time step are calculated. With the switch operation disabled, one or more different protection devices will become the next switch to operate. In this way the performance of back up protection for example can be examined.

The report can be refreshed as well as exported in html or xlsx file formats using the buttons in the top right hand corner.

Project		Industrial Example		
Study Case		02_Protection		
Time Step		0,000 s		
Display		Only active devices	<	>
<a href="#">Next Time Step</a>		<a href="#">Jump to Last Step</a>	<a href="#">Stop Trace</a>	
Protection device	Protection device	Switch	Switch	Switch
	Time [s]		Time [s]	Blocked
► 1 -> R_T1	0,035	CB1	0,035	<input type="checkbox"/>
2 R_C-E1	0,755	B 3_1	0,755	<input type="checkbox"/>
3 R_T7-2	0,860	B 19_2	0,860	<input type="checkbox"/>
4 R_T7-1	0,871	B 19_3	0,871	<input type="checkbox"/>
5 R_C1	0,920	B 3_5	0,920	<input type="checkbox"/>
6 R_T1_sec	0,920	B 3_6	0,920	<input type="checkbox"/>
7 R_GEN1	1,000	B 50_3	1,000	<input type="checkbox"/>
8 R_C-H1	2,050	B 3_4	2,050	<input type="checkbox"/>
9 R_C-J3	2,050	B 50_1	2,050	<input type="checkbox"/>
10 R_T2DC	2,050	B 6_2	2,050	<input type="checkbox"/>
11 R_M_T17	2,050	B 5_1	2,050	<input type="checkbox"/>
12 R_C-F1	2,650	B 51_1	2,650	<input type="checkbox"/>
13 R_C-F1	4,098	B 3_2	4,098	<input type="checkbox"/>
14 R_M-T10-2	4,330	B 28_1	4,330	<input type="checkbox"/>

Ln 1 | 31 Line(s) of 31 | 1 Line(s) selected | ...

Figure 33.16.4: Short-Circuit Trace Report

As an alternative to or in addition to using the report, the user can examine the results in the single line diagram. The user can also advance through the simulation time step by time step or to the end of the simulation by clicking on the relevant icons on the *Protection* toolbar. Further there is an additional icon to stop the simulation at any time. The icons are illustrated in Figure 33.16.5.

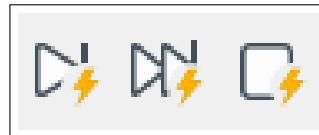


Figure 33.16.5: Short Circuit trace icons

## 33.17 Protection Graphic Assistant

The Protection Graphic Assistant is a command which is accessible via the protection toolbox. The assistant provides a convenient location where a number of graphical features relevant to protection analysis can be initiated. On the basic options page of the command shown in Figure 33.17.1 the user can choose which of the features to execute. Each of the features is executed independently from each other and only one feature can therefore be selected at a time. On this page the user should also select the protection devices to be considered by the command.

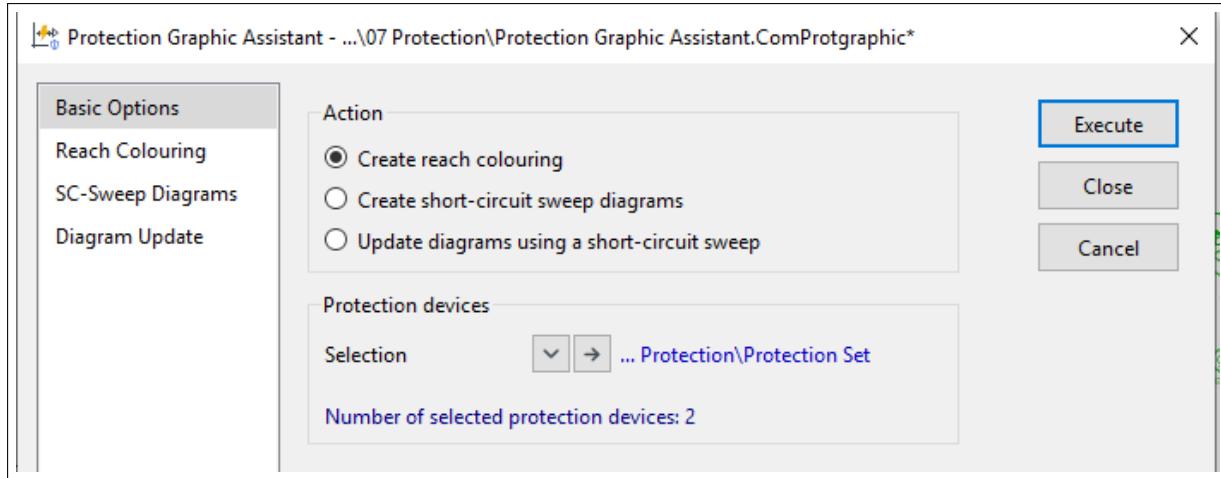


Figure 33.17.1: Protection Graphic Assistant - Basic Options

The following subsection describe the features themselves.

### 33.17.1 Reach Colouring

Reach colouring is used to visualise the zone reaches of protection relay distance elements. It can be used to overlay the zone reaches of specified relays on the single line diagram, complementing other methods of zone reach visualisation offered by the R-X plot and Time-Distance plot. A typical reach colouring representation is shown in Figure 33.17.2.

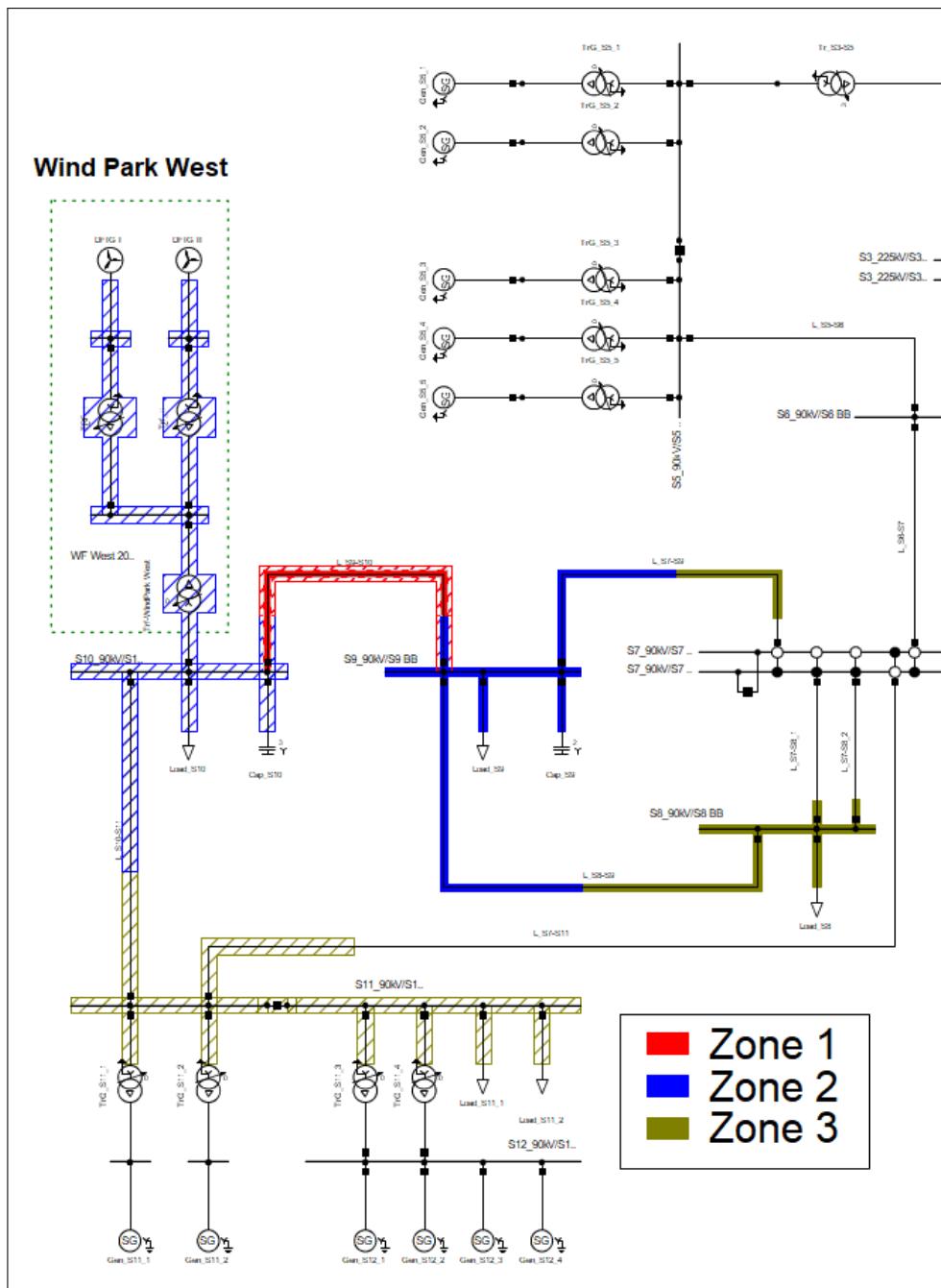


Figure 33.17.2: Distance protection reach colouring in the single line diagram

In the figure the zone reaches of two relays at either end of a single line are examined. The reaches of one of the relays is shown with solid brush style, while the reach of the other is shown with a thicker brush filled with diagonal hatching. The colouring of the three zones is indicated in the diagram's key.

Reach settings are extracted directly from the relay elements and *PowerFactory* scales the presented reach, taking account of the drawn lengths of the lines in the single line diagram. Starting elements based on impedance characteristics as well as overreach zones can also be visualised using the tool.

The Protection Graphic Assistant is used to configure the appearance of the reach colouring. Relays are distinguished by line thickness and can be distinguished further by the selection of identifying brush styles. Zones can be individually identified through the selection of appropriate colouring. Additionally, the tool allows the width of the line colouring to be controlled. These attributes are configured on the Reach Colouring page of the command's dialog. This is illustrated in Figure 33.17.3

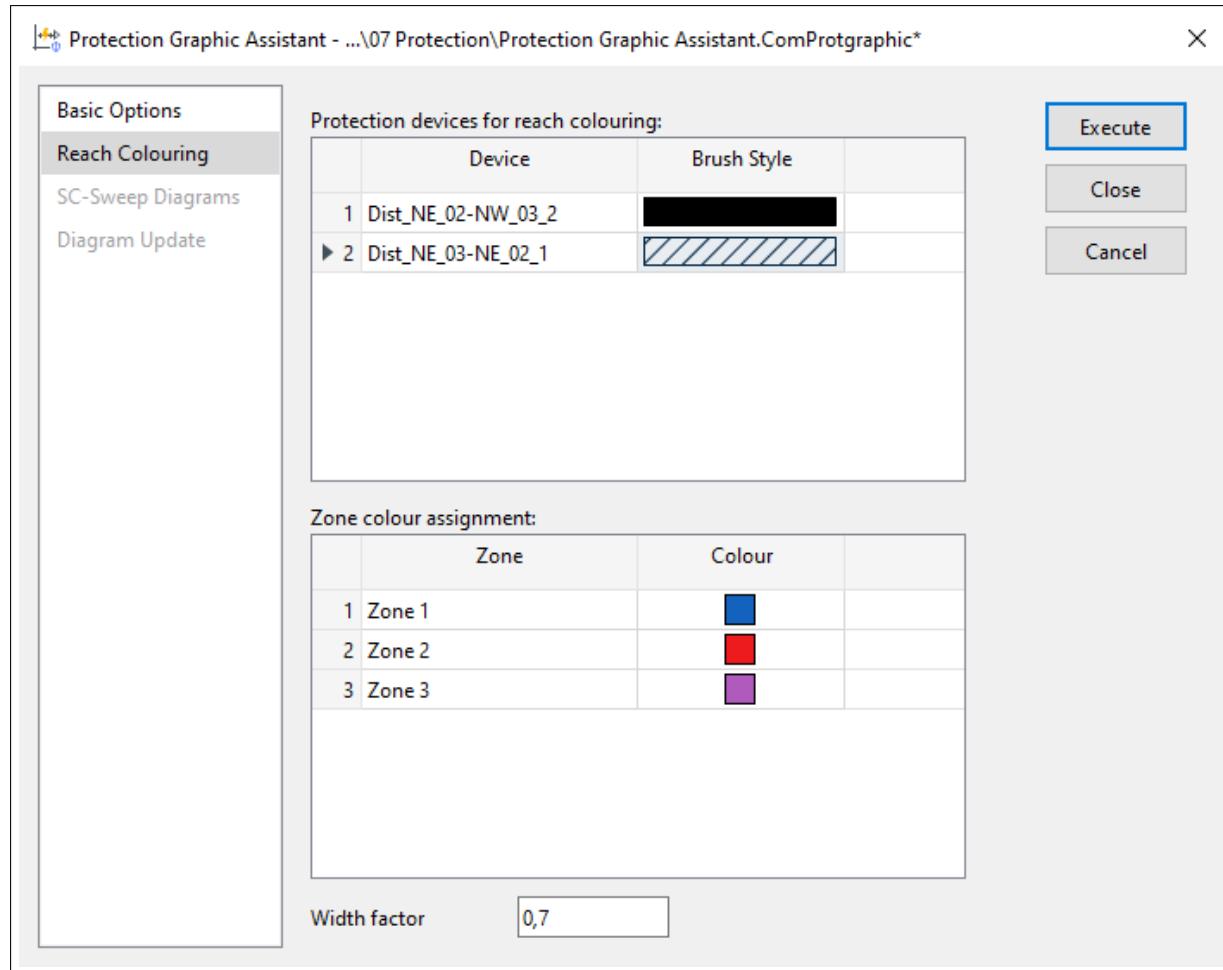


Figure 33.17.3: Protection Graphic Assistant - Reach Colouring

After addition of the protection reach colouring to the single line diagram, the colouring can later be removed, by pressing the Protection Graphic Assistant icon a second time.

The visualisation provides the user particular insight into more complex topological cases such as with the protection of parallel lines. In such a case for example, the tool will clearly illustrate the reach of the second and third zones beyond the remote busbar and back towards the relaying location via the parallel line, as well as the reach into other branches beyond the remote busbar into the remainder of the network. Additionally, for a network model populated with distance elements, by examining the visual gaps in the colouring, the tool may be used to effect a rapid protection audit. Thereby, helping the user to identify gaps in the protection scheme.

### 33.17.2 Short-Circuit Sweep Plot

The *Short-Circuit Sweep Plot* itself was described in Section 33.12 while this subsection describes the use of the Short-Circuit Sweep Plot create feature of the Protection Graphic Assistant. This feature is used to easily and quickly configure short-circuit sweep diagrams which are of particular relevance for protection purposes.

If the feature is selected on the Basic Options page of the Protection Graphic Assistant an additional field appears where the user is able to define the nature of the short circuit sweeps to be carried out using the Short-Circuit Sweep command (see Section 33.11 for more information on the short circuit sweep command).

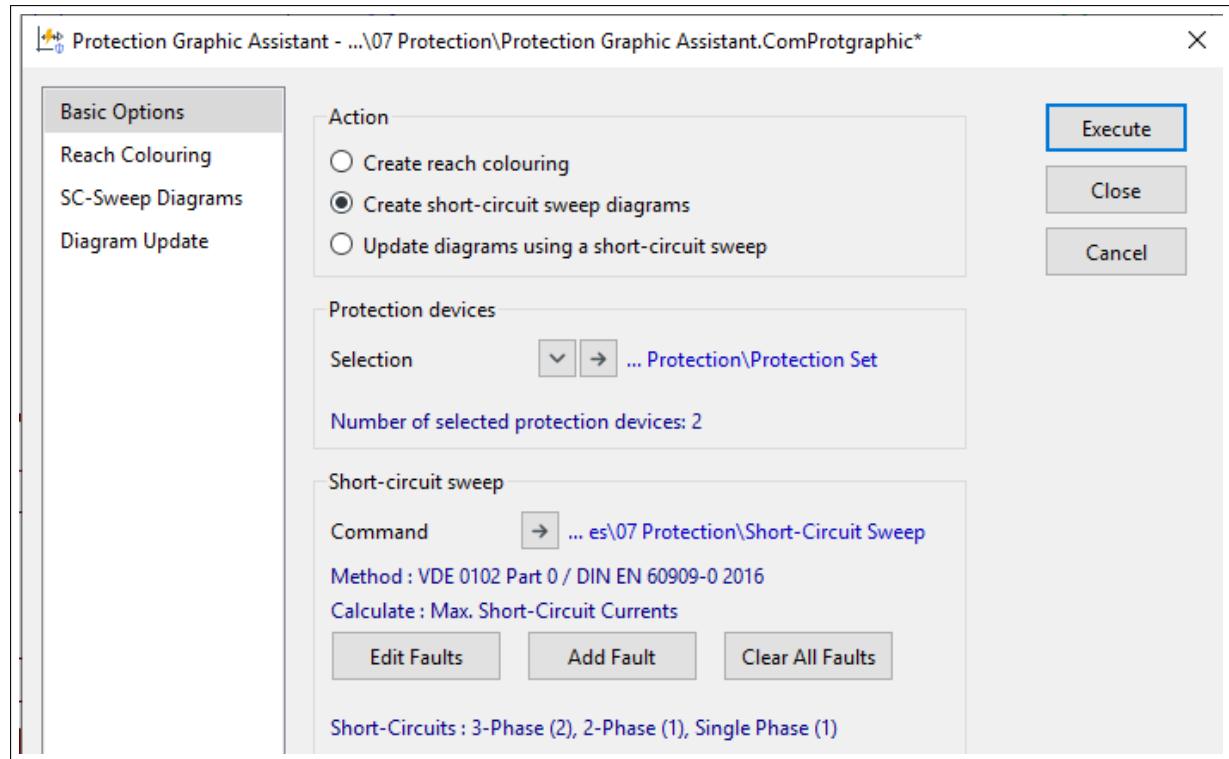


Figure 33.17.4: Protection Graphic Assistant - Basic Options for Short-Circuit Sweep Diagrams

On the SC-Sweep Diagrams page of the Protection Coordination Assistant the user provides the command with the relays of interest and the path to sweep. The types of diagrams desired are also specified. This is illustrated in Figure 33.17.5.

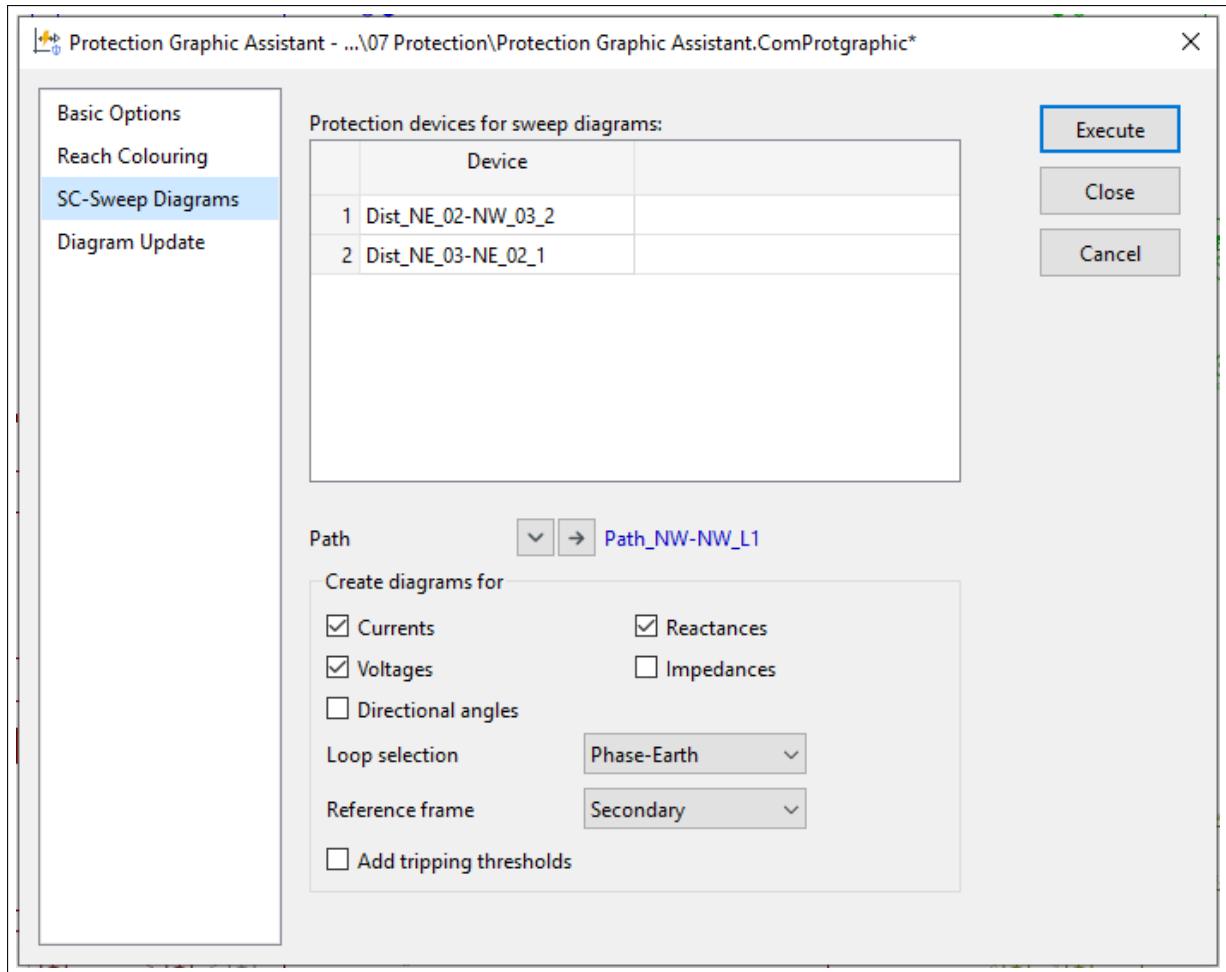


Figure 33.17.5: Protection Graphic Assistant - Short-Circuit Sweep Diagrams

The feature will then automatically determine the elements within the relays, and the corresponding result variables that must be monitored during the short circuit sweep so as to create the diagrams specified. The user can choose whether they are interested in phase-phase protection loops, phase-earth protection loops or both. They can also choose to include tripping thresholds in the diagrams and whether to display secondary result values (values calculated for the secondary side of the CT's and VT's) or primary system values (values calculated on the primary side of the CT's and VT's). Once executed the diagrams are created with a single plot page per relay. The corresponding short circuit sweeps will also be executed and the results will be visible in the plots.

### 33.17.3 Diagram Update

Once short circuit sweep diagrams or time distance diagrams have been created it may be necessary at some later stage to update the diagrams by re-executing the short circuit sweep behind them. A change to network data or a change to one or more relay settings could motivate such a course of action. The diagram update option in the Protection graphic assistant allows the user to update one or more diagrams by re-executing each relevant short circuit sweep calculation in one go. The user has the option to select all graphical plot pages or a defined selection of pages as required.

## 33.18 Building a basic overcurrent relay model

Some advanced users may need to build their own relay models. This section will outline the procedure for building a basic overcurrent relay model.

1. Create a new block definition for the relay frame
  - Select *Insert → Dynamic Model → Composite Model Frame...*
  - Give the relay frame an appropriate name.
  - Click **OK**. This creates a block definition object within the User Defined Models section of the project library.
2. Construct the relay frame.
  - Select the slot icon from the drawing toolbox located on the right side of the screen and place 6 slots within the block definitions arranged as illustrated in Figure 33.18.1 below.

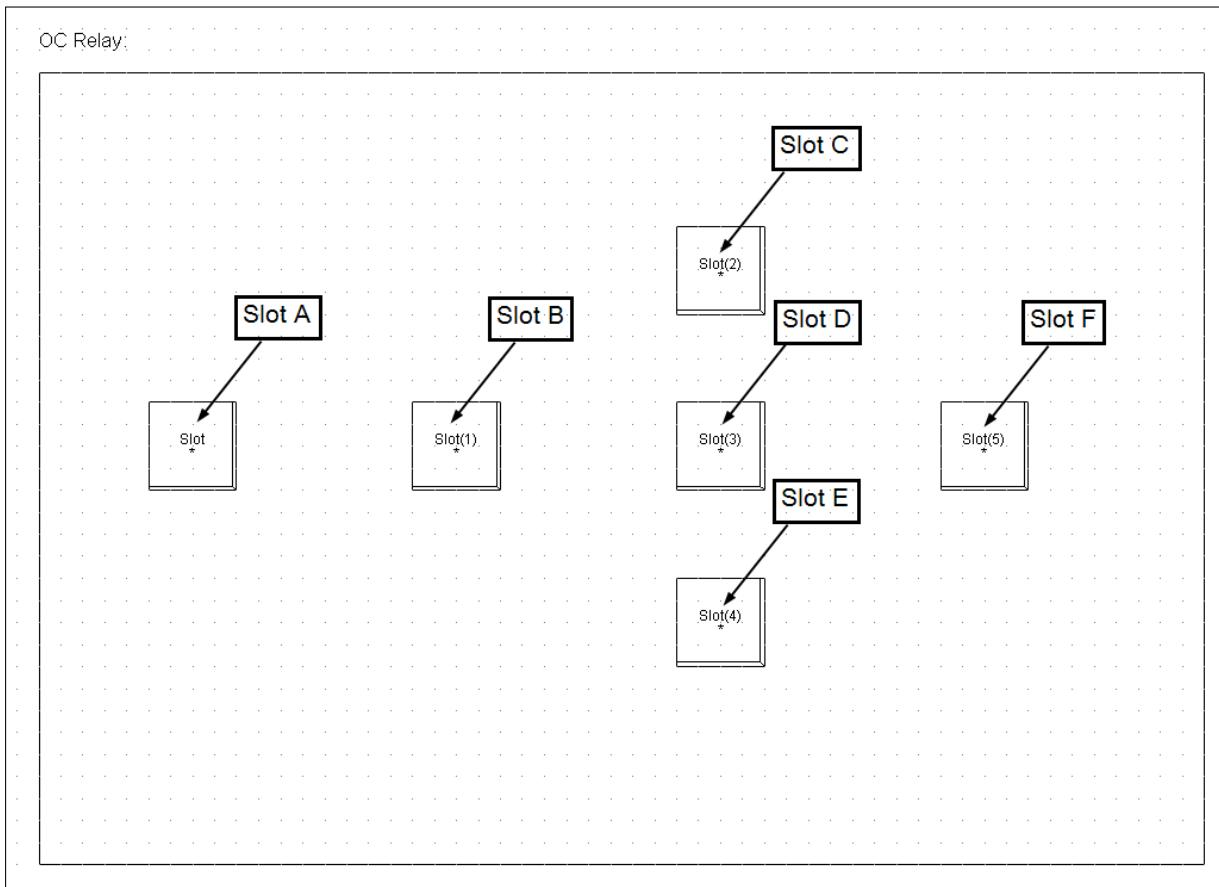


Figure 33.18.1: Arrangement of slots

3. Configure the *BlkSlot* dialog for slot A.
  - Slot A will be configured to be a CT slot. Double click on the slot symbol and the *BlkSlot* dialog will appear.
  - Enter an appropriate name for the slot e.g.. CT 3ph.
  - Enter the class name as StaCt\*.
  - Ensure that only the box linear is checked in the classification field.
  - Enter the following output signals under the variables field: I2r\_A; I2i\_A, I2r\_B; I2i\_B, I2r\_C; I2i\_C. These signals will represent real and imaginary secondary currents for phases A, B and C.

- The way in which the signal list above is defined influences the way the signals are represented in the relay frame. Signals can be grouped together and represented by a common terminal by separating the signals to be grouped with a semicolon. Where a group of signals or a single signal is to be given its own terminal representation in the relay frame then the signal or group of signals should be distinguishing from any other signals by separation with a comma.
- Once configured, click **OK**. The CT slot should now be marked with three terminals, one for each phase.

4. Configure the *BlkSlot* dialog for slot B.

- Slot B will be configured to be a Measurement slot. Double click on the slot symbol and the *BlkSlot* dialog will appear.
- Enter an appropriate name for the slot e.g.. Measurement.
- Enter the class name as RelMeasure\*.
- In the classification field, ensure that only the boxes linear and Automatic, model will be created are checked
- Enter the following output signals under the variables field: I\_A, I\_B, I\_C. These represent RMS values of current for each phase.
- Enter the following input signals under the variables field: wlr\_A; wli\_A, wlr\_B; wli\_B, wlr\_C; wli\_C. These are real and imaginary current signals supplied by the CT block.
- Once configured click **ok**.

5. Configure the *BlkSlot* dialogs for slots C, D and E.

- Slots C,D and E will be configured to be time overcurrent blocks with each one representing a different phase. Double click on the slot C symbol and the *BlkSlot* dialog will appear.
- Enter an appropriate name for the slot e.g.. TOC phase A.
- Enter the class name as RelToc\*.
- In the classification field, ensure that only the boxes linear and Automatic, model will be created are checked
- Enter the following output signals under the variables field: yout.
- Enter the following input signals under the variables field: labs. This represents the RMS current signal for phase A supplied by the measurement block.
- Once configured click **OK**.
- Repeat the steps above for slot D and E. Name these slots TOC phase B and TOC phase C.

6. Configure the BlkSlot dialog for slot F.

- Slot F will be configured to be a Logic slot. Double click on the slot symbol and the BlkSlot dialog will appear.
- Enter an appropriate name for the slot e.g.. Logic.
- Enter the class name as RelLogic\*.
- In the classification field, ensure that only the boxes linear and Automatic, model will be created are checked
- Enter the following output signals under the variables field: yout.
- Enter the following input signals under the variables field: y1, y2, y3.
- Once configured click **OK**.

All block dialogs should now be configured.

7. Connect the blocks together using signals.

- Select the signal icon from the drawing toolbox located on the right side of the screen.
- Connect blocks by clicking on the output terminal of the first block then by clicking on the input terminal of the receiving block. If a route for the signal is required which is not direct, intermediate clicks may be used.

- If a signal is intended to be passed outside of the model then a signal should be terminated on the box which surrounds the frame. In this instance the output from the logic block will be passed outside of the model.
- Connect the blocks in the frame as illustrated in Figure 33.18.2.

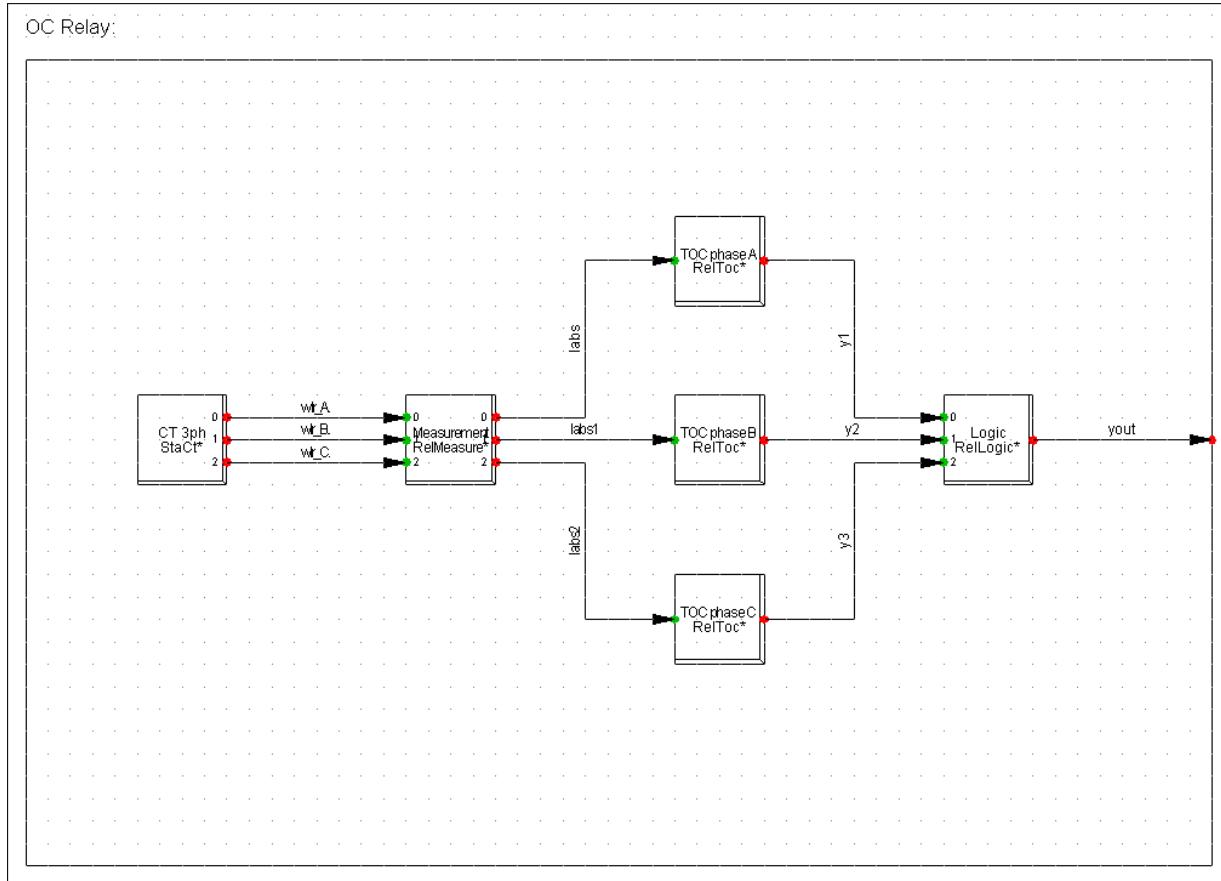


Figure 33.18.2: Signal route definition

#### 8. Rebuild the block definition

- Press the rebuild button on the local graphics window icon bar. Rebuilding the model will capture all the internal signals (signals defined between slots) and external signals (signals passed outside of the model) within the BlkDef model dialog. This concludes definition of the relay frame. The next step is to define a relay type.

#### 9. Create a relay type object

- Within the Data Manager go to the Equipment Type library folder in the project library and select the *New Object* icon
- In the dialog which appears select *Relay Type*.
- In the *TypRelay* dialog that appears give the relay type an appropriate name.
- In the relay definition field select the relay frame constructed earlier from the User Defined Models section of the project library.
- Select the category as overcurrent relay.

#### 10. Define the CT type

- The CT type can be selected by double clicking in the type column associated with the CT row.
- The desired CT should be selected from the Data Manager.

#### 11. Define the measurement type

- The measurement type can be selected by double clicking in the type column associated with the measurement row. For this example select the following options:
  - Select Type to 3ph RMS currents
  - Select nominal current to discrete with a value of 5.
  - Select measuring time to 0.001
  - Ensure no check boxes are selected.

#### 12. Define the TOC types

- The TOC types can be selected by double clicking in the type column associated with the rows of each of the three TOC slots. For this example select the following options for each TOC type:
  - Select IEC symbol  $I>t$  and Ansi symbol 51.
  - Select type to phase A, B or C current depending on the slot.
  - Select directional to none.
  - Select current range to range type: stepped, minimum: 0.5, maximum: 2 and step size: 0.25.
  - Check the characteristic includes pickup time box and set pickup time: 0.01s, Reset time: 0.04s and Reset Characteristic Configuration: Disabled.
  - Select an existing relay characteristic from another relay or create a new relay characteristic by creating a *TypChatoc* object.
  - On the Total clear curve tab ensure no boxes are checked.
  - On the blocking page, select consider blocking to disabled.
  - Select release blocking time range to range type: stepped, minimum: 0 maximum: 10000 and step size: 0.01.

#### 13. Define the Logic types

- The Logic type can be selected by double clicking in the type column associated with the logic row.
  - Select Breaker event to open.
  - Select number of inputs to 4.
  - Select number of block inputs to 4.
  - Select a logical OR operation.

This concludes definition of the relay type.

To use the relay type a relay must be created within the network. The relay type can then be selected, and the relay element parameters defined.

## 33.19 Appendix - other commonly used relay blocks

This section covers some of the other protection block not so far covered in the discussion throughout the chapter so far.

### 33.19.1 The frequency measurement block

The frequency measurement unit is used to calculate the electrical frequency for the given Measured Voltage. The Nominal Voltage is needed for per unit calculations. The Frequency Measurement Time defines the time used for calculating the frequency gradient.

### 33.19.2 The frequency block

The frequency block either trips on an absolute under-frequency (in Hz), or on a frequency gradient (in Hz/s). Which condition is used depends on the selected type. The type also defines the reset time, during which the defined frequency conditions must be present again for the relay to reset.

The time delay set in the relay element defines the time during which the defined frequency condition must be violated for the relay to trip.

### 33.19.3 The under-/overvoltage block

The under-/overvoltage relay type may define the block to trip on either

- One of the three phase line to line voltages
- One particular line to line voltage
- The ground voltage  $U_0$ .
- The positive sequence voltage  $U_1$
- The negative sequence voltage  $U_2$

The relay element allows only for setting of the pickup voltage and the time delay.

## 33.20 Relay block technical references

A number of technical references are available which are relevant for protection. Please refer to the [Protection Devices Library](#) for details on protection blocks and details regarding the CT and VT models.

# Chapter 34

## Arc-Flash Hazard Analysis

### 34.1 Introduction

This chapter describes the tools available in *PowerFactory* to perform arc-flash hazard analysis, including their technical background, descriptions of the Arc-Flash Hazard Analysis command and Arc-Flash Reports dialogs, and an example calculation. The Arc-Flash Hazard Analysis command (*ComArcflash*) can be accessed on the main toolbar under the *Protection and Arc-Flash Analysis* group by selecting the *Arc-Flash Analysis* icon .

---

**Note:** *DIGSILENT* accepts no responsibility for the use of the Arc-Flash Hazard Analysis command, or for the consequences of any actions taken based on the results. Use the Arc-Flash Hazard Analysis command at your own risk.

---

**Note:** By default, results are entered and displayed in SI units. To change to British Imperial units, on the main menu select *Edit* → *Project Data* → *Project Settings...*, change the page to *Formats and units*, and on the *Input Variables* tab, select the units to be “English-Transmission” or “English-Industry”, from the drop down list.

---

### 34.2 Arc-Flash Hazard Analysis Background

#### 34.2.1 General

Arc-flash hazard analysis calculations are performed to determine “...the arc-flash hazard distance and the incident energy to which employees could be exposed during their work on or near electrical equipment” [IEEE1584-2002][48]. One outcome of an arc-flash hazard analysis is to determine employee Personal Protective Equipment (PPE) requirements.

The Arc-Flash command builds on the existing short-circuit calculation capabilities in *PowerFactory*, and requires the following additional data, dependent on the *Calculation Method* selected:

- **IEEE-1584-2002** [48]: Conductor Gap, Distance Factor, Working Distance, and Enclosure Type.
- **IEEE-1584-2018** [49]: Conductor Gap, Working Distance, and Electrode configuration. For some Electrode configurations additionally Dimensions of enclosure might be needed

- **NFPA 70E** [53]: Working Distance and Enclosure Type.
- **DGUV 203-077** [43]: Conductor Gap, Distance Factor and Working Distance.
- **EPRI** [45]: Arc Length, Statistical Correction Factor and Working Distance.

### 34.2.2 Determination of arc duration and consideration of arcing current

Arc duration is used by all arc flash calculation methods in combination with arcing current to calculate incident energy and arc flash boundaries. Depending on the configuration of the command and the availability of protection licence modules, it can either be specified directly or determined through calculation based on protective device model tripping in response to short circuit calculation results. If it is determined through calculation then it is necessary for *PowerFactory*'s short circuit calculation to consider the partial short circuit currents seen by each separate protective device. For such cases it is necessary to determine the arcing current at the point of fault before the partial currents seen by the protective devices can be considered. Depending on the calculation method, the arcing current value used can vary as described below.

- **IEEE-1584-2002:** The bolted three phase, symmetrical, RMS fault current is calculated at the accessible location using *PowerFactory*'s short circuit calculation. At voltage levels of 15kV and below, an arcing current ( $I_a$ ) is calculated from this quantity using the equation in the standard.  $I_a$  is lower than the bolted short circuit current due to arc resistance. For voltage levels above 15kV  $I_a$  is specified to be equal to the bolted fault current. An arc duration (fault clearing time) is determined using  $I_a$ . A reduced arcing current ( $I_{a,min}$ ) is calculated from  $I_a$ . According to the standard the current corresponding with  $I_{a,min}$  should be 85% of  $I_a$ . However, the reduction factor specifiable in *PowerFactory* is user configurable. A second arc duration is determined from  $I_{a,min}$ . The incident energy arising from the two combinations of arcing current and arc duration are evaluated separately and the combination which results in the highest incident energy is selected.
- **IEEE-1584-2018:** The bolted three phase, symmetrical, RMS fault current is calculated at the accessible location using *PowerFactory*'s short circuit calculation. At voltage levels of 15kV and below, three intermediate average arcing currents are calculated from this quantity using the equation in the standard. A final arcing current ( $I_a$ ) is then calculated using equations from the standard that depend on one or more of the previously calculated intermediate average arcing currents as well as the open circuit voltage level of the fault location. In this case  $I_a$  is lower than the bolted short circuit current due to arc resistance. For voltage levels above 15kV  $I_a$  is specified to be equal to the bolted fault current. An arc duration (fault clearing time) is determined using  $I_a$ . A reduced arcing current ( $I_{a,min}$ ) considering an arcing current variation correction factor is calculated from  $I_a$  using another equation from the standard. A second arc duration is determined from  $I_{a,min}$ . The incident energy arising from the two combinations of arcing current and arc duration are evaluated separately and the combination which results in the highest incident energy is selected.
- **NFPA 70E AC:** The bolted three phase, symmetrical, RMS fault current is calculated at the accessible location using *PowerFactory*'s short circuit calculation and the arcing current used to evaluate the arc duration is directly equated with this value.
- **DGUV 203-077 AC:** The *minimum* bolted three phase, symmetrical, RMS fault current is calculated at the accessible location using *PowerFactory*'s short circuit calculation. An arcing current ( $I_a$ ) is calculated from this quantity using the equation in the standard.  $I_a$  is lower than the bolted short circuit current due to arc resistance. An arc duration (fault clearing time) is determined using  $I_a$ . Note that the arc flash power is calculated from the *maximum* bolted three phase, symmetrical, RMS fault current, calculated using *PowerFactory*'s short circuit calculation and the incident energy is calculated by multiplying the arc flash power by the arc duration.
- **EPRI:** A single phase, phase A to ground, zero impedance, symmetrical, RMS fault current is calculated at the accessible location using *PowerFactory*'s short circuit calculation. An arcing current ( $I_a$ ) and corresponding arc resistance can be calculated from this quantity using the equations in [45].  $I_a$  is lower than the bolted short circuit current due to arc resistance. An arc duration (fault clearing time) is determined using  $I_a$ .

- **NFPA 70E DC:** The bolted DC fault current is calculated at the accessible location using *PowerFactory*'s short circuit calculation and the arcing current used to evaluate the arc duration is directly equated with this value.
- **DGUV 203-077 DC:** The bolted DC fault current is calculated at the accessible location using *PowerFactory*'s short circuit calculation. An iterative calculation to determine the arcing current  $I_a$  is started using the equations from the standard. The initial estimate for the arcing current is taken as half the previously calculated value of bolted DC fault current.  $I_a$  is lower than the bolted short circuit current due to arc resistance. An arc duration (fault clearing time) is determined using  $I_a$ .

For the cases where the arcing current does not directly equate with the bolted (zero fault impedance) short circuit current *PowerFactory* calculates the arcing current and corresponding partial arcing currents using a short circuit calculation where the fault resistance (representing the arc resistance) has been tuned so as to produce the required arcing current at the accessible location. By doing this, and if necessary, *PowerFactory* is then able to calculate the partial currents seen by and the corresponding tripping times of all protective devices responsible for clearing the fault in order to determine the arc duration.

### 34.2.3 Input Data

#### 34.2.3.1 Guidance on input data values

The IEEE-1584 standard provides guidance on the selection of the Conductor Gap and Distance Factor. Table [34.2.1](#) shows the recommended values from the standard.

System Voltage [kV]	Equipment type	Typical gap between conductors [mm]	Distance x factor
0.208-1	<i>Open air</i>	10-40	2.000
	<i>Switchgear</i>	32	1.473
	<i>MCC and panels</i>	25	1.641
	<i>Cable</i>	13	2.000
>1-5	<i>Open air</i>	102	2.000
	<i>Switchgear</i>	13-102	0.973
	<i>Cable</i>	13	2.000
>5-15	<i>Open air</i>	13-153	2.000
	<i>Switchgear</i>	153	0.973
	<i>Cable</i>	13	2.000

Table 34.2.1: Factors for equipment and voltage classes [IEEE1584-2002][48]

Enclosure type	Equipment class	Default bus gaps [mm]	Enclosure size (H x W x D)
1	15 kV switchgear	152	1143 mm x 762 mm x 762 mm
2	15 kV MCC	152	914.4 mm x 914.4 mm x 914.4 mm
3	5 kV switchgear	104	914.4 mm x 914.4 mm x 914.4 mm
4	5 kV switchgear	104	1143 mm x 762 mm x 762 mm
5	5 kV MCC	104	660.4 mm x 660.4 mm x 660.4 mm
6	Low-voltage switchgear	32	508 mm x 508 mm x 508 mm
7	Shallow low-voltage MCCs and panelboards	25	355.6 mm x 304.8 mm x <= 203.2 mm
8	Deep low-voltage MCCs and panelboards	25	355.6 mm x 304.8 mm x > 203.2 mm
7 or 8	Cable junction box	13	355.6 mm x 304.8 mm x <= 203.2 mm 355.6 mm x 304.8 mm x > 203.2 mm

Table 34.2.2: Enclosure sizes for [IEEE1584-2018] arc-flash model [49]

Figure 34.2.1 shows the terminal element dialog in *PowerFactory* where parameters required for the specific Arc-Flash Hazard Analysis Calculation method can be entered.

The terminals at which arc flash calculations are to be carried out are defined by designating those terminals as accessible locations. An *Accessible Location* checkbox is available on the *General* tab of the *Arc-Flash Analysis* page of each terminal (ElmTerm) object dialog. If the checkbox is selected, the user may then enter input parameters relevant for the specific arc-flash calculation method to be used, on the tab corresponding with the method.

If the terminal resides within a substation or within a bay, the *Equipment Data* field can be set to either *Local Values*, *From Substation*, *From Bay*. When *From Substation* or *From Bay* is selected, a pointer to the relevant substation or bay is shown in the dialog and the equipment data is taken from the selected object instead of from the terminal.

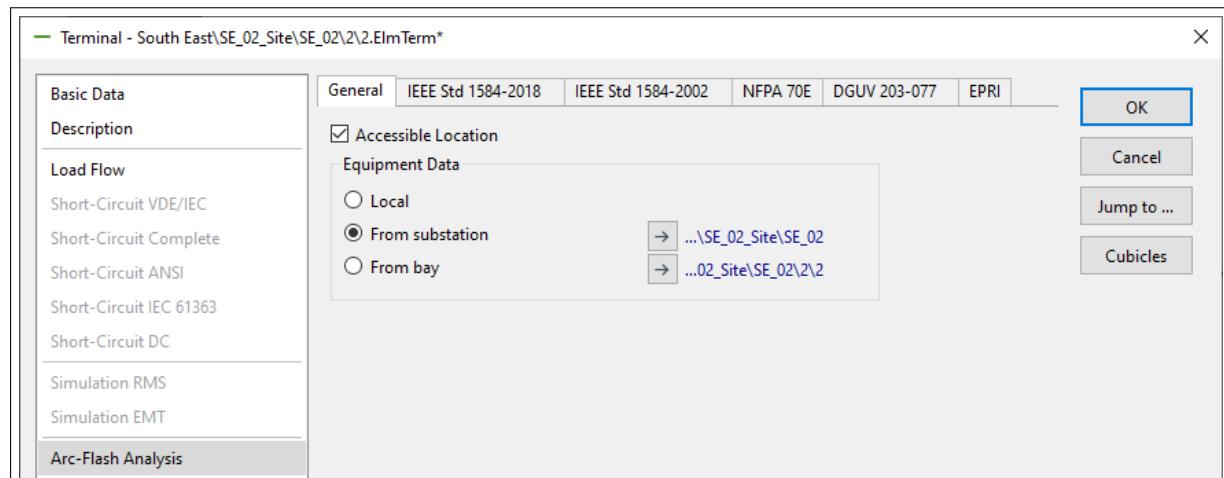


Figure 34.2.1: Arc-flash data required for terminals

Additional input data required for calculation of *Fault Clearing Times/arcing time* depends on the calculation approach with regards to whether fixed times or calculated times based on protection device modelling is to be used. This is discussed later in this chapter see Section 34.3.1.2.

### 34.2.4 Arc Models and Ranges of Calculation Models

Depending on the calculation method used different arc models are applied. Furthermore, sometimes the arc model used can vary depending on the rated voltage of the accessible location. In addition to this the applicability of arc models are often limited by other factors related to the test conditions under which the models were empirically derived. The following section aims to capture some of the details regarding the applicable ranges of the different calculation models. For detailed information regarding the range of applicability of the methods please refer to the standards and other source documentation directly.

#### 34.2.4.1 IEEE-1584-2002

The arc models and application range of the IEEE-1584-2002 method as derived from the standard is as follows:

- For accessible locations rated with three phase line to line voltage levels in the range 208 V - 15 kV, the empirically derived arc model specified in the standard applies.
- For accessible locations rated with three phase voltage levels higher than 15 kV *PowerFactory* uses the theoretically derived Lee method for the arc model. This method is expected to produce conservative results.

---

**Note:** For more accurate results at higher voltage levels the EPRI method arc model (see Section 34.2.4.5) may be considered. However, care should be taken to ensure that the test configurations considered in the derivation of the EPRI method are representative for the calculation case being considered.

- System frequency must be 50 Hz or 60 Hz.
- RMS symmetrical bolted fault current should be between 700 A and 106 kA.
- All system grounding methods are supported.
- only conductor gaps between 13 mm and 152 mm are within the range of the model.
- Intended for three phase faults.

#### 34.2.4.2 IEEE-1584-2018

The arc models and application range of the IEEE-1584-2018 method as derived from the standard is as follows:

- For accessible locations rated with three phase line to line voltage levels in the range 208 V - 15 kV, the empirically derived arc model specified in the standard applies.
- For accessible locations rated with three phase voltage levels higher than 15 kV *PowerFactory* uses the theoretically derived Lee method for the arc model. This method is expected to produce conservative results.

---

**Note:** For more accurate results at higher voltage levels the EPRI method arc model (see Section 34.2.4.5) may be considered. However, care should be taken to ensure that the test configurations considered in the derivation of the EPRI method are representative for the calculation case being considered.

- System frequency must be 50 Hz or 60 Hz.
- For voltage levels of 208 V to 600 V, RMS symmetrical bolted fault current should be between 700 A and 106 kA.
- For voltage levels of 601 V to 15 kV, RMS symmetrical bolted fault current should be between 200 A and 65 kA.
- For voltage levels of 208 V to 600 V, gaps between conductors should be between 6.35 mm and 76.2 mm.
- For voltage levels of 601 V to 15 kV, gaps between conductors should be between 19.05 mm and 254 mm.
- The model supports working distances of greater than or equal to 305mm. (see [49] for specific details.)
- No limit on the fault clearing time. (see [49] for specific details.)
- Dimensions of enclosures tested (with open front end) for open circuit voltage of 600 V as per Enclosure type 6 in Table 34.2.2.
- Dimensions of enclosures tested (with open front end) for open circuit voltage of 2.7 kV as per Enclosure type 5 in Table 34.2.2.
- Dimensions of enclosures tested (with open front end) for open circuit voltage of 15 kV as per Enclosure type 2 in Table 34.2.2.
- Maximum height or width of enclosure = 1244.6mm
- Maximum opening area of enclosure =  $1.549m^2$
- Minimum width of the enclosure is greater than four times the gap between conductors (electrodes)
- electrode configurations (VCB, VCBB, HCB, VOA, HOA), (see [49] for specific details.)

#### 34.2.4.3 NFPA 70E AC

The arc models and application range of the NFPA 70E AC method as derived from the standard is as follows:

- For accessible locations rated with three phase line to line voltage levels 600V and below, the calculation method derived from the Doughty Neal Paper as described in citeNFPA70E is used.
- The Doughty Nealpaper method applies to 3 phase arcs with short circuit currents between 15kA and 50kA.
- Typical working distances for Low voltage (600V and below) motor control centre and panelboards = 455mm
- Typical working distances for Low voltage (600V and below) switchgear = 610mm
- For accessible locations rated with three phase line to line voltage levels greater than 600V, the Ralph Lee calculation method as described in [53] is used.
- Typical working distances for medium voltage (above 600V) switchgear = 910mm

#### 34.2.4.4 DGUV 203-077 AC

The arc models and application range of the DGUV 203-077 AC method as derived from the standard is as follows:

- The arc model as defined by the equations of the standard is used.
- The method is applicable for accessible locations having rated line to line voltage levels of between 50 V and 110 kV.
- The arcing current is assumed to be equal to half the RMS symmetrical bolted fault current low voltages whilst at medium and high voltages it is considered as equal to the RMS symmetrical bolted fault current.

#### 34.2.4.5 EPRI method

For the EPRI calculation method, an arc model is used as defined in [45] where it is noted that the model has been derived from arc tests that satisfy the test setup parameters listed in Table 34.2.3. The results of the tests themselves are also reported on in [45].

The method as implemented in *PowerFactory* is intended to be used with single phase to ground, arc flash events occurring in open air and involving long arcs covering a gap length of 1 ft to 5 ft (0.31m to 1.5m) as is more commonly expected to occur in open air high voltage applications. However, the test parameters in [45] do not explicitly specify a voltage range for the arc model. For this reason the method has not been limited for application to a specific range of nominal voltage levels corresponding with the rating of the accessible location(s). If the input parameters assigned to the accessible location(s) are outside of the ranges defined by Table 34.2.3, *PowerFactory* will execute the calculation but will issue warning messages to the output window indicating that the relevant input parameters are outside of the tested range of the EPRI equations.

---

**Note:** To comply with the recommendations of [45] the EPRI method should not be used with gap lengths of less than 1 ft (0.31 m).

---

Parameter	Description
Gap Orientation	Vertical
Gap Length (ft)	1 - 5
Gap Length (m)	0.31 - 1.5
Current (kA rms)	8 - 40
Duration (s)	0.033 - 0.2
Duration (cycles)	2 - 12

Table 34.2.3: EPRI method Arc Test Parameters Considered in [45]

#### 34.2.4.6 NFPA 70E DC

The application range of the NFPA 70E DC method as derived from the standard is as follows:

- For accessible locations rated with DC voltage levels up to 1000 V DC, the calculation method derived from the Doan Paper as described in [53] is used.

#### 34.2.4.7 DGUV 203-077 DC

The application range of the DGUV 203-077 DC method as derived from the standard is as follows:

- The method described in [43] is used which is applicable for accessible locations having DC voltage levels of between 50 V DC and 1.5 kV DC.

### 34.3 Arc-Flash Hazard Analysis Calculation Options

#### 34.3.1 Arc-Flash Hazard Analysis Basic Options Page

##### 34.3.1.1 Calculation

**System type:** select the network type associated with the arc flash fault locations to be considered in the calculation: *AC* or *DC*. Depending on the system type, different arc flash calculation methods are available

**Method:** one of the following may be selected:

- according to IEEE-1584 published 2002 (AC) [48]
- according to IEEE-1584 published 2018 (AC) [49]
- according to NFPA 70E (AC or DC) [53]
- according to DGUV 203-077 (AC or DC) [43]
- according to EPRI (AC) [45]

Fault Location At:

- **User Selection:** Selection of either a single location or a pre-defined set of locations. Locations can be selected in single line diagrams or in tabular views by right clicking a single terminal or multi-selection of terminals and choosing *Calculation* → *Arc-Flash Analysis*. This option is only available, if at least one of the busbars in the selection is configured as an *Accessible Location* (see Section 34.2.3).
- **All Accessible Locations:** Locations are all terminals where *Accessible Location* is selected on the *Arc-Flash Analysis* page of the element dialog.

##### 34.3.1.2 Arc Duration

One of the following may be selected:

**Use Fixed Times:** In this case, protection device models (and associated protection licence modules) are not required by the calculation. The corresponding parameter *Get Time from* can be set to either:

- **Global:** An *Arc Duration* parameter can be specified in the command which is then used for all accessible locations.
- **Local:** The fault clearing times for each accessible location are determined from the corresponding *Arc-Flash Analysis* page of the relevant switch elements (*ElmCoup* and *StaSwitch*), where the *Switch Type* has been set to “Circuit Breaker” on the *Basic Data* page. When this option is selected an additional *Maximum Time* parameter may be set in the command dialog which is used to specify the maximum fault clearing time to be used by the arc-flash command.

**Use Relay Tripping:** In this case, the tripping time is based upon relay characteristics entered in protection models which have been defined in the network model. This option can only be used where the user's licence includes Time-overcurrent or distance protection licence modules.

---

**Note:** Only devices where the *Status* parameter on the device *Description* page is set to "Approved") will be considered by the calculation.

---

The Arc-Flash Analysis command performs incident energy calculations using the evaluated tripping time where the tripping time is based on the evaluated partial arcing currents (see Section 34.2.2 for more details). To determine the fault clearing time, *Get Time from* can be set to either:

- **Initial:** In this case the Arc-Flash command carries out a standard (not trace) short circuit calculation to determine the fault clearing time based on the longest fault clearing time of any element connected to the faulted terminal. For example, if two parallel lines are supplying a faulted terminal, and the first line has a fault clearing time of 1 s, and the second line has a fault clearing time of 2 s (where both clearing times are based on the *Initial* fault current) the Arc-Flash command will take 2 s as the fault clearing time.
- **Iteration:** In this case the Arc-Flash command determines the fault clearing time from a Short-Circuit Trace calculation. For example, assume that two parallel lines are supplying a faulted terminal, and the first line has a fault clearing time of 1 s. Then, after the first line is cleared, the second line sees a higher fault current, and subsequently clears the fault at 1.5 s. The Arc-Flash command takes 1.5 s as the fault clearing time.

---

**Note:** When there are multiple sources of fault current at a faulted terminal with different fault clearing times, *PowerFactory* takes the maximum clearance time of the connecting branch for all branches.

---

**Maximum Time:** This parameter is used to specify the maximum fault clearing time to be used by the Arc-Flash command. It applies for cases where evaluation of the relay model tripping times produces long tripping times. In such cases the value assigned to this parameter is used to determine the arc duration and incident energy etc. instead of the calculated tripping time.

---

**Note:** The option *Use Relay Tripping: Iteration* changes the short-circuit method automatically to *complete*. *PowerFactory* uses the *Short-Circuit Trace* function to determine the clearing time and the function is only available in combination with the *complete* method. See Section 33.16 for further information about the Short-Circuit Trace function.

---

---

**Note:** If *System type: DC* is selected, the option *Use Relay Tripping: Iteration* is not available.

---

### 34.3.1.3 Short-Circuit Calculation

Pointer to the *Short-Circuit Calculation* command. The short-circuit method will not be changed automatically from AC to DC, since multiple methods are available in *PowerFactory*. The user should make sure that the correct short-circuit method is selected in the short-circuit command before executing the arc-flash command. Otherwise an error message will be displayed in the output window.

#### 34.3.1.4 Show Output

If this is selected, the Arc-Flash Reports (*ComArcreport*) command will be executed when the calculation is complete. This will generate a tabular report of results and/or an arc-flash label report according to the options selected in the command. Note that the reports command can be configured via the  icon. See Section [34.4.2](#) for more details.

### 34.3.2 Arc-Flash Hazard Analysis Advanced Options Page

#### 34.3.2.1 Arc-Flash Calculation Options

The parameters that are available on this page depend on the calculation method options which have been selected on the *Basic Options* page.

##### Options common to IEEE-1584, NFPA 70E, and EPRI:

- **Energy at Flash-Protection Boundary:** This defines the energy at which the Arc-Flash protection boundary is calculated.

**Note:** If IEEE 1584:2018 is selected this parameter only influences the results for accessible locations with a rated voltage of more than 15kV. i.e. at accessible locations where an arc model is based on the Lee equations applies. Below this voltage the specific equations from the standard are used to determine an Arc Flash Boundary that corresponds with an energy density of  $5J/cm^2$ .

---

##### Specific options for IEEE-1584:2002:

- **Arcing current variation:** This option defines the percentage by which the expected arcing current is reduced for the reduced arcing current calculation (see [34.2.2](#)).
- **Apply arcing current variation to nodes > 1kV:** By selecting this parameter it is possible to consider arcing current variation for medium voltage nodes rated from >1kV up to 15kV.

**Note:** In accordance with the standard *PowerFactory* does not consider reduced arcing currents due to arcing current variation for accessible locations rated above 15kV.

---

- **Iterative Energy Calculation:** This option is only available if the *Arc Duration* option is set to *Use Relay Tripping* and the *Get Time from* option is set to *Iteration*, on the *Basic Options* page. When selected, *PowerFactory* uses the arcing current calculated at each step of the arc-flash calculation's short circuit trace for the determination of the final incident energy calculation. For each step of the short circuit trace the arc resistance calculated during the initial calculation is held at its initial value and the redistribution of short circuit currents resulting from the updated network topology is considered. This option is not available with the 2018 version of the calculation since the modified calculation equations do not allow for it.

##### for NFPA 70E with DC system type:

- **Use detailed arc model:** This option changes the arc model used in order to determine the incident energy. With the checkbox off the *Maximum Power Method* described in [53] is used. With the checkbox selected the *Stokes and Oppenlander* model described in [46] is used.

##### Specific options for DGUV 203-077:

- **Min. arc energy:** This option sets a minimum threshold where no PPE category is determined (i.e. PPE category zero).
- **Use worst-case assumption:** This option changes the determination of the current limiting factor and relative arc power so that the simplified equations are used in accordance with [43].

### 34.3.2.2 PPE-Ratings

One of the following options can be selected for the PPE-Ratings:

- **Acc. to NFPA 70E**[53]: In this case default values from the standard are used. These ratings can only be used for the IEEE-1584, NFPA 70E and EPRI calculation methods.
- **Acc. to DGUV 203-077**[43]: In this case default values from the standard are used. These ratings can only be used for the DGUV 203-077 standard.
- **User-Defined**: In this case user-defined *Category* values can be entered in the *PPE-Categories* table after inserting or appending rows. Note that the values should be entered in ascending order and that the table changes according to the standard selected on the “Basic Options” page.

## 34.4 Arc-Flash Hazard Analysis Results

### 34.4.1 Viewing Results in the Single Line Graphic

#### Diagram Colouring

Terminals can be coloured according to the calculated PPE category, Incident Energy and the calculated Loading of Thermal/Peak Short-Circuit Current. To set the diagram colouring mode, select the *Diagram Colouring* icon, and then under 3. *Other*, select *Results*, and the desired colouring mode.

The use of colouring in network diagrams is described in Section 10.3.10.1, and more detail about the use of colours and colour palettes can be found in Section 4.7.8.

#### Result Boxes

To show the default set of Arc-Flash result parameters (Arcing Current, Arcing Time, Working Distance) in the single line diagram result boxes, right-click on a terminal result box and select *Format for Short Circuit Nodes* → *Arcing Current, Time, Distance*. Energy and PPE results can also be displayed.

### 34.4.2 Arc-Flash Reports

The Arc-Flash Reports (*ComArcreport*) command dialog is accessed via the  icon. It can be used both to show calculation results in tabular format and to generate arc-flash labels. The command has a pointer to the *Result File* storing the calculation results.

#### 34.4.2.1 Create tabular report

If the user selects this option, a tabular report of pre-defined result variables will be generated. Further options are available within the report itself. For each location shown in the report, three options are available to further analyse the results. The options can be activated by either selecting a location (first column *Accessible Location*) and clicking on the relevant button, or via a right click on the location cell and by then selecting the relevant option in the context menu.

**Show intermediate results:** This option brings up a table with further results.

- For the IEEE-1584 methods both the full and reduced arc currents are shown, highlighting the worst-case which is displayed in the overview table.
- For the DGUV 203-077 method this options shows more result variables such as the current limiting factor or the relative arc power.

**Calculate short-circuit:** This option executes the short-circuit command at the selected fault location. A specific fault resistance is applied in order to reduce the short circuit current magnitude to a value which corresponds with the calculated value of the arcing current.

**Calculate short-circuit trace:** This option executes the short-circuit trace calculation at the selected fault location. A specific fault resistance is applied in order to reduce the short circuit current magnitude to a value which corresponds with the calculated value of the arcing current.

---

**Note:** When using the options *Calculate short-circuit* or *Calculate short-circuit trace*, the user must make sure that the commands are set correctly. The report option only changes the fault location and resistance. Any other settings, e.g. fault type or method, need to be set manually.

---

When the arcing current is calculated using the report options, tripping times can be easily analysed using the *Time-Overcurrent* plot or the overview table of the short-circuit trace function.

The tabular report can be exported in HTML or XLSX format, by using the icons at the top right-hand corner of the report.

---

**Note:** If the incident energy exceeds the incident energy in the maximum PPE category, the result is "N/A".

---

### 34.4.2.2 Create report document (for arc-flash labels)

If this option is selected, the standard reporting functionality can be used to generate arc-flash labels. A list of available reports will be presented, and the required reports can be selected. When the command is executed, reports are presented as PDF documents in the active graphic window, using *PowerFactory*'s inbuilt PDF viewer. From there, the documents can be exported, using the  icon.

It should be noted that the arc-flash reports which are viewed internally in *PowerFactory* are stored in the desktop of the study case, even after they have been closed, until they are actively deleted by the user.

Further options are available for users who want to customise the labels or export them in a different format:

#### Customised labels

By default, the arc-flash labels are generated using a standard report template. This determines the appearance of the label and the quantities shown. If the user wishes to modify this in any way, it can be done by creating an entirely new template, or - more likely - by making a copy of the standard template and modifying it. The new or changed template can then be stored in the project library or custom library (if available).

The general process for creating and modifying templates is described in Chapter 19: Reporting and Visualising Results, Section 19.5, but for arc-flash labels, two options are available:

- In the Arc-Flash Report dialog, click on the **Create new report...** button to create a new report template, which will be stored in the project library and can then be selected for use.
- After running Arc-Flash Analysis, open the main Report Generation command ( in the main toolbar) and from here prepare a new report template or copy the existing report template into the project or custom library and modify it as required. Then, when the Arc-Flash Reports command dialog is opened, and *Create report document* selected, the new report will be offered as an option.

### Creating labels in other formats

When labels are created using the dedicated Arc-Flash Reports (*ComArcreport*) command, they are automatically created as PDF documents. But if the user requires the labels in another format such as MS Word, this is also possible.

First, the Arc-Flash Analysis must be executed to ensure that this is the most recently-executed command. The main Report Generation command dialog is then opened, using the  icon in the main toolbar.

The report selection(s) are made on the Basic Options page; then, on the Export page, the *Export generated documents* option is checked, the target location specified, and the report format and any associated settings selected.

Then the command can be executed. Chapter 19: Reporting and Visualising Results, Section 19.5 gives more information about this general reporting command.

## 34.5 Example Arc-Flash Hazard Analysis Calculation

Consider the example network shown in Figure 34.5.1, where there are two parallel lines connected to a terminal “Terminal”. For this example, the two lines have different protection characteristics, as shown in Figure 34.5.2.

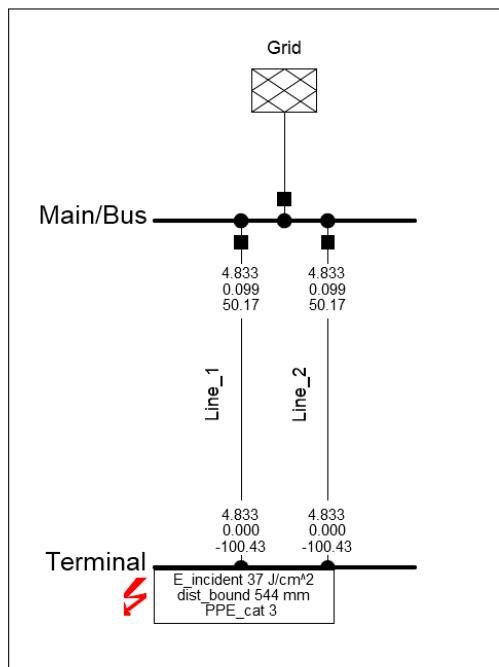


Figure 34.5.1: Example network single line graphic

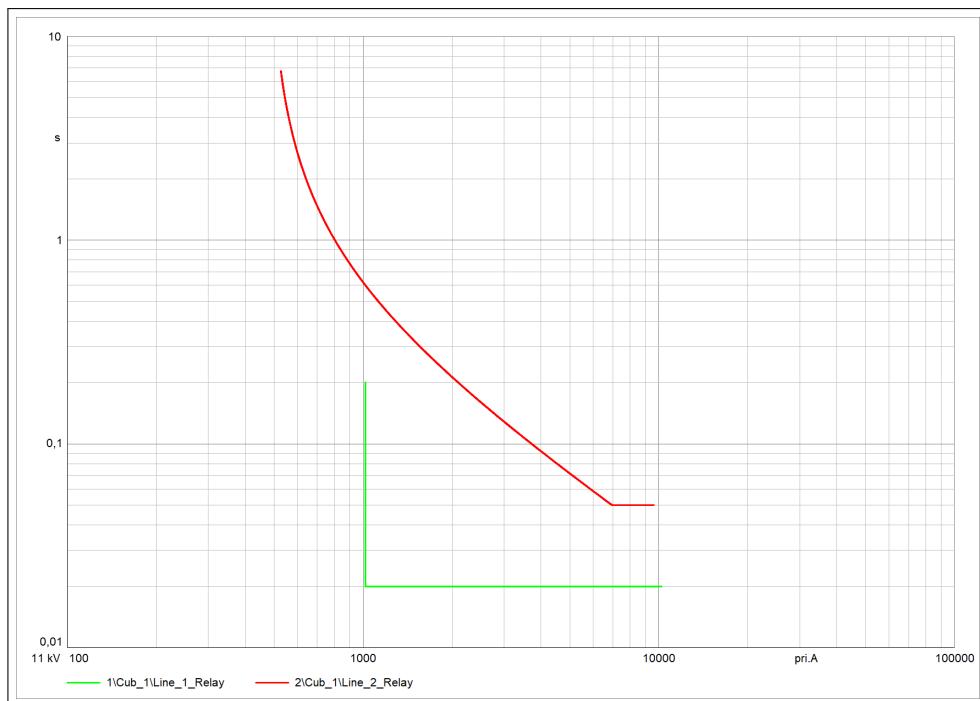


Figure 34.5.2: Protection characteristics

Arc-flash calculations are carried out using each method as follows:

- With *Use Fixed Times* and *Get Time from Global* selected, and with a total fault clearing time of 0.12 s, the key results are as follows:
  - Incident Energy:  $58 \text{ J/cm}^2$ .
  - Boundary Distance: 683 mm.
  - PPE Category: 3.
- With *Use Fixed Times* and *Get Time from Local* selected, and with a total fault clearing time of 0.10 s, the key results are as follows:
  - Incident Energy:  $49 \text{ J/cm}^2$ .
  - Boundary Distance: 624 mm.
  - PPE Category: 3.
- With *Use Relay Tripping* and *Get Time from Initial* selected (Figure 34.5.1), the key results are as follows:
  - Incident Energy:  $37 \text{ J/cm}^2$ .
  - Boundary Distance: 544 mm.
  - PPE Category: 3.
- With *Use Relay Tripping* and *Get Time from Iteration* selected, the key results are as follows:
  - Incident Energy:  $28 \text{ J/cm}^2$ .
  - Boundary Distance: 470 mm.
  - PPE Category: 2.

Of particular interest is the difference in results for the case where *Get Time from Initial* is selected, versus *Get Time from Iteration*. The former case gives conservative results (in this example), whilst in the latter case, the fault clearing time is faster due to recalculation of the fault current (as discussed in Section 34.3.1), and thus the calculated PPE requirement is lower.

A label is produced for “Terminal” (as described in 34.4), for the method where *Relay Tripping*, and *Get Time from Initial* is selected. The resultant label is shown in Figure 34.5.3.



Figure 34.5.3: Example arc-flash warning label

# Chapter 35

## Cable Analysis

The cable analysis toolbox consists of two calculation tools:

- Cable Sizing (⌚) command (*ComCabsiz*e), described in Section 35.1
- Cable Ampacity (🕒) calculation (*ComAmpacity*), described in Section 35.2

The cable sizing command uses static network calculations to verify the compliance of the configured cables in the network model against typical constraints. Further, should the constraints not be met or if an optimised selection of cable types is required it can be used to recommend cable types to meet the specified constraints.

The ampacity calculation tool is used to calculate the cable ampacity of cable models based on cable system objects (*TypCabsys*), given information on their geometry, construction, installation methods and proximity to adjacent cables. It can also be used to calculate an adiabatic short circuit current (short-time) rating of the cable.

In addition to the tools, additional features are built into various model classes which may be used throughout the network model and these features are then used by the cable analysis tools to calculate results. Specifically, features are built into line elements (*ElmLne*), line sub-section elements (*ElmLnesec*), line types (*TypLne*), single core cable types (*TypCab*) and multi core/pipe-type cable types (*TypCabmult*). For the purpose of ampacity calculation, cable system objects (*TypCabsys*) can also be embedded in a cable layout (*ElmCablay*), which is particularly important for the calculation because of the different thermal transitions.

For all these model classes the data required to parameterise the cable analysis features can be found on the *Cable Sizing* and *Cable Ampacity* page of their respective object dialogs.

---

**Note:** Throughout this chapter it may be noticeable that cables are sometimes identified as line objects. It should be made clear that the object classes *line element* (*ElmLne*), Line sub-section elements (*ElmLnesec*) and *Line Type* (*TypLne*) can also be used to define cables despite their potentially misleading names. A line type (*TypLne*) object can be identified as relating to a cable via its *Cable / OHL* parameter and a line element or line sub-section element (*ElmLne* or *ElmLnesec*) object referencing one of the aforementioned line types should therefore also be considered to represent a cable.

---

The features associated with cable analysis in *PowerFactory* shall be described in the following sections.

## 35.1 Cable Sizing

The Cable Sizing command (⌚) is used to size cables in the network to comply with user-defined constraints. The main functionalities and applications are:

- Planning a network from scratch: Assign new cable types to existing line elements without a type
- Verify existing cable types against user-defined constraints
- Recommend optimised cable types and determine the necessary cross section to fulfil all constraints
- Simplified calculation of the derating factor considering the installation conditions according to International Standards

It is important to note that the Cable Sizing command only works with the general Line Type (*TypLne*). All available settings for the cable models are described in Section 35.1.1.

All calculation settings are described in Section 35.1.2.

The available reports are described in Section 35.1.3.

### 35.1.1 Cable Models for Cable Sizing

In this chapter the available settings in the line element (*ElmLne*, *ElmLnesec*) and line type (*TypLne*) are described for the Cable Sizing.

#### 35.1.1.1 Line Type Parameters

The parameters defined on the *Cable Sizing* page of the Line Type *TypLne* as shown in Figure 35.1.1 define the characteristics of a cable which are usually given in datasheets and which are used during the cable sizing process.

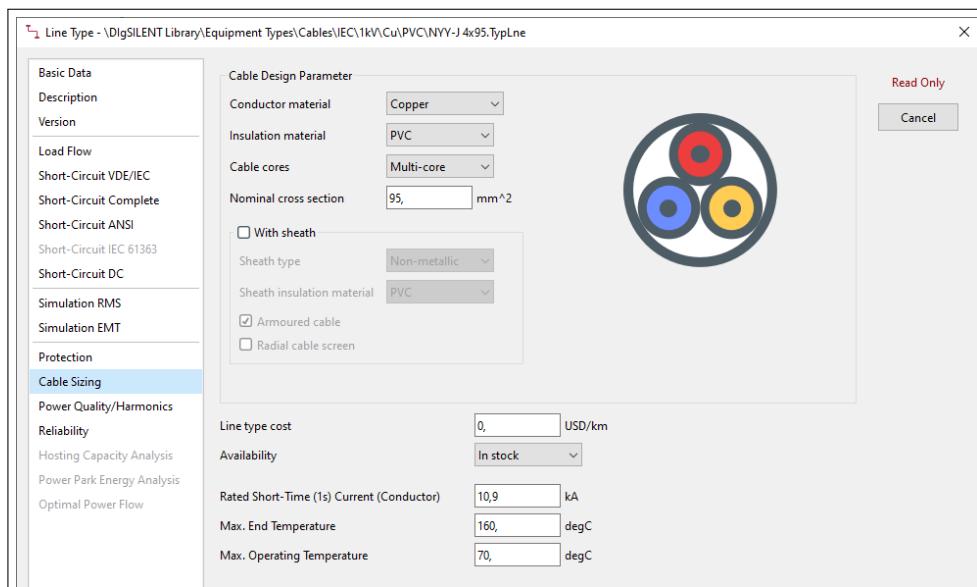


Figure 35.1.1: Line Type parameters for the Cable Sizing

For the calculation of the derating factor and the cable selection according to International Standards, the parameters that define the structure of the cable are used:

- *Conductor Material*: Select either Copper, Aluminium, Aldrey (AlMgSi), Aluminium-Steel or Aldrey-Steel.
- *Insulation Material*: Select either PVC, XLPE, Mineral, Paper, or EPR (Paper is valid only for NF C13-200, and Mineral is valid only for 0.5 kV and 0.75 kV systems and copper conductors).
- *Cable Cores*: Select either multi-core or single-core.
- *With Sheath*: Select if the cable has a sheath cover. If mineral insulation is selected and this frame is not checked, it is considered that the cable is bare with a metallic sheath.
  - *Sheath Type*: Select metallic or non-metallic.
  - *Sheath Insulation*: Select either PVC, XLPE, or EPR.
  - *Armoured Cable*: If checked, an armoured cable construction will be considered, otherwise a non-armoured cable construction is considered.
  - *Radial Cable Screen*: If checked then each conductor has its own screening. This is valid only for multi-core cables, since single-core cables always have radial screening.
  - *Exposed to touch*: For copper conductors with mineral insulation, select if the cable is exposed to touch.

The *Nominal Cross Section* is for information only, but can be used to colour the lines in the single line diagram according to the nominal cross section.

For the sizing of cables the parameters defining the current carrying capacity are used, where the rated current is taken from the “Basic Data” page:

- *Rated Short-Time (1s) Current (Conductor)*: Used to determine the permissible loading of the cable during short-circuits. Note, that the rated current is adjusted depending on the short-circuit duration.
- *Max. End Temperature*: Is the maximum temperature of the conductor for the short-circuit calculation.
- *Max. Operating Temperature*: Is the maximum operating temperature of the conductor for the load flow calculation.

The optimisation algorithm of the cable sizing also considers the costs of installing new cables. The material cost is defined in the cable type:

- *Line type cost*: The material cost of the cable per kilometre.
- *Availability*: With the options in stock and remainder the cable type can be used during the sizing process while a cable type which is not available can't be installed.

---

**Note:** When multiple cable types fulfil all requirements, then the cable types with the lowest costs are installed. Therefore, it is important to define line type costs.

---

### 35.1.1.2 Line Element Parameters

Line element parameters relevant to the Cable Sizing command are defined on the corresponding page of line element objects (*ElmLne*, *ElmLnsec*) as shown in Figure 35.1.2.

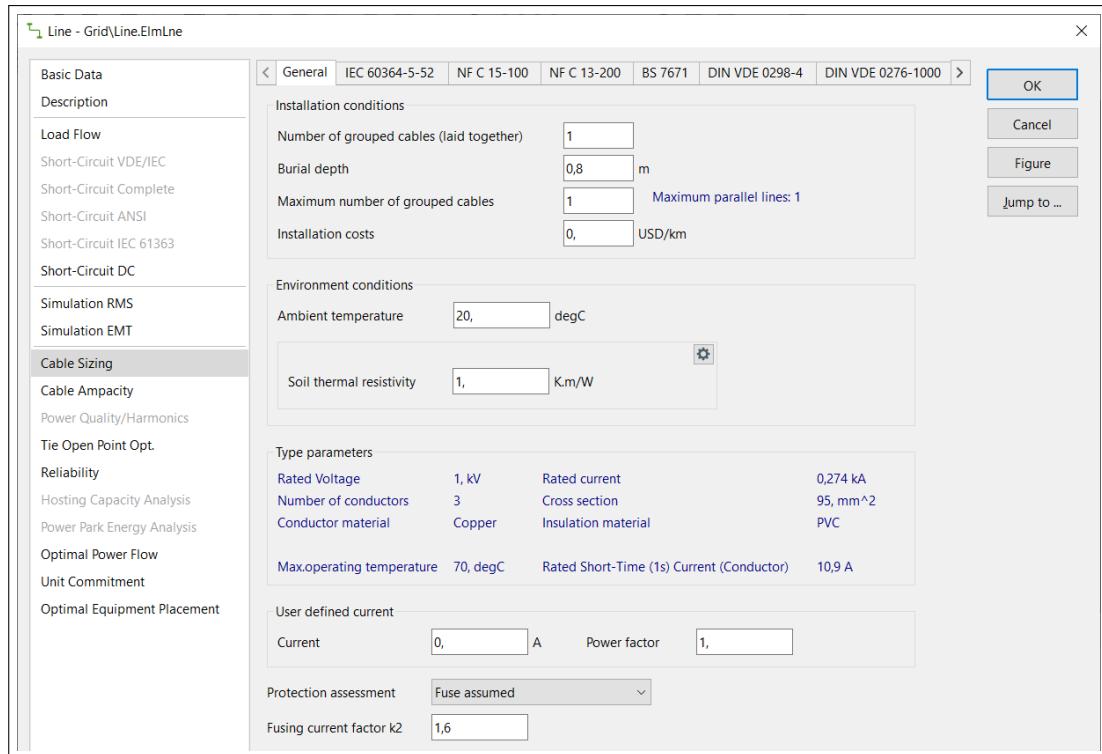


Figure 35.1.2: Line Element parameters for the Cable Sizing

The parameters on the “General” tab are used for the following purposes:

### Installation conditions

General installation conditions for the cable can be defined here which are considered in the derating factor calculation as well as the optimisation process when parallel cables are installed.

- **Number of grouped cables (laid together) (npara):** This is the number of cables which are (currently) laid together in the same trench. This does not only include this line (*ElmLne*) but possibly also other line elements in the network and is used for the derating factor calculation.
- **Burial depth (if\_depth):** Setting for the burial depth, only editable for laying condition in ground.
- **Maximum number of grouped cables (maxPara):** This describes the maximum amount of cables that can be placed within the same trench. The limit is used when new parallel cables shall be installed.
- **Maximum parallel lines (cmaxPara):** This is a calculation parameter which describes the maximum number of parallel lines of this particular line element (*ElmLne*).

$$cmaxPara = maxPara - (npara - nlnum) \quad (35.1)$$

where

- **cmaxPara** is the maximum number of parallel lines of this line element
- **maxPara** is the maximum number of grouped cables in the considered trench
- **npara** is the current number of grouped cables in the considered trench
- **nlnum** is the current number of parallel lines of this line element (from the *Basic Data* page).
- **Installation costs:** This defines the total costs to install new cables (excluding the material costs which are defined in the cable type).

## Environment conditions

- *Ambient temperature*: Defines the ambient temperature at the considered location.
- *Soil type*: Selection of all available soil types which correspond to certain thermal resistivities, only editable for laying condition in ground. The values of the thermal resistivities for all soil types in different laying configurations for all standards can be found in the system folder:  
*Database\System\Modules\Cable Sizing\...\Soil Thermal Resistivity (type)*
- *Soil thermal resistivity*: Alternative option (⚙) to the soil type to directly define the soil thermal resistivity, only editable for laying condition in ground.

## Type Parameters

Shows all relevant parameters from the line type for the cable sizing.

## User defined current

This definition can be used as an alternative to the load flow calculation. With this option no load flow calculation is necessary and the current flowing over the cable can be defined via the input parameters:

- *Current*: Amplitude of the current flowing over the cable.
- *Power Factor*: Corresponding power factor of the defined current.

The user-defined current can be used for the sizing of the cable when the corresponding option in the Cable Sizing command is selected. The following constraints can be considered:

- Thermal loading
- Voltage change along cable

---

**Note:** The input parameters for the user-defined current can e.g. be used for scripting purposes to implement additional functionalities so size cables. One possibility could be the definition of the user-defined current in relation to the upstream fuse.

---

## Protection Assessment

A protection assessment can be carried out during the cable sizing of a line, where either existing or potential protection devices are considered. A distinction can be made between fuses, circuit breakers in the domestic sector and circuit breakers in the industrial sector.

- *Ignored*: If this option is selected, protection aspects are not taken into account in the cable sizing.
- *Fuse assumed*: A fuse is considered as protection device.
  - *Fusing current factor k2*: Setting for the k2 factor. This factor is multiplied by the nominal breaking current  $I_n$  of the fuse to calculate the conventional tripping overcurrent  $I_2$ , see Figure 35.1.3. The nominal breaking current  $I_n$  is selected according to the relevant fuse size as described in international guidelines. The relevant fuse size is defined based on the load current  $I_b$ , whereby the next largest fuse size is selected.
- *Circuit-breaker assumed*: A circuit-breaker is considered as protection device.
  - *Usage*: Setting to select the usage as *Domestic* or *Industrial* protection. This affects the choice of typical factors for determining the conventional tripping current (1.45 for domestic and 1.3 for industrial).
  - *Set point*: This parameter allows the discrete nominal breaking current  $I_n$  to be adjusted to any value. The value needs to be between 0.1 and 1.

The additional tabs describe specific input parameters for all International Standards which can be used in the calculation. For more detailed information regarding these parameters the user is advised to consult the source documents.

## 35.1.2 Cable Sizing Calculation

### 35.1.2.1 Basic Options

#### Mode

- If *Verification* is selected, then the command will assess the suitability of the existing cable types against all selected constraints described in Section 35.1.2.2.
- If *Recommendation* is selected, then new cable types or additional parallel lines will be suggested in order to comply with all constraints.
- With the option *Calculate derating factor*, the command can determine the derating factor according to the selected standard. The available standards are:
  - IEC 60364-5-52 (International standard for LV cables up to 1 kV) [18]
  - DIN VDE 0298-4 (German standard for LV cables up to 1 kV) [37]
  - NF C15-100 (French standard for LV cables up to 1 kV) [11]
  - BS 7671 (British standard for LV cables up to 1 kV) [20]
  - DIN VDE 0276-1000 (German standard for cables from 1 kV up to 36 kV) [7]
  - NF C13-200 (French standard for MV cables from 1 kV up to 36 kV) [19]
  - IEC 60502-2 (International standard for cables from 1 kV to 36 kV) [25]

**Note:** Only lines with the appropriate voltage level of the standard can be considered. The definition of the installation condition is done in each line element as described in Section 35.1.1.2.

---

#### Cable type selection

This option is only available when the mode *Recommendation* is selected.

New cable types are recommended for appropriate cables with the following options:

- *According to international standards* is recommending new cable types directly from the selected standard. New cable types with the determined cross-section are created into the selected *Cable type target folder* and assigned to the appropriate cables. The construction of the new cable types can be defined on the page *Type Parameters* of the Cable Sizing command as described in Section 35.1.2.3.

**Note:** Standards tables for cable ampacity, cross-section, derating factors, and impedances are stored in the system library under *Database\System\Modules\Cable Sizing*. Note, that according to these standards, the Max. Operational Temperature as well as the Max. End Temperature of the cable type is kept at the default values of 80 degree Celsius.

---

- *User defined* is recommending new cable types from a selected *Cable type library* which can be defined (▼) as follows:
  - Via *Select...* a folder or project folder from the global or project library can be selected.
  - Via *Select...* multiple cable types (*TypLne*) can be marked from either the global or project library. After pressing OK, a selection (*SetSelect*) with all marked types is created.
  - Via *Select Set...* a pre-defined selection (*SetSelect*) of cable types can be chosen. The selection has to be used for *Cable Sizing* and the type has to be *Cable Sizing - Types*.
  - If a *SetSelect* is already defined, via *Add to Set...* additional cable types (*TypLne*) can be added to the selection.

## Lines

Defines all line elements (*ElmLne*) which are considered in the calculation.

- *All lines* considers all calculation relevant line elements in the project.
- *Selected lines* allows to define a group of line elements that shall be considered.
  - Via *Select...* multiple line elements (*ElmLne*) can be marked directly. After pressing OK, a selection (*SetSelect*) with all marked lines is created.
  - Via *Select...* it is also allowed to define grids (*ElmNet*) and feeders (*ElmFeeder*). All line elements within the selected grouping objects will then be considered in the calculation.
  - Via *Select Set...* a pre-defined selection (*SetSelect*) of line elements, grids and feeders can be chosen. The selection has to be used for *Cable Sizing* and the type has to be *Cable Sizing - Lines*.

## Load Flow type

For the calculation of the currents over the investigated lines and voltages at the terminals different options are available:

- *Standard Load Flow*: Executes a normal Load Flow calculation to determine the power flow in the network.
- *Low Voltage Load Flow*: Executes a Low Voltage Load Flow calculation based on coincidence curves to estimate the maximum currents in the network.
- *User defined current*: The currents over the investigated lines are defined in the elements itself as described in Section [35.1.1.2](#).

## Network Representation

*Balanced, positive sequence* or *Unbalanced* network representation can be selected. The Load Flow command referenced below these radio buttons is automatically adjusted to the appropriate calculation method based on this selection.

## Load Flow, Short-Circuit

These are references (pointers) to the Load Flow and Short-Circuit command used by the calculation. Depending on the selected load flow type either the Load Flow or Low Voltage Load Flow command is selected. The Short-Circuit reference only appears when a user defined cable type selection is used.

### 35.1.2.2 Constraints

#### Thermal Limits

Thermal loading limits will be verified with the selected load flow calculation type.

Optionally select to consider *Thermal constraints* with two available options:

- *Global limit for all lines*: This is the default where all lines are verified against the defined *Maximum thermal loading*.
- *Individual limit per line*: This option considers each component's unique thermal loading limit. Note, the thermal rating is specified in the field *Max. Loading* within the Load Flow tab of each line.

## Voltage Limits

Voltage limits will be verified with the selected load flow calculation type.

- **Voltage change along cables:** If selected, then the *Maximum voltage change* over each individual line is verified against the defined limit.
- **Voltage constraints per terminal:** Optionally select to consider *Voltage constraints per terminal*. There are two options for terminal voltage drop and rise constraints:
  - *Global limit for all terminals* (absolute value in p.u.): If selected, a *Lower Voltage Limit* as well as an *Upper Voltage Limit* must be entered in the corresponding fields.
  - *Individual limit per terminal*: Note, the voltage limits are specified in the Load Flow tab of each terminal.
- **Voltage constraints along feeder:** For balanced calculations, optionally select to consider *Voltage constraints along feeder*. The *Maximum voltage drop* as well as the *Maximum voltage rise* are calculated as the percentage voltage difference between the source terminal of the feeder and the final terminal of the feeder. There are two options for feeder voltage drop and rise constraints:
  - *Global limit for all feeders*. If this option is selected, then the *Maximum voltage drop* and the *Maximum voltage rise* must be entered into the corresponding fields.
  - *Individual limit per feeder*. Note, the maximum voltage drop and rise are specified in the Load Flow tab of each feeder.

---

**Note:** Depending on the system topology, on the loads and on the length of the feeder, it might not be possible to avoid voltage drop violations of some terminals or feeders. This can be mitigated by the installation of a capacitor/s during a post-processing optimisation. See Section [42.10: Optimal Capacitor Placement](#).

---

- **Terminals to be checked:** Two options are available to define which terminals in the network are considered and verified against the defined voltage limits:
  - *Terminals of the selected lines* only considers the terminals of the investigated lines.
  - *All terminals* considers each terminal in the network where the defined *Sensitivity threshold* criteria is met. This ensures that only terminals in the investigated network area are verified and violations can possibly also be solved.

## Short-Circuit Limits

Short-circuit calculation will be executed with the linked *Short-Circuit Calculation* command. Only 3-Phase faults are considered to size cables.

Enabling the option *Short-circuit constraints* will consider the short-circuit limits in the cable sizing.

If the *Verification* mode or the *Recommendation* mode with *User defined* cable type selection is selected (on page *Basic Options*):

- **Maximum Loading:** A percentage of the rated short-circuit current from the cable type data can be defined. Note, that the short-circuit duration has a large impact on the thermal loading and is specified in the *Short-Circuit Calculation* command with the parameter *Fault Clearing Time (Ith)*.
- **Consider protection device rating:** If enabled, the report displays the rating of the protection device installed in one of the connected line cubicles, if present. This selection does not affect the cable recommendation.

If the *Verification* mode (on page *Basic Options*) is selected, the option *Sweep fault clearing time (Ith)* can be enabled with the following settings:

- *Start time*: Start time of the sweep.
- *End time*: End time of the sweep.

- *Time step:* Time step of the sweep.

If the *Recommendation* mode with cable type selection *According to international standards* is selected (on page *Basic Options*):

- *Short-circuit limits calculation:* If enabled, protection relevant short-circuit limit values are calculated and shown in the report. The values are: *Max. short-circuit current*, *Min. short-circuit current*, *Max. thermal stress energy* and *Actual thermal stress time*. It does not affect the cable recommendation.
- *Protection device assessment:* If enabled, the protection assessment is carried out for the cable sizing of a line, leading to additional reported protection design relevant parameters. The assessment depends on the protection assumptions in the line settings (see Section 35.1.1.2).

Depending on the selection, the cable sizing reports are expanded and show results for short-circuit calculation and protection-relevant data. All data from the applied industry guidelines for the cable sizing and protection scheme in Figure 35.1.3 are part of the report. Cable-related data is displayed above and the protection-related data below the line.

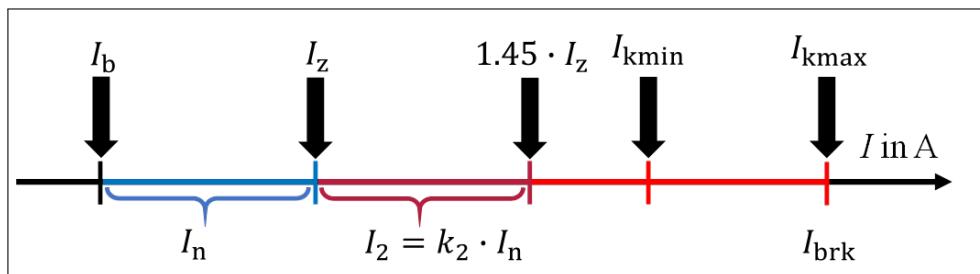


Figure 35.1.3: Cable sizing and protection scheme

The nominal breaker current  $I_n$  needs to be between the load current  $I_b$  and the current-carrying capacity  $I_z$  of the sized line. In *PowerFactory*, the current-carrying capacity is the rated current of the line. The conventional tripping overcurrent  $I_2$  is between  $I_z$  and  $1.45 \cdot I_z$  (or  $1.3 \cdot I_z$  for industrial circuit breakers). The minimal breaking capacity current  $I_{\text{brk}}$  is set to the value of the maximum short circuit current  $I_{\text{kmax}}$ .

In addition, the report shows results for the maximum thermal stress energy, the actual thermal stress time and the maximum protected length.

### Network Consistency

This option, if enabled, forces the optimisation routine to complete a final “consistency” check of the Line Type rated current based upon one of two criteria:

1. *Sum of feeding cables*  $\geq$  *Sum of leaving cables*; or
2. *Smallest feeding cable*  $\geq$  *Biggest leaving cable*.

To explain what is meant by “feeding cable” and “leaving cable” consider the example feeder shown in Figure 35.1.4. This network is defined as a single “feeder” that begins at the “Source” terminal. Consider now “Terminal A”. This terminal is supplied by “Line A” and is also connected to two other cables, “Line B” and “Line C”. In this case, for “Terminal A”, “Line A” is considered as a “feeding cable” and Lines B and C as “leaving cables”.

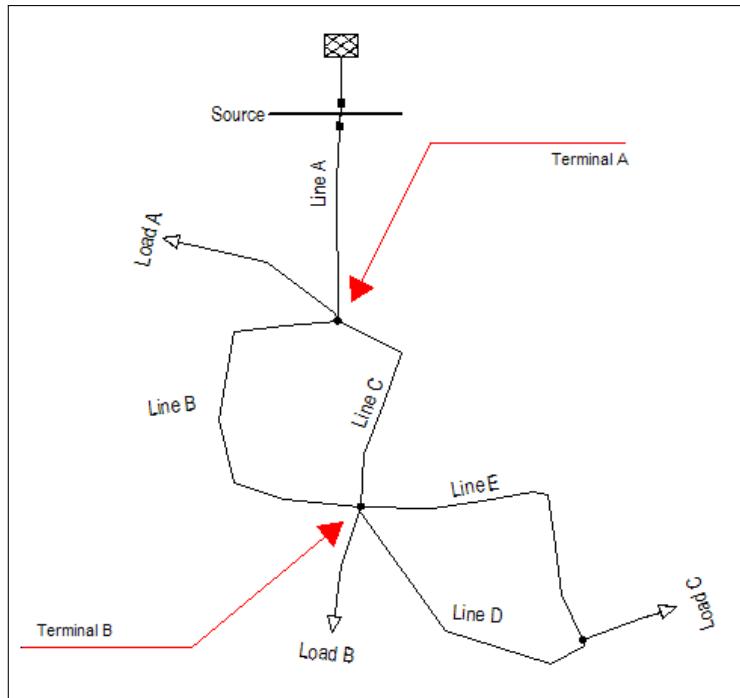


Figure 35.1.4: Example feeder network

Considering now “Terminal B”, Lines B and C are feeding cables whereas Lines D and E are “leaving cables”. “Feeding cables” are defined as those cables with a power flow direction that is into the connecting node. For a radial feeder with no embedded generation, this is generally the cables closest to the beginning of the feeder. All other cables are defined as “leaving cables”.

In consistency check option 1, the cross sectional area (or rated current) of the feeding cables are summated and compared with the sum of the cross sectional area (or rated current) of the leaving cables for each terminal. If the sum of the leaving cables is greater at any terminal then the network is considered non-consistent.

For consistency check option 2, the smallest feeding cable is compared with the largest leaving cable for each terminal. If the largest leaving cable is bigger than the smallest feeding cable, then the network is considered non-consistent.

### 35.1.2.3 Type Parameters

Type parameter options are only applicable if the mode is set to *Recommendation* on the Basic Options page and the cable type selection is done *according to international standards*.

The parameters are used to create a suitable cable type according to the selected standard.

- The following parameters can be defined for the cable types according to the International Standards:
  - Conductor Material
  - Insulation Material
  - Cable Cores: *Single-Core* or *Multi-Core*
  - If the cable shall be modelled *With sheath*, additional options are available:
    - \* Sheath Type
    - \* Sheath Insulation
    - \* Optionally select if the cable is an *Armoured Cable*
    - \* Optionally select if the cable has a *Radial Cable Screen*

- When the option *Default cable impedance value* is selected, the user can define impedance values for the positive-, negative- and zero-sequences that are taken as default values for new cable types. When this option is not selected, internal default values from the system folder (*Database\System\Modules\Cable Sizing\... \R – X Default Values*) are used for the calculation. This only applies when new cable types are created and the *Cable electrical parameters (20 °C)* are not changed.
- For the definition of the *Cable electrical parameters (20 °C)* there are different options available:
  - When the option *Not changed* is selected, existing cable types will not be changed and newly created cable types will be defined with default impedance values.
  - The option *Calculate according to IEC 60909* determines impedance values for the cable depending on the cross-section and type parameters according to the IEC 60909 standard.
  - The option *Calculate according to UTE C 15-500* determines impedance values for the cable depending on the cross-section and type parameters according to the UTE C 15-500 standard.

#### 35.1.2.4 Output

Various output options for the results are possible.

- *Output all derating factors*: If this option is selected, then there will be a detailed output of all derating factors in the output window. This option is only available when derating factors are calculated *according to international standards*.
- *Modify cables*: If this option is selected, cables in the network can be modified according to the calculation results, meaning that cable types and derating factors are directly assigned. Two options are available to modify the existing cables in the network:
  - *Change existing network*: If this option is selected, the cables in the network will be modified according to the calculation results. Changes will be written directly into the grid or recorded by the active expansion stage (if exists).
  - *Create a new Variation with recommended Cables*: If this option is selected, then a new variation will be created containing the proposed modifications of the cables according to the calculation results.

#### Results

This is a reference (pointer) to the results output. It is possible to select an alternative results file. The structure of the *Cable Sizing* result file is explained in more detail in Section [35.1.3.3](#).

#### 35.1.2.5 Recommendation Options

All options for the recommendation of new cable types or additional parallel lines can be defined on this page.

##### General

- ***Downsizing***: If selected, cables that fulfil all constraints are allowed to be downsized meaning that new types with lower cross sections are investigated which still fulfil all constraints. The resulting cable type will just about fit all requirements.
- ***Maximum voltage deviation***: This option is only used for user-defined cable selections. A deviation of the cable type rated voltage in comparison to the nominal voltage in the network can be defined. If “0%” is entered, the rated voltage on the cable type should match the nominal voltage of the terminal to which it connects. If a non-zero value is entered, the rated voltage of the cable type can differ by the defined percentage. If the deviation is larger than the given threshold,

the cable type cannot be used for a dedicated line. Important to note is that cables with a larger rated voltage than the nominal voltage of the terminal can always be installed. So the threshold only concerns cable types with lower rated voltages compared to the terminal's nominal voltage.

- **Safety margin for the cable current capacity:** This option is only used for cable selections according to international standards. A *safety margin* can be defined in percent. If a non-zero safety margin is entered, a cable with higher capacity is selected according to the selected standard which at least fulfils the safety margin requirement.
- **Existing cable type replacement:** This option is only used for cable selections according to international standards. Two options are available when new cable types are created according to the selected standard. Select whether to *Use design parameters of the "Type Parameter" page*, in which case a new type will be created according to the type design parameters from the command. Or, select to *Use the existing design parameters of the cable type*, in which case a new type will be created according to the existing cable type from its rated values (only current and cross-section values could be different). This is only applicable if the analysed cable has a type assigned. Otherwise, a new type will be created according to the command parameters.
- **Assign missing line types:** Optionally select to *Assign Missing Line Types* when there are cables in the network without types.

## Parallel Lines

All options for the creation of parallel lines can be defined on this page. When selecting the option *Creation of parallel lines* it is possible (in addition to higher cross sections) to install new parallel lines in the network. This is only applies for the user-defined cable selection.

- **Maximum number of parallel lines:** The allowed number of parallel lines can be limited for each line element (*ElmLne*) with this input parameter. There are two options available:
  - *Global limit for all lines*: If this option is selected, then the defined *Limit* is globally applied to all investigated lines.
  - *Individual limit per line*: With this option each investigated line has its own individual limits depending on the installation conditions. The settings and the calculation of the maximum number of a line are described in Section [35.1.1.2](#).
- **Priority for the optimisation process:** When the creation of parallel lines is allowed, then the calculation needs an objective function in order to decide to install cable types with a higher cross section or additional parallel lines.
  - *Prioritise cross section against parallel lines*: Cable types with higher cross section are prioritised. Only if the cable type with the highest cross section can't solve the constraints additional parallel lines are installed.
  - *Prioritise parallel lines against bigger cross section*: Additional parallel lines with the same type as the existing one are prioritised. Only if the maximum number of parallel lines can't solve all constraints other types with higher cross sections are used.
  - *Minimise costs*: The decision whether to install additional parallel lines or to use types with a higher cross sections depends on the resulting cost. The solution that causes minimal costs is determined. The costs in the line elements (*ElmLne*) as well as in the line type (*TypLne*) are considered and need to be defined.
- **Parallel lines between two busbars only:** This option limits the installation of parallel lines for a specific line element. If selected, only lines where both connecting terminals have the usage busbar are considered for the creation of additional parallel lines.

### 35.1.2.6 Advanced Options

The following options are only applicable if the mode on the *Basic Options* page is set to *Recommendation* and the cable type selection is done *According to international standards*.

- **Consider DC cables:** If selected, DC cables are also considered in the calculation. Note, that the DC cable sizing is not within the scope of the used *international standards*. But DC cables can be sized with the same rules that apply for AC cables.
- **Always include neutral conductor:** If this option is enabled, the recommended line will have a neutral conductor. Applies only for standard *NF C 15-100*, *NF C 13-200* and *BS 7671*.
- **Tolerance for the searched tabulated current capacity value:** Setting for the tolerance of the searched tabulated current capacity value. The value should be between 0 and 10. Applies only for standard *NF C 15-100*.

The following option is only applicable if the mode is set to *Verification* or if the *Recommendation* is based on the user-defined cable selection.

- **Ignore all lines for...:** If enabled, a voltage limit can be considered for the investigated lines. In this case, a *Nominal voltage above* which lines are ignored in the calculation must be entered.

### 35.1.3 Cable Sizing Results

#### 35.1.3.1 Cable Sizing Report

The *Cable Sizing Report* () can be accessed via the corresponding icon. The *Report Generation* command (see Section 19.5.1) can create all available reports:

- *Cable Verification:* Verification report with all investigated constraints
- Reports for user-defined cable selection
  - *Initial Parameters for Recommendation:* Includes all line parameters before the recommendation process.
  - *Cable Verification for Recommendation:* Includes the results of all lines before the recommendation process.
  - *Cable Recommendation:* Lists the recommended changes that were determined by the cable sizing command.
- Reports for cable selection according to international standards
  - *Cables according to Int. Standards:* Recommendation report with newly created types according to the selected international standard

#### 35.1.3.2 Cable Sizing Tabular Report

The *Cable Sizing Tabular Reports* () can also be created for the cable selection *Recommendation According to international standards*.

#### 35.1.3.3 Cable Sizing Result File

Every result file (*ElmRes*) have an own structure. Input can be retrieved via line number or via an index. The cable sizing result file has the following order:

- Initial value - Initial calculation of all parameters of feeder, *ComCabsize*, cables and terminals. (index 0)
- Derating factors - Results of calculated derating factors for each cable. (index 20)
- Thermal cable verification - Only those cables' variables are written which violate the thermal constraint. (index 1)

- Thermal cable recommendation - Only those cables' variables are written, for which a new cable type is recommended during thermal recommendation process. The cost of improvement is also written. (index 2)
- Thermal cables cannot be solved - Only those cables' variables are written, that are unsolvable and still violate thermal constraints after thermal recommendation process. (index 3)
- Voltage verification results regarding voltage change along all considered cables. (index 15)
- Voltage recommendation regarding voltage change - Only those cables' variables are written, for which a new cable type is recommended during the voltage recommendation process. The cost of improvement is also written. (index 16)
- Voltage recommendation regarding voltage change along cables cannot be solved - Only those cables' variables are written, that are unsolvable and still violate voltage constraints after voltage recommendation process. (index 17)
- Voltage verification - Only those terminals' variables are written which violate voltage constraints. (index 4)
- Voltage recommendation - Only those cables' variables are written, for which a new cable type is recommended during voltage recommendation process. The cost of improvement is also written. (index 5)
- Terminals cannot be solved - Only those terminals' variables are written which are unsolvable and still violate voltage constraints after the voltage recommendation process. (index 6)
- Consistency verification - Only those terminal's variables are written which violate network consistency. (index 7)
- Consistency recommendation - Only those cables' variables are written, for which a new cable type is recommended during the consistency improvement process. The cost of improvement are also written. (index 8)
- Consistency violation - Only those terminals' variables are written that are unsolvable and still violate network consistency after the recommendation process. (index 9)
- Changed cables load flow - Only those cables' variables are written, for which a new cable type is recommended after the complete load flow optimisation process. (index 10)
- Short-circuit verification - Only those cables' variables are written which violate short-circuit constraints. (index 11)
- Short-circuit cable recommendation - Only those cables' variables are written, for which a new cable type is recommended during short-circuit recommendation process. The cost of improvement is also written. (index 12)
- Short-circuit cables cannot be solved - Only those cables' variables are written, that are unsolvable and still violate short-circuit constraints after the recommendation process. (index 13)
- Changed cables - Only those cables' variables are written, for which a new cable type is recommended after the complete optimisation process. (index 14)

Results are indexed as follows when cables are selected according to international standards:

- Pre-recommendation results. (index 102)
- Post-recommendation results. (index 103)
- Post-recommendation results for current carrying capacity calculation (for standard DIN VDE 0276-1000). (index 104)

## 35.2 Cable Ampacity

*PowerFactory*'s Cable Ampacity calculation supports two methods described in literature to determine the current rating of cables:

- *IEC 60287* [24] [26] [13]
- *Neher-McGrath* [52]

The IEC 60287 standard series is an important international standard for the calculation of the electric cable current rating.

The Neher-McGrath method is derived from the corresponding paper [52]. It does not have the status of a standard, but some effects are taken into account that are not considered in the international standard. In addition, the IEC 60287 standard is highly influenced by the work of J. H. Neher and M. H. McGrath. For that, Neher-McGrath method is still a widely accepted method.

---

**Note:** Even if the two methods *IEC 60287* and *Neher-McGrath* are related to each other, there can be significant differences in the results due to the different parametrisations.

---

The Cable Ampacity calculation takes into account the physical model with the laying conditions and calculates the heat flow and the temperatures via a thermal equivalent circuit, see Figure 35.2.1. That is the basis for the cable ampacity calculation which can only be performed for detailed cable models. In *PowerFactory* this is done with a *Cable Definition (TypCabsys)*.

The Cable Ampacity calculation can be done for cables (*ElmLne*) or cable layouts (*ElmCablay*). In a cable layout (*ElmCablay*) there can be a group of cables (*ElmLne*) or so called cable systems (*ElmCabsys*). Cable systems (*ElmCabsys*) can also contain a group of cables (*ElmLne*).

It is necessary for the Cable Ampacity calculation, that the cables (*ElmLne*) have an assigned *Cable Definition (TypCabsys)*. Cable systems (*ElmCabsys*) can only be defined with an assigned *Cable Definition (TypCabsys)*. And every *Cable Definition (TypCabsys)* needs one or more *Single Core Cable Types (TypCab)* or one *Multicore/Pipe Cable Type (TypCabmult)*. Follow Section 12.3.5 for more information about detailed cable modelling.

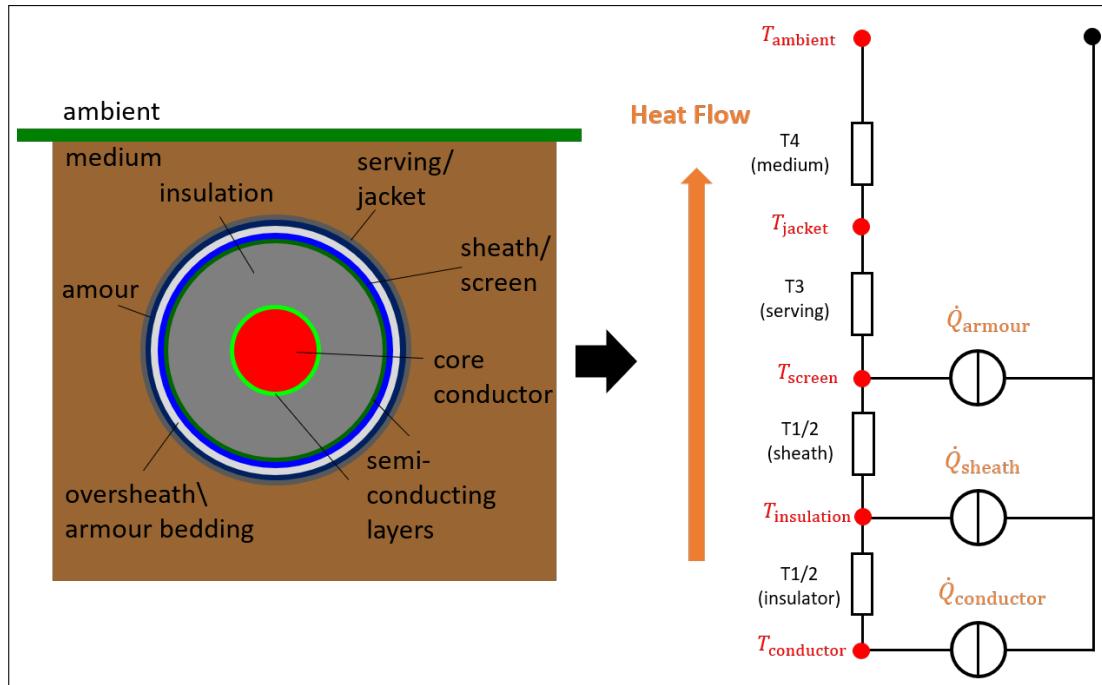


Figure 35.2.1: Cable Ampacity calculation - detailed cable model and thermal equivalent circuit

Within the *Cable Definition (TypCabsys)* there is a specification of a rudimentary laying condition (laying in row or in trefoil, laying depth). In case of several circuits, the positions between the circuits can also be entered there. This can be specified with the cable layout (*ElmCablay*) too. The difference is that circuits inside a *Cable Definition (TypCabsys)* are electrically coupled, cables (*ElmLne*) inside a cable layout (*ElmCablay*) are not electrically coupled. But with cable layouts, advanced laying conditions can be defined, which is important for the thermal coupling and thus for the ampacity calculation.

More details about the cable layout are described in Section 35.2.1.4.

Before carrying out the Cable Ampacity calculation it is necessary to provide a certain amount of input data for the involved elements and types, see Section 35.2.1.

## 35.2.1 Cable Models for Cable Ampacity

### 35.2.1.1 Cable System Type Parameters

Relevant settings in the *Cable Definition (TypCabsys)* ( ⓘ) dialog can be found on the page *Cable Ampacity*.

- Regularly transposed: This option influence the calculation of the loss factor.
- Unequally loaded cables: If this option is unmarked (default), an equally loading of the cables is assumed.

### 35.2.1.2 Core Cable Type Parameters

Data may be entered on the page *Cable Ampacity* in the dialog of the *Single Core Cable Type (TypCab)* ( ⓘ) or *Multicore/Pipe Cable Type (TypCabmult)* ( ⓘ). Both types have the following shared parameters:

- *Max. Operating Temperature*: Is the maximum operating temperature of the conductor for the load flow calculation.

- *Max. End Temperature*: Is the maximum temperature of conductor for the short-circuit calculation.
- *Max. operating Temperature (screen)*: Is the maximum operating temperature of the screen (only relevant for Neher-McGrath and if a screen exists).
- *Surface emission coefficient*: Absorption coefficient of solar radiation for cable surfaces.
- *Dried and impregnated*: This options influence the choice of the coefficients for the skin and proximity effect.
- *Fault duration*: For the adiabatic short-circuit calculation. A value of 1 s (default) is recommended.

In case of a *Multicore/Pipe Cable Type* (*TypCabmult*), additional parameters can be set. The additional parameters depend on the selected option on the *Basic Data* tab, if its a *Pipe-type* or a *Multicore cable*.

For pipe cable types:

- *Pipe type*: Defines the pipe material.
- *Pipe thermal resistivity*: Defines the pipe thermal resistivity.

For multicore cable types:

- *Filler material*: Defines the filler material between the conductors.

Additional parameters can be set for IEC 60287:

- *Diameter reduction factor (common sheath)*: Defines the diameter reduction factor of the common sheath.
- *Armour type*: Armour as *Metal tape* or *Metal wire* can be selected.
- *Belted cable (paper insulated)*: Defines, if the cable should be treat as belted cable.
- *Skin effect factor ks*: Specified coefficient for the skin effect (also available for *Single Core Cable Type* in case of unknown conductor material).
- *Proximity effect factor kp*: Specified coefficient for the proximity effect (also available for *Single Core Cable Type* in case of unknown conductor material).

---

**Note:** For a *Multicore/Pipe Cable Type* the factors ks and kp need to be user defined depending on the conductor type and its insulation system. The table in the standard can help in the decision.

---

### 35.2.1.3 Line Element Parameters

Data may be entered on the page *Cable Ampacity* in the dialog of the line element (*ElmLne*) (  ).

The IEC 60287 tab and Neher-McGrath tab have the following shared parameters:

- *Ambient temperature*: Ambient temperature.
- *Soil thermal resistivity*: Thermal resistivity of the soil.
- *Medium mean temperature*: The initial value for the mean temperature of the medium within the duct in case of ducted cable.
- *Exposed to direct sunlight*: Defines, if the cable is directly exposed to solar radiation.

The IEC 60287 tab has in addition:

- *Dry soil resistivity*: Thermal resistivity of dry soil.

- *Solar radiation intensity*: Is the intensity of solar radiation which should be taken as  $1000 \text{ W/m}^2$  for most latitudes in case of cables exposed to solar radiation.
- *Critical temperature of soil*: This is the temperature of the boundary between dry and moist zones.
- *Soil drying-out avoided*: Selection leads to another calculation according to the standard.

---

**Note:** The *Cable Ampacity* calculation according to the *IEC 60287* follows an approach of dry and moist zones. Around the cable a dry zone is defined. Depending on the current and the generated heat, the radius of the dry zone increases. To limit the extension of the dry zone a *Critical temperature of soil* can be defined. If the option *Soil drying-out avoided* is selected, then the dry zone will be reduced to a minimum, meaning that the soil around the cable will be exposed to minimal thermal stress. This can often lead to very small maximum currents which are allowed to flow through the conductor.

---

The Neher-McGrath tab has in addition:

- *Wind velocity*: Wind velocity, relevant for cables in free air.
- *Medium diffusivity*: Is a transport coefficient for the ambient medium.
- *Load factor (lf)*: Factor for the utilisation of the cable.

---

**Note:** If the line (*ElmLne*) is used in a *Cable Layout* (*ElmCablay*) and the *Cable Ampacity* calculation is performed for that *Cable Layout* (*ElmCablay*), the *Cable Ampacity* line element (*ElmLne*) parameters will be overwritten by the cable layout parameter.

---

#### 35.2.1.4 Cable Layout Parameters

The *Cable Layout* object (*ElmCablay*) () is an object specifically intended for use in the *Cable Ampacity* calculation and can be used to define all factors relevant to the derating of *groups* of cables. It conveniently brings the relevant parameters together in one object. With this object deratings can be calculated for groups of cables which are similarly installed but differently constructed, for example, the deratings applicable when a multicore cable is installed in a bank of ducts along with a 3 phase circuit consisting of single core cables. The positions of the individual cables comprising each cable system are geometrically defined. This data is then used to determine the influence of the physical proximity of the cables to one another on their respective ampacities.

---

**Note:** Parameter for the installation data, laying geometry and ambient data of a *Cable Layout* (*ElmCablay*) are always prioritised over parameter in the line element (*ElmLne*) and the *Cable Definition* (*TypCabsys*).

---

**Note:** The distinction between a *Cable System* object (*ElmCabsys*) and a *Cable Layout* object *ElmCablay* may not be immediately obvious. Both elements are associated with the grouping of cable circuits. However, it should be appreciated that the *Cable System* Element (*ElmCabsys*) is associated with the calculation of cable impedances, whilst the *Cable Layout Object* (*ElmCablay*) is associated with the calculation of the derated maximum current rating of the grouped cables.

---

Electromagnetically coupled cable systems (*ElmCabsys*) as well as uncoupled cable system objects (*TypCabsys*) can be handled by the object. Multi-core cable system models are also supported by

the calculation. Aerial, ground, duct and trench with multiple backfill layer installations as well as ambient conditions can all be considered, with deratings being calculated in accordance with either the IEC 60287 standard calculation method or the Neher-McGrath method.

For underground installations it is also possible to consider additional derating caused by the presence of external heat sources such as nearby steam pipes for example. For an IEC 60287 calculation additional consideration can be given to whether an aerially installed cable is installed with brackets, ladder or cleats or alternatively clipped to a wall. For duct installations it is possible to define the dimensions, spacing and material of the ducting.

The object can be defined by multi-selecting the relevant cable objects (lines with a *TypCabsys* as type) in the Single Line Diagram or Data Manager, clicking the right hand mouse button on one of the selected lines and then choosing the options *Network Models* → *Cable Layout* → *New...* from the context menu. The new object is stored in the *Cable Layouts* folder stored in the *Network Data* folder accessible via the *Data Manager*.

All available parameters of the *Cable Layout* are described in the following sections:

## Basic Data

- *Location*: Specify whether the cable grouping under consideration is buried in the ground or mounted in the air. It is not possible to consider the derating effects of cables mounted in the air on cables installed under ground or vice-versa. If a selected cable grouping contains a mixture of circuits, some of which are installed underground and some above ground, only the ones with a location corresponding to the *Location* setting will be considered by the calculation.
- *Number of circuits*: Specifies the number of rows in the *Circuits* table located under the setting. Using this table, additional circuits can be added to the grouping.
- *Laying*: Specifies how the cable is laid in the chosen *Location*. The options available vary depending on whether the *Location* is specified as *Air* or *Buried*. For cables installed above ground the *Laying* parameter can be specified as either *in free air* or *in duct banks*. For buried cables the options are *directly in ground*, *in duct banks* or *in trough/trench*. If cables are installed in duct banks additional setting options are displayed for the definition of the duct bank. If the option *in trough/trench* is selected an additional setting option is presented allowing the user to specify whether the trough is *backfilled* or *unfilled*. If *unfilled*, then the user is also able to specify if the cable is exposed to free air or not using a checkbox.

## Installation Data

The options at the *Installation Data* page highly depends on the selection of the *Laying* at the *Basic Data* page and if the location is in air or buried. Two tabs are available. One associated with the IEC 60287 method and one associated with the Neher-McGrath method.

- In case of a laying ***in free air***, the following option can be set:
  - *Exposed to direct sunlight*: This parameter have an influence on the calculation of the heat dissipation.

**Note:** In case of the location on air, no additional data can be added for the Neher-McGrath method.

- 
- In case of a cable laying ***directly in ground***, the option *Ducted cables* can be enabled. This allows the user to set up a table with the selection of single cable which should be installed in a duct. In case of a cable with an single core cable type (*TypCab*) the *Ducting* can be defined for *all phases* or *per phase*. In case of a cable system (*ElmCabsys*), only one kind of ducting can be selected for all circuits within the cable system. For Neher-McGrath, no additional data are necessary since this configuration is not defined in the method.
  - In case of a laying ***in duct bank***, the following option can be set (in case for buried cables also for Neher-McGrath):
    - *Exposure to direct sunlight*: This parameter have an influence on the calculation of the heat dissipation.

- *No. of ducts (vertical)*: Number of ducts one above the other.
  - *No. of ducts (horizontal)*: Number of ducts next to each other.
  - *Vertical spacing between ducts*: Vertical spacing between two ducts.
  - *Horizontal spacing between ducts*: Vertical spacing between two ducts.
  - *Distance from surface*: Distance from the surface to the middle of the duct bank system.
  - *Width*: Width of the concrete block in which the ducts are embedded.
  - *Duct outer radius*: Outer radius of each duct.
  - *Duct thickness*: Thickness of each duct.
- 

**Note:** Added installation data should not lead to conflicts with other data. For example, the outer radius of the duct cannot be smaller than the cable itself.

---

**Note:** The position of each circuit in the ducts can be defined on the page *Laying Geometry*.

---

- In case of a laying **in trough/trench**, the following option can be set:
    - *Trough/trench filling*:
      - \* *Backfilled*: The trough/trench is filled with some material.
      - \* *Unfilled*: The trough/trench is unfilled.
    - *Trough/trench*:
      - \* *Width*: Width of the trough/trench.
      - \* *Depth*: Depth of the trough/trench.
    - In case of *Unfilled*, an option *Exposed to free air* can be enabled. This parameter have an influence on the calculation of the heat dissipation.
    - In case of *Backfilled*, there will be the option to increase the *Number of backfill layers* up to 3 as well as the setting for the *Top layer thickness* and *Middle layer thickness*.
- 

**Note:** In case of the *Neher-McGrath* method, trough/trench thickness can be set for one layer.

---

For further information regarding the usage of the setting parameters, please refer to the source documentation for the relevant method.

## Ambient Data

On this page of the object dialog information about the ambient conditions of the cable group is entered. Two tabs are available. One associated with the IEC 60287 method and one associated with the Neher-McGrath method. The parameters which are displayed depend on the combination of settings specified on the *Basic Data* tab of the object.

- In case of a laying **in free air** the *Air temperature* can be set. At the IEC 60287 tab, a *Solar radiation intensity* can be set too. In addition, a *Wind velocity* and a *Solar radiation impact* can be set on the Neher-McGrath tab.
- In case of a **buried cable**, the *Ground temperature* and (in case of IEC 60287) the *Critical temperature of the soil* can be adjusted.
- In case of **ducted cables** or **duct banks**, the initial value for the *Medium mean temperature* in the duct can be set.
- In case of a laying **in trough/trench**, an additional *Surface conductance* coefficient can be set on the IEC 60287 tab.
- On the Neher-McGrath tab, a *Medium diffusivity* can be set, which is a transport coefficient for the ambient medium.
- *Thermal resistivity*: In that category, depending on the input on the *Basic Data* tab, several parameters can be set related to a part of the thermal resistivity of the cable layout, e.g. the thermal resistivity of the *Moist soil* or of the *Backfill* material as well as the *Duct material* type in case of ducts.

For further information regarding the usage of the setting parameters, please refer to the source documentation for the relevant method.

## External Heat Source

External heat sources are only supported for buried systems which are laying directly in ground or in trough/trench. On this page of the object dialog information about any external heat sources acting on the cable group can be entered. External heat sources might include for example an adjacent steam pipe radiating heat. Two tabs are available. One associated with the IEC 60287 method and one associated with the Neher-McGrath method. The *Number of heat sources* setting parameter determines the number of rows in the associated table.

For further information regarding the usage of the setting parameters, please refer to the source documentation for the relevant method.

## Advanced Data

This page is used to provide additional data which can influence the derating of the cable group. Two tabs are available. One associated with the IEC 60287 method and one associated with the Neher-McGrath method.

In case of the IEC 60287 method, the option *Soil drying-out avoided* can be selected which leads to a more conservative calculation according to the standard to minimise the heat dissipation if the cables.

In case of the Neher-McGrath method, a global load factor can be selected which describes the general utilisation of the cable.

---

**Note:** The IEC 60287 only considers a load factor of 100 %.

---

For further information regarding the usage of the setting parameters, please refer to the source documentation for the relevant method.

## Laying Geometry

This page graphically illustrates the layout of the cable group. If a duct bank or a trench is used this is also illustrated. The dimensions and geometry of the duct bank as defined on the *Basic data* page directly influence the illustration of the duct bank. Note that the geometry of an individual cable system is defined within the cable system type *TypCabsys*. When this cable system is then included as part of a cable grouping in a *Cable Layout object*, the already defined geometrical relationships must continue to be respected. A new Cartesian coordinate system for the cable group is defined with a specific origin. The origin can be displayed by choosing the *Display system reference point* check-box. For all laying methods except *in duct banks*, the origin from the cable system type is by default set equal to the origin of the cable group. However, in many cases this can result in cables occupying the same space, so by selecting the *User-defined offset* checkbox each line can be given a unique position relative to one another. For cables installed in air, the positive direction of the Y-offset is upwards into the air. For buried cables the positive direction of the Y-offset is into the ground.

For cables installed in duct banks, each duct has its own origin located at its centre. For each cable the corresponding duct can be specified as *Duct position* within the *Circuit in duct* table. The duct position numbering increases sequentially from left to right and from bottom to top.

Figure 35.2.2 show some example laying geometries for different cable layout settings.

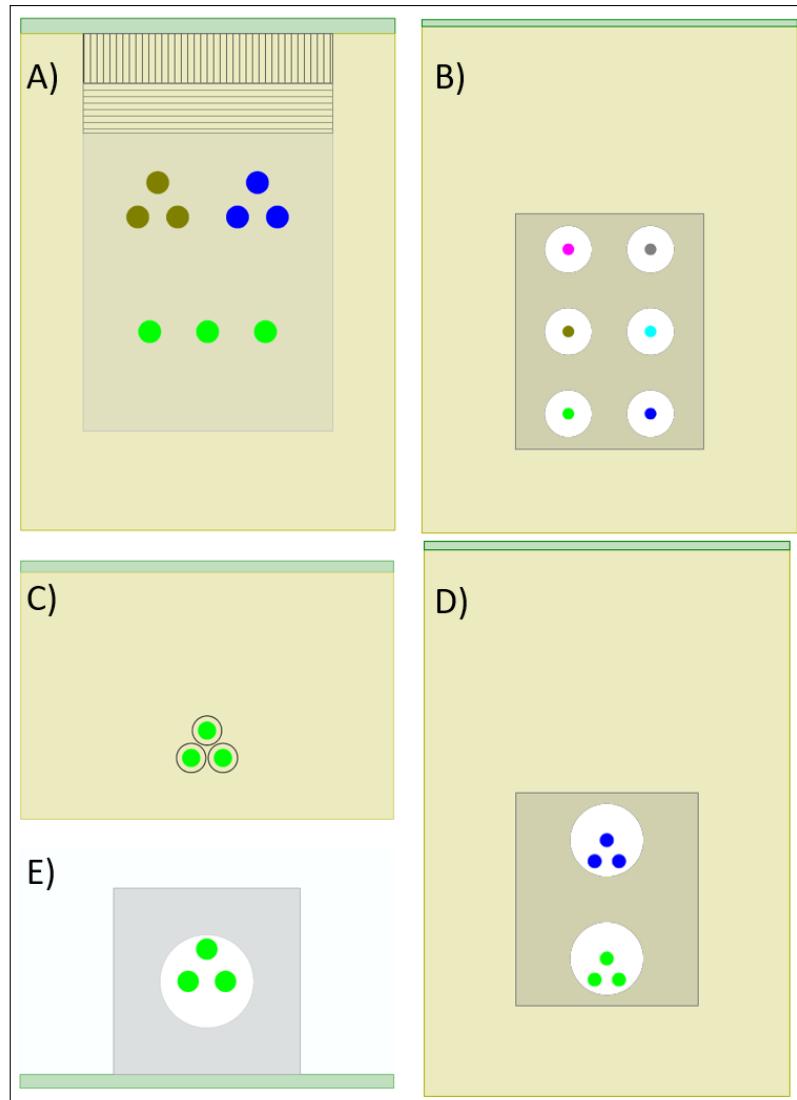


Figure 35.2.2: Cable Layout - Laying Geometry examples

- A) Three single core cables buried in tough/trench with 3 backfill layers.
- B) Six multicore cables buried with a duct bank.
- C) One single core cable buried directly in ground with ducting per phase.
- D) Two single core cables buried within a duct bank.
- E) One single core cable in air within a duct bank.

### 35.2.2 Cable Ampacity Calculation Command

The objective Cable Ampacity calculation is to calculate the current in a way, to hold the temperature in the cable under the max. allowable operating temperature. The temperature in the cable depends on the heat sources in the cable or around the cable as well as on the heat transfer (thermal resistivity) to the environment. The system can be described with an thermal equivalent circuit like in Figure 35.2.1 including heat sources and thermal resistances. The heat transfer calculation is based on an analytical algorithm. The aim is to determination the inner conductor temperature.

To access the Cable Ampacity command (*ComAmpacity*) select the *Cable Ampacity* (🕒) from the Cable Analysis toolbar.

### 35.2.2.1 Basic Data

The *Basic Data* page of the Cable Ampacity command contains following input settings:

#### Method

- IEC 60287 (selected by default)
- Neher-McGrath

#### Selection

- *Cables*: Selection of either ungrouped or grouped cables. If the option *Cables* is selected, cable ampacities will be calculated considering each selected cable independently. When selecting the lines to be analysed, line objects will only be presented by the filter if a *Cable Definition* (*TypCabsys*) is defined for the line.
- *Cable layout*: If the option *Cable Layout* is selected, *Cable Layout* objects should be chosen and cable ampacities will be calculated taking account of the various special derating effects facilitated by this object see Section 35.2.1.4 for more information.

### 35.2.2.2 Output

- *Results*: Pointer to the *Result File* (*ElmRes*).
- *Modify derating factor of lines*: If this option is selected, there will be a modification in the derating factor in the model to match the calculated maximum allowed current. The derating factor is considered to be the ratio between the max. ampacity and the rated cable current. Could be done in two different ways.
  - Modifies derating factor *in the existing network*.
  - Modifies derating factor *in the new variation*.

---

**Note:** The derating factor is an attribute within the line element. It is by default 1. The modification is done based on the calculated value and the insert rated current in the *Cable Definition* (by default 1 kA). To get a derating factor in a common scale, the rated current of the *Cable Definition* can be set to a reasonable value.

---

### 35.2.2.3 Advanced Data

- *Calculate adiabatic short-circuit rating*: If this option is checked *PowerFactory* will calculate a short circuit rating for the cable systems based on the adiabatic equation. The rating can be calculated for each cable system for a time defined in the cable ampacity command itself, or alternatively, times can be defined within individual cable systems (*TypCab* or *TypCabmult*) and with each time considered independently for each cable. Once calculated, this rating along with the calculated ampacity can then be used in further analysis, for example in the creation of cable damage curves in time overcurrent plots for protection coordination or for comparison with the thermal equivalent short circuit current calculated via short circuit analysis.
- *Initial step*: This value can be set to adjust the performance. The choice will not effect the final result. A smaller value may lead to longer iteration process.
- *CIGRE TB 880*: The technical brochure CIGRE TB 880 [39] contains power cable rating examples on the basis of the standard IEC 60287. Some special assumption *Consider eddy current losses for bonded cables* are made which leads to slightly different results. This option should be used in case of verification of the CIGRE TB 880 examples or in order to use specific assumption.

### 35.3 Cable Ampacity Report

The *Cable Ampacity Report* () command is used to print cable ampacity reports. This command is independent of the calculation, the only requirement is the valid cable ampacity results file. The command shall determine which options are available for the selected results file. The following inputs are available in the cable analysis report dialog.

- *Results*: Pointer to the *Result File* (*ElmRes*).
- *Report*: Contains options whether to show additional data for the ampacity calculation.
- *Used format*: Pointer to the *Form Manager* to be used for the selected results file and report.

The report shows the calculated max. current. Furthermore it lists up some additional variables like the thermal resistances and the temperatures. Figure 35.2.1 can be used to check out the physical representative of some variables.

# Chapter 36

# Power Quality and Harmonics Analysis

## 36.1 Introduction

One of the many aspects of power quality is the harmonic content of voltages and currents. Harmonics can be analysed in either the frequency domain, or in the time-domain with post-processing using Fourier Analysis. The *PowerFactory* harmonics functions allow the analysis of harmonics in the frequency domain. The following functions are provided by *PowerFactory*:

- [36.2 Harmonic Load Flow](#) (including harmonic load flow according to IEC 61000-3-6 [16] and flicker analysis according to IEC 61400-21 [17])
- [36.3 Frequency Sweep](#)

*PowerFactory*'s harmonic load flow calculates harmonic indices related to voltage or current distortion, and harmonic losses caused by harmonic sources (usually non-linear loads such as current converters). Harmonic sources can be defined by either a harmonic current spectrum or a harmonic voltage spectrum. In the harmonic load flow calculation, *PowerFactory* carries out a steady-state network analysis at each frequency at which harmonic sources are defined.

A special application of the harmonic load flow is the analysis of ripple-control signals. For this application, a harmonic load flow can be calculated at one specific frequency only.

The harmonic load flow command also offers the option of calculating long- and short-term flicker disturbance factors introduced by wind turbine generators. These factors are calculated according to IEC standard 61400-21 [17], for wind turbine generators under continuous and switching operations.

In contrast to the harmonic load flow, *PowerFactory*'s frequency sweep performs a continuous frequency domain analysis. A typical application of the frequency sweep function is the calculation of network impedances. The result of this calculation facilitates the identification of series and parallel resonances in the network. These resonance points can identify the frequencies at which harmonic currents cause low or high harmonic voltages. Network impedances are of particular importance in applications such as filter design.

For both, the harmonic load flow and frequency sweep analysis, commands the option to consider contingencies is available. This allows to analyse the network in all available fault cases.

*PowerFactory* provides a toolbar for accessing the different harmonic analysis commands. This toolbar can be displayed (if not already active) by clicking the *Change Toolbox* ▾ button and selecting Power Quality and Harmonic Analysis.

The *Power Quality and Harmonic Analysis* toolbar provides icons to access four pre-configured command dialogs:

-  *Harmonic Load Flow*
-  *Impedance Frequency Characteristic (Frequency Sweep)*
-  *Flickermeter*
-  *Connection Request Assessment*

These command dialogs can also be accessed via *PowerFactory*'s main menu:

- *Calculation* → *Power Quality and Harmonics Analysis* → *Harmonic Load Flow...*
- *Calculation* → *Power Quality and Harmonics Analysis* → *Impedance Frequency Characteristic...*
- *Calculation* → *Power Quality and Harmonics Analysis* → *Flickermeter...*
- *Calculation* → *Connection Request Assessment...*

The Harmonic Load Flow and Frequency Sweep functions and their usage are described in this chapter. The Flickermeter function is described in Chapter 38, the Connection Request Assessment is described in Chapter 37.

## 36.2 Harmonic Load Flow

To calculate a harmonic load flow, click on the *Calculate Harmonic Load Flow* icon  to open the dialog for the Harmonic Load Flow (*ComHldf*) command.

For a detailed description of how harmonic injections are considered by *PowerFactory*, refer to Section 36.4 (Modelling Harmonic Sources), in which the analysis and the major harmonic indices are described. A full list of available result variables and their definition is given in the dedicated documentation about [Result Variables for Harmonics Analysis](#). The following sections describe the options available in the harmonic load flow command.

### 36.2.1 Basic Options Harmonic Load Flow

#### 36.2.1.1 Network Representation

**Balanced.** In the case of a symmetrical network and balanced harmonic sources, characteristic harmonics either appear in the positive sequence component (7th, 13th, 19th, etc.), or in the negative sequence component (5th, 11th, 17th harmonic order, etc.). Therefore, at all frequencies a single-phase equivalent (positive or negative sequence) can be used for the analysis.

For calculations where the checkbox “Only positive sequence” is checked, the calculation for the harmonic orders being naturally in the negative sequence will be performed with the positive sequence impedance.

**Unbalanced, 3-phase (ABC).** For analysing non-characteristic harmonics (3rd-order, even-order, inter-harmonics), unbalanced harmonic injections, or harmonics in non-symmetrical networks, the *Unbalanced, 3-phase (ABC)* option for modelling the network in the phase-domain should be selected.

### 36.2.1.2 Calculate Harmonic Load Flow

**Single Frequency.** Selecting this option will perform a single harmonic load flow calculation at the given *Output Frequency* (parameter name: *fshow*) or at the given *Harmonic Order* (parameter name: *ifshow*). A common application for this input mode is the analysis of ripple control systems. The results of the analysis are shown in the single line diagram, in the same way as for a normal load flow at the fundamental frequency.

**All Frequencies.** Selecting this option will perform harmonic load flow calculations for all frequencies for which harmonic sources are defined. These frequencies are gathered automatically prior to the calculation. The results for all frequencies are stored in a results object, which can be used to create bar chart representations of harmonic indices (see also Section 19.8 (Plots)). The results of the analysis at the given *Output Frequency* are shown in the single line diagram.

**Range of harmonic orders.** Selecting this option will perform harmonic load flow calculations for all frequencies within the defined range for which harmonic sources are defined.

### 36.2.1.3 Nominal Frequency, Output Frequency, Harmonic Order

**Nominal Frequency.** The harmonics analysis function in *PowerFactory* can only calculate harmonics of AC-systems with identical fundamental frequencies. The relevant nominal frequency must be entered here (usually 50 Hz or 60 Hz).

**Output Frequency.** This is the frequency for which order specific results may be displayed in the single-line graphic or selected in a flexible data page. In the case of a *Single Frequency* calculation, this is the frequency for which a harmonic load flow is calculated. When the option *All Frequencies* is selected, this parameter only affects the display of results in the single line diagram or in flexible data pages. It does not influence the calculation itself. A change made to the *Output Frequency* will cause the *Harmonic Order* to be automatically changed accordingly.

**Harmonic Order.** This is the same as the *Output Frequency* but input as the *Harmonic Order* (f/fn). The *Harmonic Order* multiplied by the *Nominal Frequency* always equals the *Output Frequency*. Both floating-point and integer values are valid as inputs. A change made to the *Harmonic Order* will cause the *Output Frequency* to be automatically changed accordingly.

### 36.2.1.4 Consider contingencies

**Consider contingencies.** When selected, harmonic load flow calculations for all defined fault cases within the linked *Contingency Analysis* command are executed in addition to the base case (no contingency). The results for each contingency are recorded in a dedicated result object which is further described in Section 36.2.2.

---

**Note:** Only the calculation method *AC Load Flow Calculation* is supported for the contingency analysis within harmonic load flow calculations

---

---

**Note:** Only static contingencies are supported for the harmonic load flow calculation. The different options shown in the command are described in Section 27.4.1.2.

---

---

**Note:** The creation of contingency cases is described in Section 27.2.1.1.

---

### 36.2.1.5 Calculate Flicker

**Calculate Flicker.** When selected, the long- and short-term flicker disturbance factors are calculated according to IEC standard 61400-21. See Section [36.5 \(Flicker Analysis \(IEC 61400-21\)\)](#) for more detailed information.

### 36.2.1.6 Calculate Sk at Fundamental Frequency

**Calculate Sk at Fundamental Frequency.** When selected, the short-circuit power, Sk, and impedance angle, psik, are calculated at the point of common coupling (PCC) for all 3-phase buses in the network being analysed. This calculation is only performed at the fundamental frequency. For an unbalanced harmonic load flow, impedance ratios (as described in Section [36.6.2.1 \(Calculation of Impedance Ratios\)](#)) at 3-phase buses are also calculated. See Section [36.6 \(Short-Circuit Power\)](#) for more detailed information.

### 36.2.1.7 Load Flow

This displays the load flow command used by the calculation. Click on the arrow button → to inspect and/or adjust the load flow command settings.

The Load Flow settings define the temperature, for which the load flow is calculated. The same temperature is used for the Harmonic Load Flow calculation, mentioned in the dialog under “Temperature Dependency: Line/Cable Resistances”.

## 36.2.2 Results/Output

**Results.** This option is available if *Calculate Harmonic Load Flow* option *All Frequencies or Range of harmonic orders* has been selected. It is used to select the target results object for storing the results of the harmonic load flow. See Section [36.7 \(Definition of Result Variables\)](#) for more information regarding specifying and defining result variables.

The result object for the *Harmonic Load Flow* which is linked → in the command consists of sub-result objects:

- *Order specific*: contains order specific results for all recorded variables at each calculated frequency (e.g. the harmonic distortion *m:HD*).
- *Summary*: contains all summary results for all calculated frequencies (e.g. the total harmonic distortion *m:THD*).

When contingencies are considered, *order specific* result objects are created for each contingency as well as the base case. The *summary* result object contains the summary results for each contingency and the base case as well.

**Summary results.** Shows the recording options for the summary results.

- The button *Variable selection* shows all summary result variables that are recorded in the harmonic load flow calculation. New variable selections can be defined according to Section [36.7.1](#).
- The button *Add default variables* adds all default variables to the selected result file. Separated by classes, the default summary variables are:
  - Terminals (*ElmTerm*): *m:THD* (Total harmonic distortion in %)

- The option *Record if THD is above X % (referred to limit)* is only available when contingencies are considered and allows the user to only record summary results when they exceed a user-defined limit referred to the defined *Harmonic distortion limits*.
  - *Example:* Record if THD is above 50 % (referred to limit, which is for example 5 %), then the summary results for an element are only recorded if its THD is above 2.5 %.

**Order specific results.** Shows the recording options for the order specific results.

- The button *Variable selection* shows all order specific result variables that are recorded in the harmonic load flow calculation. New variable selections can be defined according to Section 36.7.1.
- The button *Add default variables* adds all default variables to the selected result file. Separated by classes, the default order specific variables are:
  - Terminals (*ElmTerm*): *m:HD* (Harmonic distortion in %)
- The option *Record if HD is above X % (referred to limit)* is only available when contingencies are considered and allows the user to only record order specific results when they exceed a user-defined limit referred to the defined *Harmonic distortion limits*.
  - *Example:* Record if HD is above 100 % (referred to limit, which is 1.5 % for the 5th), then the order specific results for an element are only recorded if its HD for the 5th is above 1.5 %.

**Harmonic distortion limits.** This option is shown when contingencies are considered (*Basic Options* page). It contains a selection of voltage-level dependent harmonic distortion limits which can be used to limit the recording of the defined result parameters. When used, only results over a specified limit are recorded. The *Harmonic distortion limits* are described in Section 36.2.5.1.

**Settings.** Contains the selection of harmonic limits (*SetDislm*) that are used to limit the recorded parameters. For each row, a nominal voltage along with the corresponding distortion limits can be selected. The *Distortion Limits* objects (*SetDisfc*) contain HD and THD limits.

---

**Note:** User-defined *Distortion Limits* objects (*SetDisfc*) can be created in the project under *Settings → Limits*. Objects located here can be selected in the voltage-dependent harmonic limit selection in the harmonic load flow command.

---

**Record variables required for “Filter Analysis”.** When selected, all required variables for the “Filter Analysis” reports are recorded. A list of all default variables for shunts/filters is given in Section 36.7.

### 36.2.3 IEC 61000-3-6

#### Treatment of Harmonic Sources

The alpha exponent values on this page will only be considered by the harmonic load flow (that is to say that the calculation will be carried out according to the IEC 61000-3-6 standard [16]) if at least one harmonic source in the network is defined as IEC 61000 (see Section 36.4.1 (IEC 61000 Harmonic Sources)). On this page, if *According to IEC 61000-3-6* is selected, these tables display the alpha exponent values as given in the IEC 61000-3-6 standard, as read-only values. If *User Defined* is selected, the definition of the alpha exponent values is user-definable in terms of integer and/or non-integer harmonic orders.

### 36.2.4 Advanced Options Harmonic Load Flow

#### 36.2.4.1 Calculate HD and THD

**Based on fundamental frequency values.** All values are based on fundamental frequency values, as defined by various standards.

**Based on rated voltage/current.** All values are based on the rated voltage/current of the buses and branches in the network, respectively.

#### 36.2.4.2 Max. harmonic order for calculation of THD and THF

The harmonic order up to which RMS values are summed in the calculation of THD and THF.

#### 36.2.4.3 Calculate Harmonic Factor (HF)

The calculation of the harmonic factor is optional as it requires extra processing time. Leave unticked if performance is critical.

#### 36.2.4.4 Calculate R,X at output frequency for all nodes

Calculates the impedance for all buses at the defined output frequency. Leave unticked if performance is critical.

#### 36.2.4.5 Calculation of Factor-K (BS 7821) for Transformers

Exponent used in calculation of factor-K (according to BS 7821) for transformers. This exponent depends on the construction of the transformer and is usually available from the manufacturer. Typical values lie between 1.5 - 1.7.

### 36.2.5 Harmonic Load Flow Result Analysis

#### 36.2.5.1 Harmonic Distortion Limits

In order to make a detailed harmonic load flow result analysis, distortion limits need to be defined. Harmonic distortion limits are linked in the *Harmonic Load Flow* command, the *Harmonic Load Flow Plot* command and *Harmonic Load Flow Tabular Reports* command. The definition consists of two objects, the *Selection of Harmonic Limits (SetDislm)* and *Distortion Limits (SetDisfc)*.

##### **Selection of Harmonic Limits (*SetDislm*):**

The object contains a table to assign voltage dependent distortion limits where a *Standard* with a corresponding *Category* can be assigned to a *Nominal voltage range*. The range is defined for the voltage levels greater than the value of the *Nominal voltage range*. For example, if the DIN EN 5160 (2020) for medium voltage category should be considered, the *Nominal voltage range* needs to be 1 kV. This means, the standard will not be applied for terminals with nominal voltages of 1 kV or less.

##### **Distortion Limits (*SetDisfc*):**

*Distortion Limits* objects are linked in the column *Category* in the *Selection of Harmonic Limits (SetDislm)* and contain HD limits for defined harmonics orders as well as a THD limit for all orders.

- A *Variable* can be defined for which the defined limits apply. The limits only apply for this variable and can therefore only be assigned to plots displaying this variable.
- A *Factor* can be specified which is multiplied with the defined limits.

All available *Distortion Limits* for different standards are located in the *Data Manager* under: System\Standard\Settings\Limits\...

User-defined *Distortion Limits* can be created in the project under ...\\Settings\\Limits\\

### 36.2.5.2 Harmonic Distortion Plot

In order to display the harmonic distortion levels corresponding with each harmonic order, harmonic distortion plots can be used. Harmonic distortion results can be displayed after the execution of a harmonic load flow calculation. The Data Series of this plot type is described in Section 19.8.2.

The selection of harmonic limits can be done on the same page of the data series dialog as the curve definition, and the fill style and colours for the harmonic distortion limits are user-defined. Figure 36.2.1 shows an example of a harmonic distortion plot.

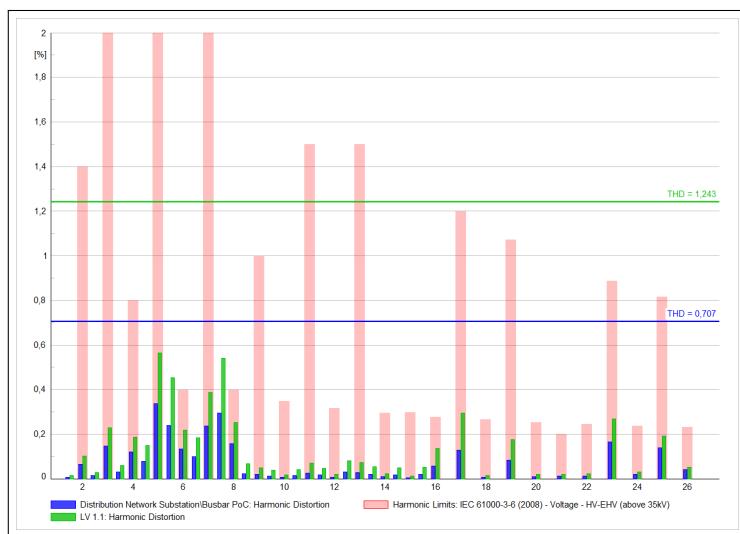


Figure 36.2.1: Harmonic Distortion Plot

The *Harmonic Distortion Plot* can be automatically created for different terminals and contingencies by using the *Create Distortion Plot* command described in Section 36.2.5.4.

### 36.2.5.3 Waveform Plot

The waveform plot is used to display the waveform of a voltage or a current following a harmonic load flow calculation. The harmonics are typically emitted by a harmonic voltage or current source, as described in Section 36.4 (Modelling Harmonic Sources).

The waveform is calculated according to the following formula:

$$u(t) = \sum_{i=1}^n u(i) \cdot \cos(2\pi(f(i) \cdot t + \varphi(i))) \quad (36.1)$$

where:

$i$	Index of frequency
$n$	Number of frequencies
$t$	Time
$f(i)$	Frequency at index i
$u(i)$	Magnitude at frequency i
$\phi(i)$	Angle at frequency i

If a successful harmonic load-flow calculation with the option *All Frequencies* is performed, the waveform plot will show the results of any distorted or pure sinusoidal variable, e.g. voltages or currents, from any element in the network.

The waveform plot can be created even if the preceding harmonic load-flow is not active anymore. An example plot of harmonic distortion is shown in Figure 36.2.2.

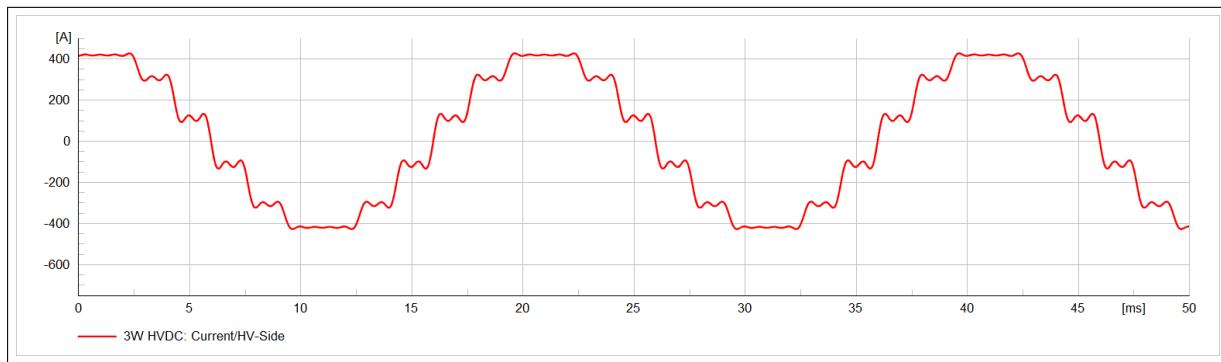


Figure 36.2.2: Use of the Waveform Plot to display Harmonic Distortion

The variable definition requires a reference to the results object and the element as per the curve plot, but in contrast to it, the magnitude of the variable and the angle relating to the magnitude can also be defined.

The appropriate angle is automatically matched to the selected magnitude, if such angle is available in the results and if the variable is a voltage or a current. When no appropriate angle is found, one may be selected manually. Although the angle can be defined, the parameter is not obligatory.

### The Waveform Plot Settings

The Data Series of this plot type is described in Section 19.8.2. The usage, settings and tools for this plot type are similar to the curve plot, described in Chapter 19: Reporting and Visualising Results, Section 19.8.9.1, however there are some specific settings unique to the waveform plot, these include the step size and time range. The step size and time range are specified within the waveform plot settings object stored in the “Settings” directory of the active project.

To change the waveform plot settings, open the data series plot dialog and edit the relevant *Waveform* plot settings object located in the *Function* column of the *Curves* table, either by double clicking or by choosing *right-click → Edit*. A common *Settings Waveform Plot* object (*SetWave*) is used to define the *Step Size* and the displayed time *Range* for all curves shown in the plot (see Figure 36.2.3).

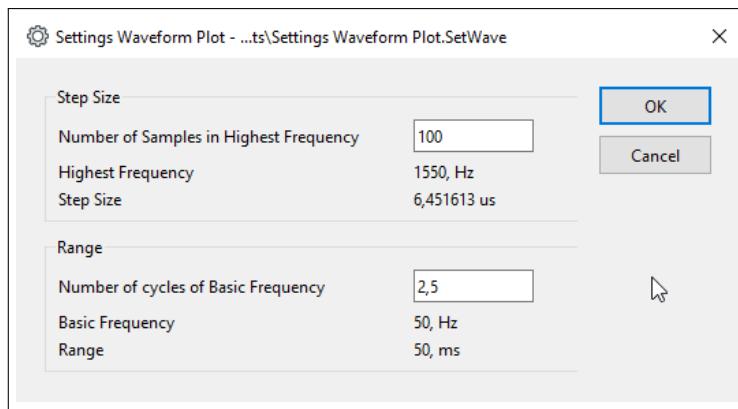


Figure 36.2.3: The waveform plot settings dialog

### Step Size

The visible waveforms are calculated by the waveform plot itself.

The *Number of Samples in Highest Frequency* defines how many intermediate points of the highest frequency are considered for the drawing of the waveform. In order to get a smooth waveform values above 20 are recommended. The resulting *Step Size* is mentioned in the dialog together with the *Highest Frequency* calculated in the Harmonic Load Flow calculation.

### Range

To be independent of the basic frequency, the time range of the waveform is entered in *Number of cycles of Basic Frequency*. *Basic Frequency* and the resulting *Range* are shown for information.

#### 36.2.5.4 Create Distortion Plot

The *Create Distortion Plot* command  allows the automatic creation of harmonic distortion plots for selected busbars/terminals and contingencies. With the generic *Insert Plot* icon  the plots can be created from scratch.

**Results.** The used result object for generating plots can be selected here for either balanced or unbalanced AC harmonic load flow calculations. By default, the first found result object from the active study case for the corresponding calculation method is used.

**Plot type.** Two different plot types are available to be created automatically:

- Distortion plot
- Waveform plot

**Selection.** A selection of recorded *Elements* as well as *Contingencies* that shall be plotted can be made here.

---

**Note:** Selections can also be made directly from certain table reports by using the *Create distortion plot* button as described in Section 36.2.5.5.

---

**Variable name.** The available variables for the automatic creation of harmonic distortion plots can be selected here. Only recorded variables are shown here.

**Magnitude.** The available magnitude variables for the automatic creation of harmonic waveform plots can be selected here. Only recorded variables are shown here.

**Angle.** The available angle variables for the automatic creation of harmonic waveform plots can be selected here. Only recorded variables are shown here.

**Advanced Options.** Additional options for the generation of plots can be selected on the *Advanced Options* page.

- *Show harmonic distortion limits:* When selected, the linked *Selection of Harmonic Limits* is used in the plot creation as well. For a terminal the harmonic limits of its corresponding nominal voltage is taken. The *Harmonic distortion limits* are described in Section 36.2.5.1.

---

**Note:** Harmonic distortion limits can only be displayed if the variable of the distortion limit matches the plotted variable.

---

- *Display each element in a separate plot:* When not selected, the results of all elements are shown in the same plot.

When selected, the results of each busbar/terminal are shown in a separate plot. The defined *maximum number of plots per page* specifies how many plots are shown per plot page.

---

**Note:** While the results of recorded elements can be plotted on separate plots, the results of all selected contingencies are always shown in the same plot.

---

### 36.2.5.5 Harmonic Load Flow Tabular Reports

Several reports are available for the *Harmonic Load Flow* calculation via the corresponding icon  in the toolbox. The *Harmonic Load Flow Reports* provide a means to summarise and examine system conditions over the simulated frequency range with and without considered contingencies.

The command consists of the following sections:

**Basic Options.** The used result file for generating reports is linked here as well as the voltage-dependent *Harmonic distortion limits* that are displayed in the reports. The *Harmonic distortion limits* are described in Section 36.2.5.1.

**Overview.** Different reports can be selected that provide a good result overview.

- *Harmonic distortion (HD) violations:* lists all HD violations at each recorded terminal and additionally for each contingency which is considered.
- *Harmonic distortion (HD) summary for a busbar/terminal:* gives a detailed summary for one terminal considering all analysed contingencies. For each harmonic order, the maximum HD value is compared to its limit and the base case distortion. The number of contingencies that violate the limit are additional given for each order. Only available when contingencies are considered.
- *Highest total harmonic distortion (THD) for busbars/terminals:* lists the highest THD value at each terminal along with the contingency where it occurs. The number of contingencies that violate the THD limit is also given. Can be filtered by a *threshold*. Only available when contingencies are considered.
- *Highest total harmonic distortion (THD) for contingencies:* reports the maximum THD value along with the terminal for each contingency. Only available when contingencies are considered.
- *Failed contingencies:* lists all contingencies for which the calculation failed. Only available when contingencies are considered.

**Note:** Some reports include a *Create distortion plot* button, which can be used to create a plot directly from the report. To do so, mark rows in the table that include terminals and contingencies that shall be plotted and press the *Create distortion plot* button. All terminals and contingencies from the selection will be transferred to the *Harmonic Distortion Plot* command to create HD plots which is further described in Section 36.2.5.4.

---

**Detailed.** Reports with detailed results for all terminals and considered contingencies can be selected on this page.

- *Total harmonic distortion (THD) for busbars/terminals*: gives the total harmonic distortion at each recorded terminal for a selected contingency which is additionally compared to the base case. Can be filtered by a *threshold*.
- *Harmonic order specific results for terminals*: reports the harmonic distortion for each order and at each recorded terminal for a selected contingency

**Filter Analysis.** Reports for all shunts and filters in the network can be selected on this page.

- *Shunt/filter layout*: gives an overview of all shunts and filters in the network and lists layout parameters of the used components for the fundamental and harmonics. Not available for contingencies.
  - *Shunt/filter results*: reports detailed results for all shunts and filters in the network for each recorded frequency. Different considered contingencies can be selected.
- 

**Note:** If a report is missing values, please check the settings on the *Results/Output* page in the *Harmonic Load Flow* command to verify if filters for recording results are defined. More information in Section 36.2.2.

---

### 36.2.5.6 Load Harmonic Load Flow Results

To reload results of a specific harmonic order and contingency (if considered) into the network, the *Load Harmonic Load Flow Results* icon  can be used.

**Results.** The used result object to load results can be selected here for either balanced or unbalanced AC harmonic load flow calculations. By default, the first found result object from the active study case for the corresponding calculation method is used.

**Contingency.** The contingency which is to be considered can be selected here. If no contingency is selected, then the results of the base case are loaded.

**Results to be loaded.** Options for which results are loaded into the network can be selected here.

- *Summary results only*: Only recorded summary results (e.g. THD) are loaded into the network.
- *Summary and order specific results*: All recorded summary results as well as order specific results of a selected *Harmonic order* are loaded into the network. The selection contains all recorded harmonic orders.

## 36.3 Frequency Sweep

To calculate frequency dependent impedances, the impedance characteristic can be computed for a given frequency range using the Frequency Sweep command (*ComFsweep*). This function is available by clicking on the *Calculate Frequency Impedance Characteristic* icon  available in the *Harmonics* toolbar.

Harmonic analysis by frequency sweep is normally used for analysing self- and mutual- network impedances. However, it should be noted that not only self- and mutual-impedances can be analysed and shown. The voltage source models (*ElmVac*, *ElmVacbi*) available in *PowerFactory* allow the definition of any spectral density function. Hence, impulse or step responses of any variable can be calculated in the frequency domain. Furthermore, a *Frequency Dependent Network Equivalent Data* element is used to represent equivalent impedances and admittances between buses (refer to Section 36.7.2).

The most common application is the analysis of series and parallel resonance problems for filter design purpose, power quality analyses or to verify oscillations seen in EMT simulations.

For the purpose of investigating the impact of capacitances on the network impedance and its resonances, additional result variables of the impedance are available for a pure resistive/inductive network, where all capacitive components will be neglected.

All analysis can be extended by considering contingencies as well in order to investigate different fault cases along with the base case.

The following sections describe the options available in the harmonic frequency sweep command.

### 36.3.1 Basic Options Frequency Sweep

#### 36.3.1.1 Network Representation

**Balanced, positive sequence.** This option uses a single-phase, positive sequence network representation, valid for balanced symmetrical networks. A balanced representation of unbalanced objects is used.

**Unbalanced, 3-phase (ABC).** This option uses a full multiple-phase, unbalanced network representation.

By default any balanced or unbalanced frequency sweep is initialised with a load flow calculation (ticked “Load Flow Initialisation” setting). The load flow calculation is used to determine for example tap positions of capacitor banks, shunt reactors, filters and transformers, where controls are configured and where the load flow results have an impact on the impedances involved. If the checkbox is not set, the network is considered with actual positions for tap changers, consumption values for loads, dispatch power for generators, etc. and the frequency characteristic of the network impedance at each point of interest is evaluated based on this actual state of the network and its components.

#### 36.3.1.2 Impedance Calculation

The frequency sweep will be performed for the frequency range defined by the *Start Frequency* and the *Stop Frequency*, using the given *Step Size*. The *Automatic Step Size Adaptation* option allows an adaptive step size. Enabling this option will normally speed up the calculation, and enhance the level of detail in the results by automatically using a smaller step size when required. The settings for step size adaptation can be changed on the *Advanced Options* tab.

#### 36.3.1.3 Nominal Frequency, Output Frequency, Harmonic Order

**Nominal Frequency.** This is the fundamental frequency of the system, and the base frequency for the harmonic orders (usually 50 Hz or 60 Hz).

**Output Frequency.** This is the frequency for which the results in the single line diagram are shown. This value has no effect on the actual calculation.

**Harmonic Order.** This is the harmonic order equivalent of the *Output Frequency*. The *Harmonic Order*

multiplied by the *Nominal Frequency* always equals the *Output Frequency*. Both floating-point and integer values are valid as inputs.

#### 36.3.1.4 Consider contingencies

When selected, frequency sweep calculations for all defined fault cases within the linked *Contingency Analysis* command are executed in addition to the base case (no contingency). The results for each contingency are recorded in a dedicated sub result object.

---

**Note:** Only the calculation method *AC Load Flow Calculation* is supported for the contingency analysis within harmonic load flow calculations

---

**Note:** Only static contingencies are supported for the harmonic load flow calculation. The different options shown in the command are described in Section [27.4.1.2](#).

---

**Note:** The creation of contingency cases is described in Section [27.2.1.1](#).

---

#### 36.3.1.5 Result Variables and Load Flow

**Result Variables.** Used to select the target results object which will store the results of the harmonic frequency sweep. See Section [36.7 \(Definition of Result Variables\)](#) for more information regarding specifying result variables.

The result object for the *Frequency Sweep* which is linked → in the command consists of sub-result objects:

- *Base case*: for the calculation without contingency.
- *Contingencies*: for each contingency a dedicated result object with a corresponding name is created to store the calculation results.

**Load Flow.** This displays the load flow command used by the calculation. Click on the arrow button to inspect and/or adjust the load flow command settings.

The Load Flow settings define the temperature, for which the load flow is calculated. The same temperature is used for the frequency sweep, mentioned in the dialog under “Temperature Dependency: Line/Cable Resistances”. In case that the “Load Flow Initialisation” checkbox is not ticked, the temperature can be selected manually in the frequency sweep dialog. The available temperature settings correspond to the ones from the load flow command dialog.

The results of *PowerFactory*'s frequency sweep analysis are the characteristics of the impedances over the frequency range.

### 36.3.2 Advanced Options Frequency Sweep

Selecting the option *Automatic Step Size Adaptation* on the *Basic Data* page of the frequency sweep command is one way to increase the speed of the calculation. This option enables the use of the step size adaptation algorithm for the frequency sweep. With this algorithm, the frequency step between two calculations of all variables is not held constant, but is adapted according to the shape of the sweep. When no resonances in the impedance occur, the frequency step can be increased without

compromising accuracy. If the impedance starts to change considerably with the next step, the step size will be reduced again. The frequency step is set such that the prediction error will conform to the two prediction error input parameters, as shown below:

- *Maximum Prediction Error* (typical value: 0.01)
- *Minimum Prediction Error* (typical value: 0.005)
- *Step Size Increase Delay* (typically 3 frequency steps)

**Calculate R, X at output frequency for all nodes.** *PowerFactory* calculates the equivalent impedance only at selected nodes over the complete analysed frequency range. When this option is selected, following the harmonic calculation, the equivalent impedance is available for all nodes, but only for the output frequency.

### 36.3.3 Frequency Sweep Result Analysis

#### 36.3.3.1 Plots

With the help of plots, the results of the *Frequency Sweep* calculation can be analysed conveniently. There are two plot types that are generally used:

- *Frequency Sweep Curve*: The selected impedance variable is plotted over the investigated frequency range. The plot type *Curve Plot* is used as described in Section [19.8.9.1](#).
- *R-X plot*: The real and imaginary part of the impedance are plotted against one another. It is often labeled as “Impedance Loci”. The plot type *XY Plot* is used as described in Section [19.8.9.3](#).

---

**Note:** In the data series of the R-X plot, the generic option *Specify time range* can be used to limit the plotted frequency range. The start and end time correspond to the start and end harmonic order for the frequency sweep plot.

---

Both plots can be created automatically, also including contingencies, with the *Create frequency sweep plots* command as described in Section [36.3.3.2](#).

#### 36.3.3.2 Create frequency sweep plots

The *Frequency Sweep Plot* command  allows the automatic creation of frequency curves for the impedance and its phase angle at selected busbars/terminals and contingencies. In addition, R-X plots can be created to display the impedance in the R-X plane. With the generic *Insert Plot* icon  the plots can be created from scratch.

**Results.** The used result object for generating plots can be selected here for either balanced or unbalanced AC calculations. By default, the first found result object from the active study case for the corresponding calculation method is used.

**Frequency sweep curve.** When selected, frequency sweep curves can be created for the available impedance variables. Only recorded variables for busbars/terminals (*ElmTerm*) are shown here.

**R-X plot.** When selected, R-X plots can be created for the available variables of the real and imaginary part of the impedance. Only recorded variables for busbars/terminals (*ElmTerm*) are shown here.

**Selection.** A selection of recorded *Busbars/Terminals* as well as *Contingencies* that shall be plotted can be made here.

---

**Note:** Selections can also be made directly from table reports by using the *Create frequency sweep plot* button as described in Section 36.3.3.3.

**Advanced Options.** Additional options for the generation of plots can be selected on the *Advanced Options* page.

- *Display each busbar/terminal in a separate plot:* When not selected, the results of all busbars/terminals are shown in the same plot.

When selected, the results of each busbar/terminal are shown in a separate plot. The defined *maximum number of plots per page* specifies how many plots are shown per plot page.

**Note:** While the results of busbars/terminals can be plotted on separate plots, the results of all selected contingencies are always shown in the same plot.

### 36.3.3.3 Frequency Sweep Tabular Reports

Several reports are available for the *Frequency Sweep* calculation via the corresponding icon  in the toolbox. The *Frequency Sweep Reports* provide a means to analyse the impedance and its resonances over the simulated frequency range with and without considered contingencies.

The command consists of the following sections:

**Basic Options.** The used result file for generating reports is linked here. The report for *Failed contingency cases* can be selected as well.

**Resonances.** Different reports can be selected that provide a good result overview.

- *All resonances:* reports all occurring resonances over the analysed frequency range for each recorded terminal and contingency.
- *Resonances in a frequency range:* resonances for each recorded terminal and contingency can be filtered according to the frequency range of interest which can be defined by a *Start* and *Stop Frequency*.
- *Resonances near existing harmonic injections:* takes existing harmonic injections from sources in the network into account and lists all resonances which are close to the existing injections. All resonances within the specified *allowed frequency band around investigated orders* will be listed in the report.

For the *Resonance detection* available variables can be selected. A resonance is defined as local maximum (parallel resonance) or minimum (series resonance) of the impedance where the sign of the corresponding impedance angle changes.

**Note:** Some reports include a *Create frequency sweep plot* button, which can be used to create a plot directly from the report. To do so, mark rows in the table that include terminals and contingencies that shall be plotted and press the *Create frequency sweep plot* button. All terminals and contingencies from the selection will be transferred to the *Create frequency sweep plots* command to create impedance plots which are further described in Section 36.3.3.2.

### 36.3.3.4 Load Frequency Sweep Results

To reload results of a specific harmonic order and contingency (if considered) into the network, the *Load Frequency Sweep Results* icon  can be used.

**Results.** The used result object to load results can be selected here for either balanced or unbalanced frequency sweep calculations. By default, the first found result object from the active study case for the corresponding calculation method is used.

**Contingency.** The contingency which is to be considered can be selected here. If no contingency is selected, then the results of the base case are loaded.

**Harmonic Order.** Specify for which harmonic order the results shall be loaded into the network. Any number (harmonic order) can be used in the input field. When it does not exist in the selected result object, the *nearest order* is detected and used in the loading command.

## 36.4 Modelling Harmonic Sources

Every switched device produces harmonics and should therefore be modelled as a harmonic source. In *PowerFactory*, harmonic sources can be either current sources or voltage sources. The models listed in Table 36.4.1 can be used to generate harmonics.

The spectrum type can either be a current spectrum, defined by the linked type (*TypHmccur*) or a voltage spectrum defined in the element itself or a linked type (*TypHmcvolt*). The list also highlights those models where a type assignment is not mandatory to consider the element as harmonic source. In those cases the model itself calculates the injected harmonic spectrum based on the preceding load flow calculation and the resulting firing angle of the power electronics circuit. Apart from the user defined assignment of a spectrum, an idealised and a more realistic spectrum considering overlap angles can be selected.

Element	Object class	Spectrum type	Model spectrum	Remarks
General Load	<i>ElmLod</i>	I		
MV Load	<i>ElmLodmv</i>	I		Norton Equivalent
Rectifier/Inverter	<i>ElmRecmono</i> , <i>ElmRec</i>	I	✓	
PWM Converter	<i>ElmVscmono</i> , <i>ElmVsc</i>	I/(U)		Voltage source with series reactor or Norton Equivalent
AC Current Source	<i>Elmlac</i> , <i>Elmlaci</i>	I		Inner Admittance
Static Generator	<i>ElmGenstat</i> , <i>ElmPvsys</i>	I		Norton Equivalent
Static Var System	<i>ElmSvs</i>	I	✓	
TCSC	<i>ElmTcsc</i>	I	✓	
HVDC LCC	<i>ElmHvdclcc</i>	I, I	✓, ✓	Separate definitions for inverter and rectifier side of the HVDC link
Doubly-Fed Induction Machine	<i>ElmAsm</i> , <i>ElmAsmsc</i>	I		Norton Equivalent or pure current source
Synchronous machine	<i>ElmSym</i>	U		Spectrum defined in type
AC Voltage Source	<i>ElmVac</i> , <i>ElmVacbi</i>	U		Spectrum defined in element
External Grid	<i>ElmXnet</i>	U		Spectrum defined in element

Table 36.4.1: List of models which can be configured as harmonic sources

See Section 36.4.1 ([Definition of Harmonic Injections](#)) for information on how to define harmonic injec-

tions for these sources based on the spectrum type.

---

**Note:** Harmonic injections can be modelled in EMT simulations using the Fourier source object. For further details refer to the [Technical References Document](#).

---

### 36.4.1 Definition of Harmonic Injections

When defining the spectrum via the *Harmonic Sources* type object, the harmonic infeeds can be entered according to one of three options: *Balanced, Phase Correct* or *Unbalanced, Phase Correct*, or *IEC 61000*. The *Harmonic Sources* object is a *PowerFactory* 'type' object, which means that it may be used by many elements who have the same basic type. Multiple current source loads may, for example, use the same *Harmonic Sources* object. Note that *PowerFactory* has no corresponding element for this type.

#### Phase Correct Harmonic Sources

For the *Balanced, Phase Correct* harmonic sources option, in both balanced and unbalanced harmonic load flows, the magnitudes and phases of positive and negative sequence harmonic injections at odd integer harmonic orders can be defined. This facilitates the representation of typical power system odd-order harmonics (without including triplen).

For the *Unbalanced, Phase Correct* harmonic sources option, the magnitudes and phases for the available phases of the harmonic injections at integer and non-integer harmonic orders can be defined. In the case of a balanced harmonic load flow, harmonic injections in the zero sequence are not considered, and harmonic injections at non-integer harmonic orders are considered in the positive sequence. In the case of an unbalanced harmonic load flow, harmonic injections in the zero sequence and at non-integer harmonic orders are considered appropriately. See Table 36.4.3 for a complete summary.

#### IEC 61000 Harmonic Sources

The IEC 61000-3-6 standard [16] describes a “second summation law”, applicable to both voltage and current, which is described mathematically as:

$$U_h = \sqrt[\alpha]{\sum_{m=0}^N |U_{m,h}|^\alpha} \quad (36.2)$$

where  $U_h$  is the resultant harmonic voltage magnitude for the considered aggregation of  $N$  sources at order  $h$ , and  $\alpha$  is the exponent as given in Table 36.4.2 [16].

$m$  are the harmonic injections where  $m = 0$  is the vectorial sum of all phase correct injections and  $m = 1, 2, \dots, N$  are IEC injections. The calculation process of the resulting harmonic magnitude can be described with the following process:

1. First, for a given harmonic order consider a superposition of all the currents and voltages arising in the network produced by all phase correct sources. The injections of all phase correct sources are considered together. For a given location the current (or voltage) arising at that location from each phase correct source is vectorially summed to the resulting magnitude ( $m = 0$ ).
2. Then, consider each harmonic source connected to the network in isolation (with the other sources disconnected). We only loop over individual sources if they are modelled according to IEC61000; that is, these are truly considered “in-turn”. For each considered harmonic IEC source ( $m = 1, 2, \dots, N$ ), at each harmonic order, the connected source will inject a current into the network and

this current will be distributed throughout the network according to the impedance and topology of each branch of the network. Parallel units within a model are considered as separate sources.

3. Finally, for a given harmonic order, a superposition of all the currents and voltages arising in the network produced by all phase correct sources and each IEC source is computed. For each given location the magnitude of all phase correct contributions ( $m = 0$ ) is taken into account, but additionally for each given location the current (or voltage) arising at that location from each IEC source ( $m = 1, 2, \dots, N$ ) is summed using the second summation law (according to IEC 61000-3-6).

Alpha Exponent Value	Harmonic Order
1	$h < 5$
1.4	$5 \leq h \leq 10$
2	$h > 10$

Table 36.4.2: IEC 61000-3-6 Summation Exponents According to Harmonic Order

The *Harmonic Sources* type set to option *IEC 61000* allows the definition of integer and non-integer harmonic current magnitude injections. In the case of balanced and unbalanced harmonic load flows, zero sequence order injections and non-integer harmonic injections are considered in the positive sequence. This is summarised in Table 36.4.3. It should be noted that in order to execute a harmonic load flow according to IEC 61000-3-6, **at least one harmonic source in the network must be defined as IEC 61000**.

Table 36.4.4 describes the consideration of the sequence components of the harmonic orders for the AC voltage source (*ElmVac*, *ElmVacbi*), external grid (*ElmXnet*) and synchronous machine (*ElmSym*).

Additionally, the voltage source allows the following to be input for use in the Frequency Sweep calculation:

- Spectral density of voltage magnitude;
- Spectral density of voltage angle;
- Frequency dependencies (in the form of a *Frequency Polynomial Characteristic*). See Section 36.4.3 ([Frequency Dependent Parameters](#)) and Chapter 18 ([Parameter Characteristics, Load States, and Tariffs](#)) for further details.

## Background Harmonics

*PowerFactory* facilitates the modelling of background harmonics. This is done using either the external grid element (*ElmXnet*) or the AC voltage source element (*ElmVac*, *ElmVacbi*), on their respective *Power Quality/Harmonics* pages. If only the harmonic voltage amplitude is known (and not the angle), the *IEC 61000* option can be selected. Table 36.4.4 describes the consideration of the sequence components of the harmonic orders for the AC voltage source and external grid.

## Selection of Type of Harmonic Sources

The *Harmonic Sources* object (*TypHmccur*) is independent of whether the harmonic source is a voltage or a current source. The decision as to whether harmonic sources are fed into the system as harmonic voltages or as harmonic currents is made exclusively by the element to which the *Harmonic Sources* object is assigned. The consideration by the calculation of the sequence components of harmonic injections is given in Table 36.4.3.

## Magnitudes and Phase Values

The quantities of the spectrum type are defined as percentage based on a reference value. The reference value for current (or voltage) can be selected in the element, which uses the linked harmonic spectrum. For phase correct spectra, either the fundamental frequency result, determined through the preceding load flow calculation, or the rated value of the element can be chosen as reference. For

spectra acc. to IEC 61000, the reference is always the rated value. The injected current at frequency  $f_h$  of an IEC source is therefore defined by:

$$I_h = k_h \cdot I_r \quad (36.3)$$

where  $k_h = I_h/I_r$  is the harmonic content defined in the spectrum and  $I_r$  is the rated current of the source element. As phase angle information is not available, only magnitudes can be used for the calculation.

For phase correct spectra, the actual harmonic current of a current source at frequency  $f_h$  is calculated by:

$$I_h = k_h \cdot e^{\Delta\varphi_h} \cdot I_1 \cdot e^{h \cdot \varphi_1} \quad (36.4)$$

where

$$k_h = \begin{cases} I_h/I_1 & \text{if balanced} \\ I_{ah}/I_{a1} & \text{if unbalanced phase a} \\ I_{bh}/I_{b1} & \text{if unbalanced phase b} \\ I_{ch}/I_{c1} & \text{if unbalanced phase c} \end{cases} \quad (36.5)$$

$$\Delta\varphi_h = \begin{cases} \varphi_h - h \cdot \varphi_1 & \text{if balanced} \\ \varphi_{ah} - h \cdot \varphi_{a1} & \text{if unbalanced phase a} \\ \varphi_{bh} - h \cdot \varphi_{b1} & \text{if unbalanced phase b} \\ \varphi_{ch} - h \cdot \varphi_{c1} & \text{if unbalanced phase c} \end{cases} \quad (36.6)$$

The values at the fundamental frequency,  $I_1$  and  $\varphi_1$ , are taken from a preceding load flow calculation if the spectrum is applied based on the fundamental frequency value. If the spectrum is applied based on the rated value, instead of the load flow result the rated current of the source element is used as base ( $I_1 = I_r$  in equations 36.4 and 36.5) and only  $\varphi_1$  is taken from the load flow calculation. A normal load flow calculation is therefore required prior to a harmonic load flow calculation.

In the case of balanced systems in which only characteristic harmonics of orders 5, 7, 11, 13, 17, etc. occur, the option *Balanced, Phase Correct* should be selected in the *Type of Harmonics Sources* section. In this context, 'Balanced' refers to characteristic harmonics. In the balanced case, the harmonic frequencies are determined by the program and only characteristic orders are available in the spectrum to assign values to (note that in the unbalanced case, the harmonic frequencies can be freely-defined).

For harmonic sources which produce non-characteristic, unbalanced or inter-harmonics, the option *Unbalanced, Phase Correct* should be set in the *Type of Harmonics Sources* section. In the *Unbalanced, Phase Correct* case, the harmonic frequency, magnitude and phase angle of each phase can be chosen individually for each harmonic frequency. This mode therefore caters for every possible kind of harmonic source.

The problem commonly arises as to how one can represent the harmonic content in a system which differs from the native modal system (positive, negative or zero sequence system). The following example illustrates how to represent the 3rd harmonic in a positive or negative sequence system (as opposed to the native zero sequence system).

In the symmetrical case, the phase shift between the three phases is:

$$\begin{aligned} A &: 0^\circ \\ B &: -120^\circ \\ C &: +120^\circ (-240^\circ) \end{aligned}$$

For harmonics of order  $n$ :

$$\begin{aligned}A &: 0^\circ \\B &: -n * 120^\circ \\C &: +n * 120^\circ\end{aligned}$$

Taking the 3rd harmonic as an example:

$$\begin{aligned}A &: 0^\circ (= 0^\circ) \\B &: -360^\circ (= 0^\circ) \\C &: +360^\circ (= 0^\circ)\end{aligned}$$

Consequently, the 3rd harmonic in the ideally balanced case is only effective in the zero sequence, as its native modal system. For representing 3rd harmonics (and multiples thereof) in the positive sequence system, the following phase correction needs to be entered:

$$\begin{aligned}A &: 0^\circ \\B &: +(n - 1) * 120^\circ \\C &: -(n - 1) * 120^\circ\end{aligned}$$

Again taking the 3rd harmonic as an example:

$$\begin{aligned}A &: 0^\circ (= 0^\circ) \\B &: -360^\circ + 240^\circ (= -120^\circ) \\C &: +360^\circ - 240^\circ (= +120^\circ)\end{aligned}$$

<b>Harmonic Load Flow Command Setting</b>	<b>Harmonic Current Source Type</b>	<b>Sequence Components of Harmonic Injections</b>
<i>Balanced</i>	<i>Balanced, Phase Correct</i>	<ul style="list-style-type: none"> <li>Positive (e.g. 7, 13, 19, ...), negative (e.g. 5, 11, 17, ...);</li> <li>Typical integer orders only.</li> </ul>
	<i>Unbalanced, Phase Correct</i>	<ul style="list-style-type: none"> <li>Positive (e.g. 4, 7, 10, ...), negative (e.g. 2, 5, 8, ...);</li> <li>Triplen harmonics (e.g. 3, 6, 9, ...) are ignored and non-integer harmonic orders (e.g. 5.5, 6.2, 8.35, ...) are considered in the positive sequence.</li> </ul>
	<i>IEC 61000</i>	<ul style="list-style-type: none"> <li>Positive (e.g. 4, 7, 10, ...), negative (e.g. 2, 5, 8, ...);</li> <li>Triplen harmonics and non-integer harmonics are considered in the positive sequence.</li> </ul>
<i>Unbalanced</i>	<i>Balanced, Phase Correct</i>	<ul style="list-style-type: none"> <li>As for balanced harmonic load flow.</li> </ul>
	<i>Unbalanced, Phase Correct</i>	<ul style="list-style-type: none"> <li>Positive, negative, zero sequence;</li> <li>Integer and non-integer harmonics.</li> </ul>
	<i>IEC 61000</i>	<ul style="list-style-type: none"> <li>As for balanced harmonic load flow.</li> </ul>

Table 36.4.3: Consideration of Sequence Components of Harmonic Injections for *TypHmccur*

Harmonic Load Flow Command Setting	ElmVac/ElmXnet/ElmSym Setting	Sequence Components of Harmonic Injections
<i>Balanced</i>	<i>Phase Correct</i>	<ul style="list-style-type: none"> <li>Positive (i.e. 4, 7, 10, ...), negative (i.e. 2, 5, 8, ...);</li> <li>Non-integer harmonic orders (i.e. 5.5, 6.2, 8.35, ...) are considered in the positive sequence.</li> <li>Triplen harmonics (i.e. 3, 6, 9, ...) are ignored (with warning).</li> </ul>
	<i>IEC 61000</i>	<ul style="list-style-type: none"> <li>Positive, negative;</li> <li>Triplen harmonics and non-integer harmonics are in the positive sequence.</li> </ul>
<i>Unbalanced</i>	<i>Phase Correct</i>	<ul style="list-style-type: none"> <li>Positive, negative, zero sequence;</li> <li>Non-integer harmonics are considered.</li> </ul>
	<i>IEC 61000</i>	<ul style="list-style-type: none"> <li>As for balanced harmonic load flow.</li> </ul>

Table 36.4.4: Consideration of Sequence Components of Harmonic Injections for AC Voltage Source, External Grid and Synchronous Machine

### 36.4.2 Assignment of Harmonic Injections

The assignment of harmonic injections to the elements listed in Table 36.4.1 is done via the individual element's dialog on the *Power Quality/Harmonics* page containing the following settings:

- Harmonic Currents:** Used to assign the *Harmonic Sources* type (*TypHmccur*).
- Type of Harmonic Sources:** Displays the type of harmonic source selected in the assigned *Harmonic Sources* type (*TypHmccur*).
- Harmonic current referred to.** For phase correct sources, the harmonic current may be referred to either the fundamental current or the rated current. If the harmonic current source type has been selected to be IEC 61000, the harmonic current is always referred to the rated current and this option is read-only.

**Note:** When a **Norton Equivalent** is used, be aware that the injected harmonic currents at the point of connection differ from the harmonic injections (percentage values) entered in the *Harmonic Source* (*TypHmccur*). The harmonic injections are defined for the ideal (inner) current source. Due to the Norton Admittance, the observed harmonic current at the terminal is lower than the harmonic current the ideal (inner) source produces.

Harmonic injections defined for external grids (*ElmXnet*) and voltage sources (*ElmVac*, *ElmVacbi*) are implicitly assigned, as they are defined on the elements' respective *Power Quality/Harmonics* pages. No further assignment is therefore necessary. See Section 36.4.1 (Definition of Harmonic Injections) for further information.

### 36.4.3 Frequency Dependent Parameters

Due to the skin effect and variations in internal inductance, resistances and inductances are usually frequency dependent. This can be modelled in *PowerFactory* by associating a “frequency characteristic” with these quantities. Two types of characteristic may be used: either a *Frequency Polynomial Characteristic (ChaPol)* as illustrated in Figure 36.4.1, or a user-defined frequency table (*TriFreq* and *ChaVec*). These kinds of characteristics are then assigned via the *Power Quality/Harmonics* page of the corresponding element’s dialog, as illustrated by the example in Figure 36.4.2 for a line type.

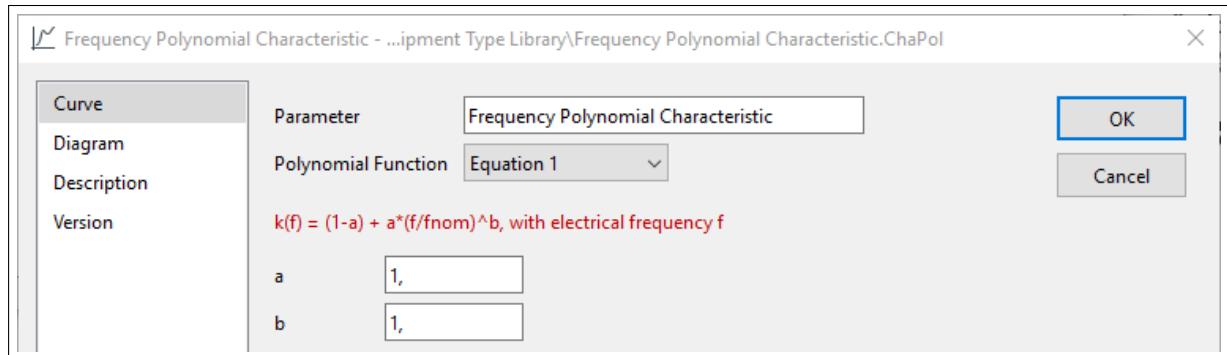


Figure 36.4.1: The Frequency Polynomial Characteristic (*ChaPol*)

For the frequency polynomial characteristic object illustrated in Figure 36.4.1, the formula given by (36.7) is used:

$$y(f_h) = (1 - a) + a \cdot \left(\frac{f_h}{f_1}\right)^b \quad (36.7)$$

The parameters *a* and *b* are specified in the *Frequency Polynomial Characteristic* dialog. Variable *y* is usually expressed as a percentage of the corresponding input parameters. For example, the resulting line resistance is given by (36.8):

$$R(f_h) = R \cdot y(f_h) \quad (36.8)$$

An example of the use of the polynomial characteristic for a line type is shown in Figure 36.4.2.

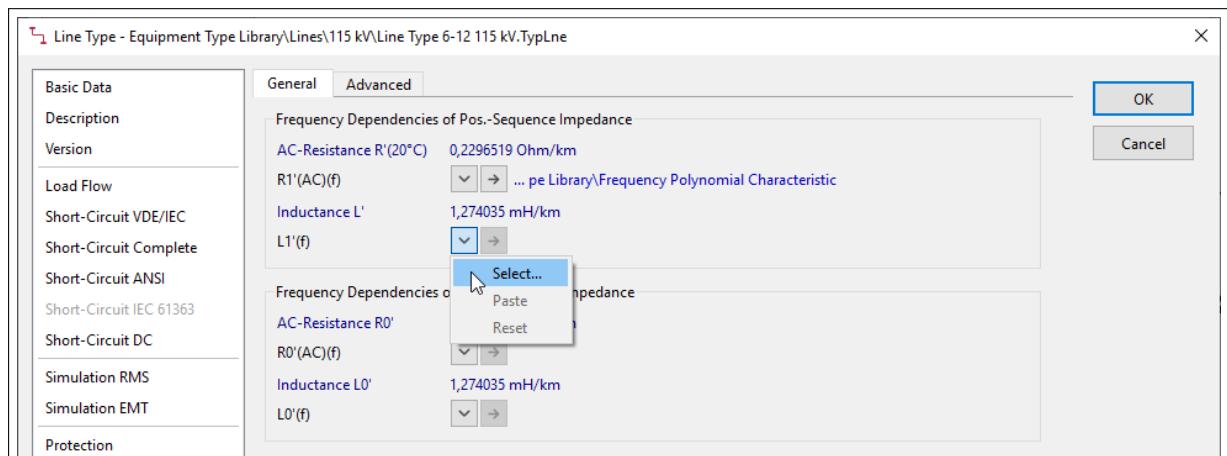


Figure 36.4.2: Frequency Dependencies in a Line Type

It is also possible to define frequency dependent characteristics using a vectorial parameter characteristic (*ChaVec*).

The following objects can have frequency dependent parameters defined using a frequency characteristic.

Lines, Cables and Series Impedances:

- Line type (*TypLne*)
- Series Reactor, Resistor, Capacitor (*ElmSind*, *ElmScap*)
- Series RLC-Filter (*ElmSfilt*)

Transformers and Grounding Elements:

- 2-, 3-, 4-winding transformers (*TypTr2*, *TypTr3*, *TypTr4*)
- Step-Voltage Regulator (*TypVoltreg*)
- NEC/NER (*ElmNec*)

Generators and Loads:

- Synchronous machine type (*TypSym*)
- Asynchronous machine types (*TypAsmo*, *TypAsm1*)
- Static Generator (*ElmGenstat*, *ElmPVsys*)
- Complex load type (*TypLodind*)
- Medium Voltage Load (*ElmLodmv*)

Sources:

- External Grid (*ElmXnet*)
- AC voltage source, single and double<sup>1</sup> port (*ElmVac*, *ElmVacbi*)
- AC current source, single<sup>1</sup> and double<sup>1</sup> port (*ElmIac*, *ElmIacbi*)

Shunts and Filters:

- Shunt/Filter (*ElmShnt*)
- Harmonic Filter (*ElmFilter*)

Power Electronic Devices:

- PWM Converter, single and double port (*ElmVscmono*, *ElmVsc*)
- Thyristor Controlled Series Capacitor - TCSC (*ElmTcsc*)

Frequency-dependent impedances are automatically considered for lines represented by either a tower type (*TypTow*, *TypGeo*) or a cable system type (*TypCabsys*).

## 36.5 Flicker Analysis (IEC 61400-21)

The IEC standard 61400-21 [17] describes the measurement and assessment of power quality characteristics of grid connected wind turbine generators (WTGs). One of these power quality characteristic parameters pertains to voltage fluctuations. Voltage fluctuations can produce undesirable effects on the

---

<sup>1</sup>Frequency dependencies on spectral densities, considered only in the Frequency Sweep

consumer side which may manifest as 'flicker' (visible flickering effects from light sources), and voltage changes (voltage magnitude being too high or too low).

Even though the IEC 61400-21 standard is for wind turbines, the approach for assessing flicker is also used for other areas, such as photovoltaic parks or industrial applications.

However, the approach is only valid in upstream direction of radial networks. Typically, the approach is used for assessment of the flicker disturbance at the point of common coupling (PCC) of a power plant (e.g. a wind park), for which it has been designed and fits. In downstream direction or in meshed networks, the approach does not give valid results.

In the assessment of a WTG's power quality in terms of voltage fluctuations, the operation of WTGs can be subdivided into two modes: continuous operation and switching operations (see Sections 36.5.1 ([Continuous Operation](#)) and 36.5.2 ([Switching Operations](#)) for definitions). These modes of operation are considered by the *PowerFactory* flicker calculation, which calculates the short-term and long-term flicker disturbance factors. See Section 36.5.6 ([Flicker Results Variables](#)) for a list of the flicker results variables available. The calculation of flicker is performed optionally as part of the harmonic load flow command. For a detailed description of how to configure and execute a harmonic load flow, including the calculation of flicker, refer to Section 36.2.1 ([Basic Options Harmonic Load Flow](#)).

### 36.5.1 Continuous Operation

Continuous operation is defined in IEC standard 61400-21 as the normal operation of the wind turbine generator (WTG) excluding start-up and shut-down operations. The short-term and long-term flicker disturbance factors during continuous operation are defined in [17] as:

$$P_{st} = P_{lt} = c(\psi_k, v_a) \cdot \frac{S_n}{S_k} \quad (36.9)$$

where  $P_{st}$  is the short-term flicker disturbance factor;  $P_{lt}$  is the long-term flicker disturbance factor;  $c$  is the flicker coefficient for continuous operation;  $\psi_k$  is the network impedance angle (degrees);  $v_a$  is the average annual wind speed (m/s);  $S_n$  is the rated apparent power of the wind turbine (VA); and  $S_k$  is the short-circuit apparent power of the grid (VA).

- The short-circuit power  $S_k$  is calculated under normal operating conditions at each bus in the network according to Section 36.6. Therefore, also the impedance of loads and generation units in the network is taken into account.
- At each terminal to which a flicker source is connected, the complex impedance is calculated between this terminal and all other terminals in the network. The network impedance angle  $\psi_k$  is derived from this calculation in order to determine the flicker coefficient  $c$ .

When more than one WTG exists at the point of common coupling (PCC), the summed short-term and long-term flicker disturbance factors for continuous operation are described in [17] as:

$$P_{st\Sigma} = P_{lt\Sigma} = \frac{1}{S_k} \sqrt{\sum_{i=1}^{N_{wt}} (c(\psi_k, v_a) \cdot S_{n,i})^2} \quad (36.10)$$

where  $N_{wt}$  is the number of wind turbine generators at the PCC.

### 36.5.2 Switching Operations

Switching operations are defined in IEC standard 61400-21 as start-up or switching between wind turbine generators (WTGs). In this mode of operation, the short-term and long-term flicker disturbance

factors during switching operations are defined in [17] as:

$$P_{st} = 18 \cdot N_{10}^{0.31} \cdot k_f(\psi_k) \cdot \frac{S_n}{S_k} \quad (36.11)$$

where  $N_{10}$  is the number of switching operations in a 10-minute period;  $k_f$  is the flicker step factor;  $\psi_k$  is the network impedance angle (degrees);  $S_n$  is the rated apparent power of the wind turbine (VA); and  $S_k$  is the short-circuit apparent power of the grid (VA).

$$P_{lt} = 8 \cdot N_{120}^{0.31} \cdot k_f(\psi_k) \cdot \frac{S_n}{S_k} \quad (36.12)$$

where  $N_{120}$  is the number of switching operations in a 120-minute period;  $k_f$  is the flicker step factor;  $\psi_k$  is the network impedance angle (degrees);  $S_n$  is the rated apparent power of the wind turbine (VA); and  $S_k$  is the short-circuit apparent power of the grid (VA).

When more than one WTG exists at the PCC, the short-term flicker disturbance factor under switching operations is defined in [17] as:

$$P_{st\Sigma} = \frac{18}{S_k} \left[ \sum_{i=1}^{N_{wt}} N_{10,i} \cdot (k_{f,i}(\psi_k) \cdot S_{n,i})^{3.2} \right]^{0.31} \quad (36.13)$$

Likewise, the long-term flicker disturbance factor under switching operations is defined as:

$$P_{lt\Sigma} = \frac{8}{S_k} \left[ \sum_{i=1}^{N_{wt}} N_{120,i} \cdot (k_{f,i}(\psi_k) \cdot S_{n,i})^{3.2} \right]^{0.31} \quad (36.14)$$

where  $N_{wt}$  is the number of WTGs at the PCC.

The relative voltage change (in units of %) due to the switching operation of a single WTG is computed as [17]:

$$d = 100 \cdot (k_u(\psi_k) \cdot \frac{S_n}{S_k}) \quad (36.15)$$

### 36.5.3 Flicker Contribution of Wind Turbine Generator Models

The calculation of flicker according to IEC standard 61400-21 in *PowerFactory* considers flicker contributions of the following generator models:

- Static generator (*ElmGenstat*)
- Asynchronous machine (*ElmAsm*)
- Doubly-fed asynchronous machine (*ElmAsmsc*)

In order for these models to be able to contribute flicker, their flicker contributions must first be defined and assigned, as described in Sections 36.5.4 (Definition of Flicker Coefficients) and 36.5.5 (Assignment of Flicker Coefficients).

### 36.5.4 Definition of Flicker Coefficients

Flicker coefficients are defined in *PowerFactory* by means of the *Flicker Coefficients* type (*TypFlicker*), as illustrated in Figure 36.5.1. When created, this is stored by default in the *Equipment Type Library* folder in the project tree.

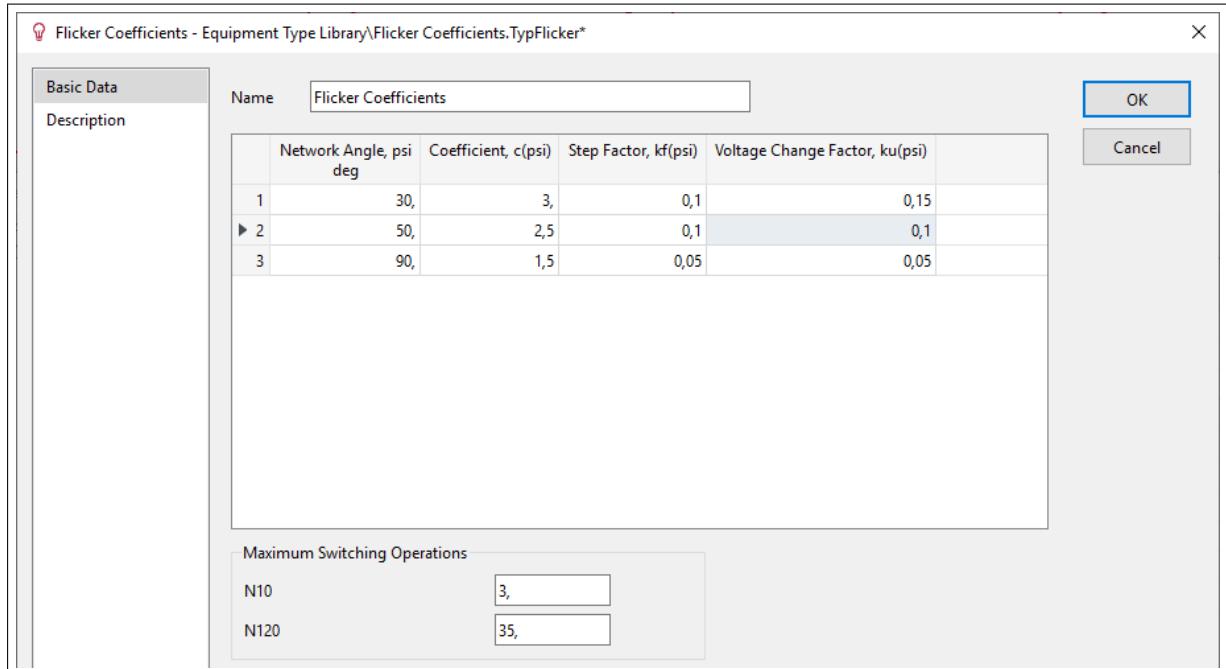


Figure 36.5.1: Definition of Flicker Coefficients using the Flicker Coefficients Type (*TypFlicker*)

The *Flicker Coefficients* type allows the input of six parameters (all of which are defined in IEC standard 61400-21):

- **Network Angle, psi (degrees).** This is the network impedance angle and must be entered in either the range [-180,180] (default), or [0,360]. Any mix of these ranges is not permitted. Network angles must be entered in ascending order.
- **Coefficient, c(psi).** The flicker coefficient as a function of the network impedance angle.
- **Step Factor, kf(psi).** The flicker step factor as a function of the network impedance angle.
- **Voltage Change Factor, ku(psi)** The voltage change factor as a function of the network impedance angle.
- **Maximum Switching Operations: N10.** The maximum number of switching operations in a 10-minute period.
- **Maximum Switching Operations: N120.** The maximum number of switching operations in a 120-minute period.

### 36.5.5 Assignment of Flicker Coefficients

The *Harmonics* page of these elements' dialogs contains a *Flicker Contribution* section or separate tab which allows the assignment of *Flicker Coefficients*.

If the reference selection *Flicker Coefficients* is left unassigned, the generator is then considered to be an ideal source for the flicker calculation.

### 36.5.6 Flicker Results Variables

Following the calculation of flicker according to IEC 61400-21, the following result variables for every node in the network are available in the single line graphic. It should be noted that *PowerFactory* calculates these flicker disturbance factors and relative voltage change for impedance angles with lines at 20 degrees Celsius and at maximum operation temperature. The following result variables are the worst-case values in the impedance angle range, which is based on the temperature range:

- $Pst\_cont$ ;  $Plt\_cont$ : short-term and long-term flicker disturbance factors for continuous operation of the wind turbine generator/s;
- $Pst\_sw$ ;  $Plt\_sw$ : short-term and long-term flicker disturbance factors for switching operations of the wind turbine generator/s;
- $d\_sw$ : relative voltage change (as a percentage).

For the mathematical definitions of these result variables, refer to Sections [36.5.1 \(Continuous Operation\)](#) and [36.5.2 \(Switching Operations\)](#).

## 36.6 Short-Circuit Power

For power quality assessment, the network impedance of the grid under normal operating conditions is usually used as the basis for calculations. This impedance is represented by the short-circuit power,  $Sk$ , of the grid. Hence, for power quality assessment at a point  $V$  in the grid, this short-circuit power,  $SkV$ , under normal operating conditions is used instead of the short-circuit power of the faulted grid according to short-circuit calculations. If the Harmonic Load Flow command option *Calculate Sk at Fundamental Frequency* is enabled (see Section [36.2.1 \(Basic Options Harmonic Load Flow\)](#)), the short-circuit power of the grid under normal operation is available in the calculation results.

### 36.6.1 Balanced Harmonic Load Flow

For the balanced harmonic load flow, the calculation of the short-circuit power,  $Sk$ , at each bus is as follows:

$$Sk = \frac{1}{|Z_{bus}|} \quad (\text{MVA}) \quad (36.16)$$

where  $Z_{bus}$  is the impedance calculated at the bus.

The calculation of the impedance angle,  $psik$ , at each bus is as follows:

$$psik = \angle Z_{bus} \quad (\text{degrees}) \quad (36.17)$$

where  $Z_{bus}$  is the impedance calculated at the bus.

### 36.6.2 Unbalanced Harmonic Load Flow

For the unbalanced harmonic load flow, the calculation of the short-circuit power,  $Sk$ , at each bus is as follows:

$$Sk = \frac{1}{|Z1_{bus}|} \quad (\text{MVA}) \quad (36.18)$$

where  $Z1_{bus}$  is the positive sequence impedance calculated at the bus.

The calculation of the impedance angle,  $psik$ , at each bus is calculated as follows:

$$psik = \angle Z1_{bus} \quad (\text{degrees}) \quad (36.19)$$

where  $Z1_{bus}$  is the positive sequence impedance calculated at the bus.

### 36.6.2.1 Calculation of Impedance Ratios

The following impedance ratios are calculated following an unbalanced harmonic load flow (if option *Calculate Sk at Fundamental Frequency* has been selected):

$$z2tz1kV = \frac{|Z2|}{|Z1|} \quad (36.20)$$

$$x0tx1kV = \frac{X0}{X1} \quad (36.21)$$

$$r0tx0kV = \frac{R0}{X0} \quad (36.22)$$

### 36.6.3 Sk Result Variables

Following either a balanced or an unbalanced harmonic load flow calculation with the option *Calculate Sk at Fundamental Frequency* selected, the following result variables are available for every 3-phase bus in the network:

- $SkV$ : short-circuit power (MVA)
- $psikV$ : impedance angle (degrees)

For the mathematical definitions of these result variables, refer to Section [36.6.1 \(Balanced Harmonic Load Flow\)](#), and [36.6.2 \(Unbalanced Harmonic Load Flow\)](#).

Following an unbalanced harmonic load flow with the option *Calculate Sk at Fundamental Frequency* selected, the following additional result variables are available for every 3-phase bus in the network:

- $z2tz1kV$ : Impedance ratio
- $x0tx1kV$ : Impedance ratio
- $r0tx0kV$ : Impedance ratio

For the mathematical definitions of these impedance ratio result variables, refer to Section [36.6.2.1 \(Calculation of Impedance Ratios\)](#).

### 36.6.4 Short-Circuit Power of the External Grid

The external grid element, *ElmXnet*, allows the calculation of the network impedance to be based on the short-circuit power,  $Sk$ , and impedance angle,  $psik$ . This option can be selected in the external grid element on the *Power Quality/Harmonics* page, by using the *Use for calculation* drop-down box and selecting  $Sk$ . User input fields are then available for the short-circuit power,  $Sk$  (MVA), impedance angle,  $psik$  (deg), and impedance ratios  $z2tz1kV$ ,  $x0tx1kV$ , and  $r0tx0kV$ .

The impedance of the external grid, which is taken into account for power quality assessment, is calculated internally based on either the short-circuit power,  $Sk$ , at normal operation; the maximum short-circuit power,  $Sk''max$  for faulted operation; or the minimum short-circuit power  $Sk''min$  for faulted operation, depending on the user's selection.

Data for input parameters ( $SkV$ ,  $psikV$ ,  $z2tz1kV$ ,  $x0tx1kV$ , and  $r0tx0kV$ ) can first be calculated from a detailed network model using the Harmonic Load Flow command option *Calculate Sk at Fundamental Frequency* (refer to Section [36.6.3 \(Sk Result Variables\)](#)), performed, for example, by the network operator. A third party, (i.e. a wind farm planner) could get this information for the point of common coupling (PCC for the planned wind farm) from the network operator. The planner can then enter the data into the external grid element, which is a simplified representation of the network as seen from the PCC.

## 36.7 Definition of Result Variables

In order to record the results of either the *Harmonic Load Flow* or *Frequency Sweep* calculation, the variables of interest must be defined. However, for each of these calculations, a small selection of variables are recorded by default in the results object defined on each command's *Basic Data* page by the *Result Variables* parameter. For the *Harmonic Load Flow* the following variables are recorded by default:

- Harmonic order (-);
- Frequency (Hz);
- HD (%) (for terminals);
- Voltage across inductor (p.u.) ( $u_{rl}$ ) (for shunts/filters);
- Voltage across capacitor (p.u.) ( $u_c$ ) (for shunts/filters);
- Current through inductor (A) ( $i_{lL}$ ) (for shunts/filters);
- Current through resistor  $R_p$  (A) ( $i_{Rp}$ ) (for shunts/filters);
- Current through capacitor C (A) ( $i_C$ ) (for shunts/filters);
- Voltage across capacitor  $C_1$  (p.u.) ( $u_{c1}$ ) (for shunts/filters);
- Voltage across capacitor  $C_2$  (p.u.) ( $u_{c2}$ ) (for shunts/filters);
- Voltage across resistor  $R_p$  (p.u.) ( $u_{rp}$ ) (for shunts/filters);
- Voltage across inductor (kV) ( $U_{rl}$ ) (for shunts/filters);
- Voltage across capacitor (kV) ( $U_c$ ) (for shunts/filters);
- Voltage across capacitor  $C_1$  (kV) ( $U_{c1}$ ) (for shunts/filters);
- Voltage across capacitor  $C_2$  (kV) ( $U_{c2}$ ) (for shunts/filters);
- Voltage across resistor  $R_p$  (kV) ( $U_{rp}$ ) (for shunts/filters);
- Voltage across inductor  $L_1$  (p.u.) ( $u_{rl1}$ ) (for harmonic filters);
- Voltage across resistor  $R_{p1}$  (p.u.) ( $u_{rp1}$ ) (for harmonic filters);
- Current through inductor  $L_1$  (A) ( $i_{l1}$ ) (for harmonic filters);
- Current through resistor  $R_{p1}$  (A) ( $i_{Rp1}$ ) (for harmonic filters);
- Current through capacitor  $C_1$  (A) ( $i_{C1}$ ) (for harmonic filters);
- Voltage across inductor  $L_2$  (p.u.) ( $u_{rl2}$ ) (for harmonic filters);
- Voltage across resistor  $R_{p2}$  (p.u.) ( $u_{rp2}$ ) (for harmonic filters);
- Current through inductor  $L_2$  (A) ( $i_{l2}$ ) (for harmonic filters);
- Current through resistor  $R_{p2}$  (A) ( $i_{Rp2}$ ) (for harmonic filters);
- Current through capacitor  $C_2$  (A) ( $i_{C2}$ ) (for harmonic filters);
- Voltage across inductor  $L_1$  (kV) ( $u_{rl1}$ ) (for harmonic filters);
- Voltage across resistor  $R_{p1}$  (kV) ( $u_{rp1}$ ) (for harmonic filters);
- Voltage across inductor  $L_2$  (kV) ( $u_{rl2}$ ) (for harmonic filters);
- Voltage across resistor  $R_{p2}$  (kV) ( $u_{rp2}$ ) (for harmonic filters);

For the *Frequency Sweep*, the following variables are recorded by default:

- Harmonic order (-);
- Frequency in Hz (Hz);

In order to define additional variables to be recorded, a two-step process is required of firstly creating the desired *Variable Selection* and then selecting the variables for recording within these sets. These steps are described in section [Definition of Variable Selections](#).

### 36.7.1 Definition of Variable Selections

To define a *Variable Selection*, right-click on a network component (or multi-select several network components and right-click), either in the single-line diagram or in the Data Manager, and select the option *Results Variables* → *Harmonic Load Flow...*; or *Results Variables* → *Frequency Sweep...*. This will add a new (but still empty) variable selection for the selected object to the results object (referred to by parameter *Result Variables* on the *Basic Options* page of the *Harmonic Load Flow* or *Frequency Sweep* command dialog).

All results of harmonic analysis, with the exception of the harmonic load flow using option *Single Frequency* (for which no results are recorded), are stored in a normal results object (*ElmRes*). This results object stores the result variables against the frequency for which they were calculated. For more information about the results object, see Section [19.7 \(Results Objects\)](#).

To access the variable selection objects, click on the *Edit Result Variables* button (=+) on the main toolbar. There are two instances of this button: one associated with the *Harmonic Load Flow* and one associated with the *Frequency Sweep*. Select the button associated with the relevant calculation.

The variable selection manager dialog will open which displays the result file type. If the list is empty, a new variable selection can also be defined by clicking on the *New Object* button (+) and the variables defined. Each variable selection contains the variables of interest for a single object.

The Variable Selection object is described in Section [19.3: Variable Selection](#).

### 36.7.2 Definition of Frequency Dependent Network Equivalent Data

The purpose of this element is to provide a means to access the frequency dependent network equivalent self- and mutual- impedances and admittances of the selected buses.

A *Frequency Dependent Network Equivalent Data* element (*ElmEquivalent*) can be defined by right-clicking on multiple buses and selecting *Network Models* → *Freq. Dep. Network Equivalent Data* → *New...* from the context menu. The element dialog will then be displayed, with the bus/es as an entry in the shown table. Multiple buses can be selected/added to this table. The checkboxes for the “Oscillating generator” and “Influencing device” only need to be selected if the radially factor (RF) is to be computed. If this is the case, then one box should be checked in each column; the “Oscillating generator” terminal should be different from the “Influencing device” terminal.

The calculation results are the impedance and admittance matrices that represent the equivalent system of user-selected buses. In addition, for the purpose of evaluating the risk of device-dependent sub-synchronous oscillations (DDSO), a frequency-dependent network impedance based index known as the “radiality factor” (RF) can be calculated, as described in literature<sup>2</sup>.

$$RF(f) = \frac{2Y_{12}(f)}{Y_{11}(f) + Y_{22}(f)} \quad (36.23)$$

<sup>2</sup>M. S. Annakkage, C. Karawita and U. D. Annakkage, “Frequency Scan-Based Screening Method for Device Dependent Sub-Synchronous Oscillations”, in IEEE Transactions on Power Systems, vol. 31, no. 3, pp. 1872-1878, May 2016, doi: 10.1109/TPWRS.2015.2442653.

where

- $Y_{11}(f)$  equivalent frequency-dependent self-admittance seen at the oscillating generator rotor side (bus 1)  
 $Y_{22}(f)$  equivalent frequency-dependent self-admittance seen at the influencing device (bus 2)  
 $Y_{12}(f)$  frequency-dependent mutual admittance between bus 1 and bus 2

The resulting matrices must be defined and accessed using a zero-based index. The buses are ordered in the matrices in the order that they appear in the equivalent element table (the first row in the table corresponds to index 0, the second row to index 1 and so on). For example, assuming that two buses have been selected, one configuration of results can be as follows:

- Z:0:0:r The real part of the frequency dependent self-impedance (in Ohm) at the first bus (index 0) defined in the *Frequency Dependent Network Equivalent Data* element.  
Z:0:1:mag The magnitude of the frequency dependent mutual impedance (in Ohm) between the first (index 0) and second buses (index 1) defined in the *Frequency Dependent Network Equivalent Data* element.  
Y:1:1:phi The angle of the frequency dependent self-admittance (in degrees) of the second bus (index 1) defined in the *Frequency Dependent Network Equivalent Data* element.

# Chapter 37

## Connection Request Assessment

For power quality assessment, *PowerFactory* offers a Connection Request Assessment command ( *ComConreq*) and corresponding element ( *ElmConreq*). The Connection Request Assessment command, in conjunction with the Connection Request element, facilitates the execution of a power quality assessment according to the *Method* selected in the command.

### 37.1 How to execute a Connection Request Assessment

This section gives a small guideline and describes the steps for configuring and executing a new connection request assessment in *PowerFactory*.

1. Place the connection request assessment element
  - Use the corresponding element ( *ElmConreq*) from the drawing toolbox to create a new connection request element.
  - The node to which the connection request is connected is used as the point of connection by default. However, a user-defined junction point can be selected in the *Connection topology* tab of each calculation method in the connection request element.
2. Configure the connection request element
  - Basic Data page
    - *Consider in Connection Request Assessment* - The relevant guidelines must be selected here. It is recommended to only select one guideline.
    - *Consider in other calculations* - With this selection it is possible to decide with the values of which guideline the connection request element should be considered in other calculations like Load Flow or Short-Circuit calculations.
    - *Power factor definition* - The power factor of the entire installation/power plant can either be defined with a single *Specific (minimum) value* or can be *Calculated* from the single load, generation and storage units.
  - Calculation method specific pages
    - The connection request element can be defined for the guidelines that are to be assessed. Each guideline has a dedicated page with all input parameters.
    - The definition pages for each guideline are separated into tabs. Each tab corresponds to a calculation and includes all necessary parameters.

- 
- Note:** Some general hints about the definition and assessment of connection requests:
- The *Installation/Power Plant definition* tab in each guideline is always necessary to be configured since it contains the basic definition.

- The other tabs only need to be configured if the calculation is to be executed, e.g. when no Flicker assessment is to be done, the corresponding page does not need to be modified.
  - For a very simplified assessment (loading and slow voltage change) only the *Installation/Power Plant definition* tab has to be configured.
  - For more information about the input parameters, consult the [Technical References Document](#) of the connection request element (*ElmConreq*).
- 

### 3. Execute the *Connection Request Assessment*

- To assess one or more connection requests in the network, open the *Connection Request Assessment* command (🔗).
- Select the guideline that is to be assessed and its edition or publication year.
- Optionally, it is possible to select specific *Selected Feeder(s)*. This means that only network elements in these feeders are taken into account for the assessment.
- Select the calculations that are to be executed.
- Select the *Load Flow type* that shall be used to evaluate voltage changes and loadings:
  - Standard Load Flow (see Chapter [25](#))
  - Low Voltage Load Flow (see Section [42.5](#))

### 4. Result evaluation

- Dedicated reports are available for each guideline in the *DlgSILENT Library* and can be directly accessed with the *Connection Request Assessment Report* icon (📄).
  - The reports can be created with the Report Generation command which is described in Section [37.6](#).
- 

**Note:** On the *Parameters* page of the Report Generation command, the details of the connection request report can be adapted.

---

## User-defined Limits

For the following standards, it is possible to select user-defined limits on the *Advanced* page of the connection request assessment:

- D-A-CH-CZ, Edition 2 and 3
- VDE-AR-N 4100/4105, Published 2018
- VDE-AR-N 4110, Published 2018 and 2023

For these standards, two options for the limits are available:

- **Default:** The default limits defined in the respective standard are taken for the assessment of the different calculations.
  - **User-defined:** The user can change the default limits defined in the respective standard and use its own user-defined limits for the assessment of the different calculations. For the D-A-CH-CZ, different limits can be selected for the HV, MV and LV voltage levels. It has to be noted that user-defined limits might be not compliant with the standard. By using the button *Restore defaults*, the user-defined limits are set back to its default values according to the respective standard.
- 

**Note:** The user-defined *Thermal loading limit* for VDE-AR-N 4100/4105 and 4100 is multiplied with the loading limit in the object. For example, if a line has a limit of 70 % and the limit is set to 90 % in the command, this results in a total limit of 63 % ( $0.7 \cdot 0.9 \cdot 100\%$ ), which is taken into account in the assessment.

---

## 37.2 Connection Request Assessment: D-A-CH-CZ

The selection of *D-A-CH-CZ* as the *Method* in the Connection Request Assessment command carries out a power quality assessment according to Edition 2 ([14], [22]) or Edition 3 ([33], [34], [35], [36]). The Edition can be selected in the command in addition to the method. These guidelines are valid for 50 Hz networks, operating at low-voltage (LV), medium-voltage (MV) or high-voltage (HV) levels.

The assessment considers the following aspects of power quality:

- Voltage changes and flicker
- Voltage unbalance
- Harmonics (*only Edition 2*)
- Harmonics, interharmonics, supraharmonics (*only Edition 3*)
- Commutation notches
- Interharmonic voltages (*only Edition 2*)

**The following assumptions apply to *PowerFactory*'s Connection Request Assessment according to D-A-CH-CZ:**

- Connection request elements are intended for use by the Connection Request Assessment. They may be considered by the Load Flow but for all other calculations, it is recommended to set them out of service.
- Calculations are valid for 50 Hz networks only;
- The PCC is the busbar to which the Connection Request element is connected;
- Each Connection Request element is assessed independently from all other Connection Request elements (with the exception of slow voltage change assessment for generating stations);
- The short-circuit power,  $S_k$ , is calculated using a line/cable temperature of 70 °C for LV networks, and 20 °C (Edition 2) respectively 70 °C (Edition 3) for MV and HV networks.
- For the calculation of flicker, voltage changes are regular and rectangular;
- For the calculation of flicker,  $P_{lt}=P_{st}$  (i.e. 2h observation period);
- Classifications of voltage levels (LV, MV and HV) are strictly according to [14], [22] and [33]. Dialog options pertaining to HV networks are only available for Connection Request elements which are connected to a HV PCC.

### 37.2.1 Basic Options D-A-CH-CZ

#### Method

- **According to D-A-CH-CZ.** Assesses network disturbances according to Edition 2 ([14], [22]) or Edition 3 ([33], [34], [35], [36]).

#### Considered feeder(s)

- One or several feeders can be selected here in order to limit the area of assessment to the selection.
  - Limit violations are only assessed for elements within the selected feeder(s).
  - For the calculation of the *slow voltage change*, only generation units within the selected feeder(s) are considered.

## Calculations

All calculations are carried out according to the selection of *Method* and *Edition* described above.

- **Short-circuit power at the PCC.** The short-circuit power at the point of common coupling (PCC) is calculated by default and according to Section 36.6.

- The short-circuit power,  $Sk$ , is calculated using a line/cable temperature of 70 °C for LV networks, and 20 °C (Edition 2) respectively 70 °C (Edition 3) for MV and HV networks.

---

**Note:** The conductor temperature can be user-defined in the *Connection Request Assessment* command on the *Advanced* page.

---

- **Slow voltage change (generating stations only)** Calculates the voltage change via a balanced load flow.

- The *PowerFactory* elements that are considered as generation (for outaging) are: *ElmSym*, *ElmAsm*, *ElmGenstat*, *ElmPvsys*, *ElmVsc*, *ElmVsmono* and *ElmConreq*. The criterion used to detect this generation is based on their calculated power flow. It should be noted that generation defined as part of an MV load *ElmLodmv*, for example, is not considered. Only generation which is connected to higher voltage levels (HV) is not outaged. Therefore, users may define any LV generation that should not be considered by the assessment as an MV load.

- If a higher voltage level exists in the network, and any step-up transformers with tap-changing functionality are found (i.e. using voltage control on the LV side), their tap changer will be set to continuous for the assessment (and returned to their original setting upon completion of the assessment). If a higher voltage level exists and no such transformers are found, a warning is issued. This is to avoid any voltage changes in the HV network impacting on the assessment at lower voltage levels.

- **Voltage changes and flicker.** Calculates the voltage change,  $d$  and flicker severity  $Pst$ ,  $Plt$  at the point of common coupling, and assesses these according to appropriate limits. For the calculation of flicker, voltage changes are assumed to be regular and rectangular.

- **Voltage change:**

- If no step-up transformer is found in the network, an assessment regarding the sudden voltage change will not be provided (i.e. it will be “N/A”). This is because the voltage drop across the transformer can not be considered.

---

**Note:** In Edition 3 one or several supply transformers can be user-defined on the corresponding page in the *Connection Request* element.

---

- An initial load flow is executed with the network as-is, and using the Load Flow command configured as-is. The tap positions of transformers and shunts are then saved. The outage of **each connection request element in turn** is then considered via a subsequent load flow, however this time configured with Load Flow options *Automatic tap adjustment of transformers* and *Automatic tap adjustment of shunts deselected*. Following the assessment, all transformer and shunt taps are set back to their original positions.

- **Voltage unbalance (only LV and MV).** Calculates the voltage unbalance factor,  $kU$ , and assesses this according to appropriate limits.

- **Harmonics (only Edition 2).** Assesses the harmonic content based on user input in the Connection Request element, and makes an assessment according to appropriate limits. For HV networks, the consideration of resonances is optional.

- **Harmonics, interharmonics, supraresonances (only Edition 3).** Assesses the harmonic content based on user input in the Connection Request element, and makes an assessment according to appropriate limits.

- **Commutation notches (only LV and MV).** Calculates the relative short-circuit voltage of the commutation reactance,  $ukkom$ . It should be noted that no approval status is provided following the Commutation Notches calculation, but instead a recommendation.

- **Interharmonic voltages (only Edition 2).** For LV and MV networks, assesses the effect of user-defined interharmonic voltages in the flicker critical range and on the ripple control frequency. For LV and MV networks, limits are taken from EN 61000-2-2. For HV networks, assesses the user-defined harmonic load and user-defined interharmonic voltages against appropriate limits.

## Resonances

---

**Note:** In *Edition 2*, resonances are only considered in HV networks.

- **Consideration of resonances.** Selection of either no consideration of resonances or approximate consideration of the first parallel resonance point. It should be noted that limits for resonances are not a criteria for the assessment, but are output as information.
- **Grid contains mostly.** Selection of either cables or overhead lines. Used in the selection of the resonance factor,  $k_v$  (as defined in Edition 2 ([14], [22]) or Edition 3 ([33], [34], [35], [36])), which is also dependent upon the harmonic order. It should be noted that the highest resonance factors within the ranges stated in the guideline are chosen.
- **Calculation of emission limits.** Calculates the emission limits according to either the simplified method or the general (detailed) method, as defined in the guideline of Edition 2 ([14], [22]) or Edition 3 ([33], [34], [35], [36]).
- **Operation Scenario.** An Operation Scenario may be optionally specified for the calculation of the actual short-circuit power,  $SkV_{akt}$ . This scenario should represent the typical network operation (switching status, etc), and must yield an actual short-circuit power such that  $SkV \leq SkV_{akt}$ . This implies that the network in its state at the time of execution of the Connection Request command will produce a worst-case short-circuit power,  $SkV$ . The Connection Request command will then appropriately activate the specified scenario and deactivate it following the calculation of  $SkV_{akt}$ . If no scenario is specified, it is assumed that  $SkV_{akt} = SkV$ .

### 37.2.2 Advanced D-A-CH-CZ

- **Flicker strength of voltage changes (Edition 3):** For the calculation of the flicker disturbance factors, two different calculation approaches according to the guideline can be selected:
  - Determination using the reference curve
  - Analytical approach

## 37.3 Connection Request Assessment: BDEW, 4th Supplement

The selection of *BDEW, 4th Supplement* as the *Method* in the Connection Request Assessment command carries out a power quality assessment according to [15] and [23]. This guideline considers the following aspects of power quality for 50Hz networks, operating at the medium-voltage (MV) level:

- Loading of network elements
- Admissible voltage changes
- Sudden voltage changes and flicker
- Harmonics, interharmonics and audio-frequency ripple control
- Commutation notches
- Maximum short-circuit current

**The following assumptions apply to PowerFactory's Connection Request Assessment according to BDEW, 4th Supplement:**

- Connection request elements are intended for use by the Connection Request Assessment. They may be considered by the Load Flow and have a simplified consideration by the Short-Circuit. For all other calculations, it is recommended to set them out of service.
- Calculations are valid for 50Hz networks only;
- The junction point is that specified in the Connection Request element; otherwise is automatically selected to be the busbar to which the Connection Request element is connected;

### 37.3.1 Basic Options BDEW

#### Method

- **According to BDEW, 4th Supplement.** Assesses network disturbances according to [15] and [23].

#### Considered feeder(s)

- One or several feeders can be selected here in order to limit the area of assessment to the selection.
  - Limit violations are only assessed for elements within the selected feeder(s).
  - For the calculation of the *admissible voltage changes*, only generation units within the selected feeder(s) are considered.

#### Calculations

All calculations are carried out according to the selection of *Method* described above.

- **Short-circuit power at junction point.** The short-circuit power at the junction point is calculated by default and according to Section 36.6.
  - The short-circuit power is calculated using a line/cable temperature of 70 °C for LV networks, and 20 °C for MV and HV networks.
- **Loading of network elements.** Subsequent load flow calculations are executed for the following cases: (i) loading without *ElmConreq* elements; (ii) loading with *ElmConreq* elements. Reports overloadings (if any) of network equipment resulting from the VDE Connection Request elements. If no overloading occur, the five highest loadings in each case are displayed in the report.
- **Admissible voltage changes.** Uses the defined generation in any BDEW connection request elements and executes a balanced load flow calculation. The same load flow calculation is then carried out excluding all generation. The voltage change at buses is calculated and violations are reported.
  - The PowerFactory elements that are considered as generation (for outaging) are: *ElmSym*, *ElmAsm*, *ElmGenstat*, *ElmPvsy*, *ElmVsc*, *ElmVsmono* and *ElmConreq*. The criterion used to detect this generation is based on their calculated power flow. It should be noted that generation defined as part of an MV load *ElmLodmv*, for example, is not considered. Only generation which is connected to higher voltage levels (HV) is not outaged. Therefore, users may define any LV generation that should not be considered by the assessment as an MV load.
  - If a higher voltage level exists in the network, and any step-up transformers with tap-changing functionality are found (i.e. using voltage control on the LV side), their tap changer will be set to continuous for the assessment (and returned to their original setting upon completion of the assessment). If a higher voltage level exists and no such transformers are found, a warning is issued. This is to avoid any voltage changes in the HV network impacting on the assessment at lower voltage levels.

- **Sudden voltage changes and flicker.** Calculates the sudden voltage change,  $d$  and flicker severity  $Plt$  at the junction point, and assesses these according to appropriate limits. The sudden voltage change as a result of switching of the entire plant is calculated using a balanced load flow.
  - If no step-up transformer is found in the network, an assessment regarding the sudden voltage change will not be provided (i.e. it will be “N/A”). This is because the voltage drop across the transformer can not be considered.
  - An initial load flow is executed with the network as-is, and using the Load Flow command configured as-is. The tap positions of transformers and shunts are then saved. The outage of **each connection request element in turn** is then considered via a subsequent load flow, however this time configured with Load Flow options *Automatic tap adjustment of transformers* and *Automatic tap adjustment of shunts deselected*. Following the assessment, all transformer and shunt taps are set back to their original positions.
- **Sudden voltage change due to switching of the entire plant.** When assessing the sudden voltage change due to the plant (as opposed to the individual units), a step-up transformer is required. If no step-up transformer is found in the network, an assessment regarding the sudden voltage change will not be provided (i.e. it will be “N/A”). This is because the voltage drop across the transformer can not be considered. An initial load flow is executed with the network as-is, and using the Load Flow command configured as-is. The tap positions of transformers and shunts are then saved. The outage of each connection request element in turn is then considered via a subsequent load flow, however this time configured with Load Flow options *Automatic tap adjustment of transformers* and *Automatic tap adjustment of shunts deselected*. Following the assessment, all transformer and shunt taps are set back to their original positions.
- **Harmonics.** Assesses the harmonic content based on user input in the Connection Request element, and makes an assessment according to appropriate limits.
- **Commutation notches.** Calculates the relative short-circuit voltage of the commutation reactance,  $ukkom$ . It should be noted that no approval status is provided following the Commutation Notches calculation, but instead a recommendation.
- **Maximum short-circuit current.** Uses the defined short-circuit current injection in any BDEW connection request elements and executes a short-circuit calculation according to VDE. Short-circuit current limit violations (obtained from busbar Types and line Types) are reported. The rated short-time thermal current limits of busbars and lines are based on the *Fault Clearing Time (Ith)* that is set in the short-circuit command.

#### **Agreed service voltage, $U_c$**

The supply voltage agreed between the system operator and the connectee. According to the standard it needs to be defined and is used to calculate the limits for the *Sudden voltage change*.

## **37.4 Connection Request Assessment: VDE-AR-N 4100/4105**

The selection of *VDE-AR-N 4100/4105* as the *Method* in the Connection Request Assessment command carries out a power quality assessment according to VDE-AR-N 4100 [30] and VDE-AR-N 4105 [21] [28]. The VDE-AR-N 4100 assesses consumers while the VDE-AR-N 4105 assesses generators. In *PowerFactory* consumers and generators can be considered in the same connection request if an installation includes both types.

The guidelines consider the following aspects of power quality for 50Hz networks, operating at the low-voltage (LV) level:

- Minimum short-circuit power at junction point
- Loading of network components
- Permissible voltage changes

- Rapid voltage changes and flicker
- Voltage unbalance
- Harmonics, interharmonics and audio-frequency ripple control
- Commutation notches
- Maximum short-circuit current

**The following assumptions apply to *PowerFactory*'s Connection Request Assessment according to VDE-AR-N 4100/4105:**

- Calculations are valid for 50Hz networks only;
- The junction point is that specified in the Connection Request element; otherwise is automatically selected to be the busbar to which the Connection Request element is connected;

### 37.4.1 Basic Options VDE-AR-N 4100/4105

#### Method

- **According to VDE-AR-N 4100/4105.** Assesses network disturbances of generation units ([21] in version 2011, [28] in version 2018) and loads ([30] in version 2018).

#### Considered feeder(s)

- One or several feeders can be selected here in order to limit the area of assessment to the selection.
  - Limit violations are only assessed for elements within the selected feeder(s).
  - For the calculation of the *Permissible voltage changes*, only generation units within the selected feeder(s) are considered.

#### Published

This option offers a sub-selection for the selected Method, where the version of the standard to be used can be selected according to the year in which it was issued. The most recent standard is 2018, however 2011 is still available for the verification of documented results. The selection of version 2018 enables the additional calculation of the *minimum short-circuit power at junction point*.

#### Calculations

All calculations are carried out according to the selection of *Method* described above.

- **Short-circuit power at junction point.** The short-circuit power at the junction point is calculated by default and according to Section 36.6.
  - The short-circuit power is calculated using a line/cable temperature of 70 °C for LV networks, and 20 °C for MV and HV networks.

---

**Note:** On the *Advanced* page in the *Connection Request Assessment* command there are two options available to adapt the calculation of the short-circuit power:

- For the calculation the harmonic load flow or short-circuit command can be used.
- The conductor temperature can be user defined.

---

- **Minimum short-circuit power at junction point.** Applicable to the 2018 version of VDE-AR-N 4105 only. The assessment calculates the short-circuit power (see Section 36.6) at the junction point and at the supply transformer busbar and verifies the results against the rated power of pre-selected Type-1 generators in the connection request element according to VDE-AR-N 4105 limits.

- For the calculation of the limits, the following elements are taken into account if selected in the connection request element: synchronous generator (*ElmSym*), static generator (*ElmGenstat*), connection request generation unit (*ElmConreq*) and external grid (*ElmXnet*). The user is required to specify those elements that represent “Type-1” elements according to the standard.
- If no supply transformer is specified in the connection request element, an assessment of N/A may be issued.

---

**Note:** On the *Connection Topology* page in the *Connection Request* element, the supply transformer can be defined or automatically searched.

---

- **Loading of network components.** Subsequent load flow calculations are executed for the following three cases: (i) loading without *ElmConreq* elements; (ii) loading with *ElmConreq* elements (*Loads* table only); and (iii) loading with *ElmConreq* elements (*Generation/Storage Units* table only). This will ensure that the worst case is considered. Overloadings of network equipment resulting from the VDE-AR-N 4100/4105 connection request elements are reported. If no over-loadings occur, the five highest loadings from all cases are displayed in the report.
- **Permissible voltage changes.** Uses the generation defined in any VDE-AR-N 4105 connection request elements and executes a balanced load flow calculation. The same load flow calculation is then carried out excluding all generation. The voltage change at buses is calculated and violations are reported.
  - The assessment of the permissible voltage change requires that generation in the LV network is outaged. The elements that are considered as generation (for outaging) are: *ElmSym*, *ElmAsm*, *ElmGenstat*, *ElmPvsys*, *ElmVsc*, *ElmVsmono* and *ElmConreq*. The criterion used to detect this generation is based on their calculated power flow. Only generation which is connected to higher voltage levels (MV,HV) is not outaged.
  - If a higher voltage level exists in the network, and any step-up transformers with tap-changing functionality are found (i.e. using voltage control on the LV side), their tap changer will be set to continuous for the assessment (and returned to their original setting upon completion of the assessment). If a higher voltage level exists and no such transformers are found, a warning is issued. This is to avoid any voltage changes in the HV network impacting on the assessment at lower voltage levels.
- **Rapid voltage changes.** Calculates the sudden voltage change  $d$  for each generation unit and load at the junction point, and assesses these according to appropriate limits.
- **Rapid voltage changes due to switching off the entire plant.** The rapid voltage change as a result of switching off the entire plant is calculated using a balanced load flow. When assessing the rapid voltage change due to the plant (as opposed to the individual units), a step-up transformer is required.
  - If no step-up transformer is found in the network, an assessment regarding the sudden voltage change will not be provided (i.e. it will be “N/A”). This is because the voltage drop across the transformer can not be considered.
  - An initial load flow is executed with the network as-is, and using the Load Flow command configured as-is. The tap positions of transformers and shunts are then saved. The outage of each connection request element in turn is then considered via a subsequent load flow, however this time configured with Load Flow options *Automatic tap adjustment of transformers* and *Automatic tap adjustment of shunts* deselected. Following the assessment, all transformer and shunt taps are set back to their original positions.
- **Flicker.** Calculates the flicker severity  $P_{lt}$  of the individual generation units and loads as well as the whole plant and assesses the flicker disturbance factors according to appropriate limits.
- **Voltage unbalance.** Calculates the voltage unbalance from user-input data and assesses this according to the VDE-AR-N 4100/4105 limit.

- **Harmonics, interharmonics and audio frequency ripple control.** Assesses the harmonic and interharmonic content based on user input in the Connection Request element, and makes an assessment according to appropriate limits. In version 2018, a simplified calculation is available. By activating the option *Simplified calculation of harmonic current limits* in the command, simplifying assumptions according to the standard are used. For the audio frequency ripple control it verifies if the level is within the permitted range.
  - The non-simplified assessment of all harmonic limits (version 2018) can only be done when the pre-defined k-factors in the connection request element are not zero. When the resonance factor  $k_V$  and/or the share factors  $k_B$ ,  $k_E$  and  $k_S$  are zero, the harmonic limits can't be calculated and a warning is printed into the output window.
  - When the k-factors can't be determined, it is recommended to use the *Simplified calculation of harmonic current limits* (version 2018).
- **Commutation notches.** Calculates the relative short-circuit voltage of the commutation reactance,  $uk_{kom}$ . It should be noted that no approval status is provided following the Commutation Notches calculation, but instead a recommendation.
- **Maximum short-circuit current.** Uses the short-circuit current injection defined in any VDE-AR-N 4100/4105 connection request elements and executes a short-circuit calculation according to IEC 60909 for a balanced 3-phase fault. The command *Calculation* option *Max. short-circuit currents* is used and the *Max. Voltage Tolerance for LV-Systems* is temporarily set to 10%. Short-circuit current limit violations (obtained from busbar and line types) are reported. The rated short-time thermal current limits of busbars and lines base on the *Fault Clearing Time (Ith)* that is set in the short-circuit command.

## 37.5 Connection Request Assessment: VDE-AR-N 4110

The selection of *VDE-AR-N 4110* as the *Method* in the Connection Request Assessment command carries out a power quality assessment according to [29] (version 2018) or [38] (version 2023). This guideline considers the following aspects of power quality for 50Hz networks, operating at the medium-voltage (MV) level:

- Minimum short-circuit power at junction point
- Loading of network components
- Admissible voltage changes
- Rapid voltage changes and flicker
- Voltage unbalance
- Harmonics, interharmonics and audio-frequency ripple control
- Commutation notches
- Maximum short-circuit current

The guideline not only applies to generation plants such as wind turbines, photovoltaic and combustion engines but it also considers storage systems and mixed plant (consumption and generation). It should be noted that only generating plants that fulfil the following criteria are considered for the Connection Request Assessment:

- Plants with a maximum active power (based on the installed power) of  $P_{A,max} \geq 135 \text{ kW}$ .
- Plants existing of individual unit types (such as CHP generation units, wind and hydropower generation units, Stirling generators, fuel cells and asynchronous generators) where at least one of the units has a total effective power of  $P_{E,max} \geq 30 \text{ kW}$ .

If only one or none of these requirements is fulfilled the assessment of the generation plant should be carried out and certified according to the VDE-AR-N 4105 (see Section 37.4) regardless of the voltage level.

**The following assumptions apply to *PowerFactory*'s Connection Request Assessment according to VDE-AR-N 4110:**

- The MV network should be radial with the higher level network represented by an equivalent external grid
- There should be a single supply transformer (user-defined)

---

**Note:** On the *Connection Topology* page in the *Connection Request* element, the supply transformer can be defined or automatically searched.

---

### 37.5.1 Basic Options VDE-AR-N 4110

#### Method

- **According to VDE-AR-N 4110.** Assesses network disturbances according to [38] (version 2023) or [29] (version 2018).

#### Considered feeder(s)

- One or several feeders can be selected here in order to limit the area of assessment to the selection.
  - Limit violations are only assessed for elements within the selected feeder(s).
  - For the calculation of the *admissible voltage changes*, only generation units within the selected feeder(s) are considered.

#### Calculations

All calculations are carried out according to the selection of *Method* described above.

- **Short-circuit power at junction point.** The short-circuit power at the junction point is calculated by default. According to VDE-AR-N 4110, the short-circuit command is used with the following (automatic) configuration in order to calculate *Skss* at the junction point terminal:
  - *Method*: IEC 60909
  - *Fault Type*: 3-Phase Short-Circuit
  - *Calculate*: Min. Short-Circuit Currents
  - *Conductor Temperature*: 20 °C
  - Generation units (*ElmSym*, *ElmAsm*, *ElmPvsys*, *ElmGenstat*), external grids (*ElmXnet*) and loads (*ElmConreq*, *ElmLod*, *ElmLodmv*, *ElmVac*) downstream of the supply transformer are set out of service. The topological search ends when a step-up transformer or an open point is reached.

---

**Note:** The conductor temperature can be user-defined in the *Connection Request Assessment* command on the *Advanced* page.

---

- **Minimum short-circuit power at junction point.** The assessment calculates the short-circuit power at the junction point and at the supply transformer busbar and verifies the results against rated power of pre-selected Type-1 generators in the connection request element according to VDE-AR-N 4110 limits.

- For the calculation of the limits, the following elements are taken into account if selected in the connection request element: synchronous generator (*ElmSym*), static generator (*ElmGenstat*), connection request generation unit (*ElmConreq*) and external grid (*ElmXnet*). The user is required to specify those elements that represent “Type-1” elements according to the standard.
  - If no supply transformer is specified in the connection request element, an assessment of N/A may be issued.
- **Loading of network components.** Subsequent load flow calculations are executed for the following three cases: (i) loading without *ElmConreq* elements; (ii) loading with *ElmConreq* elements (*Loads* table only); and (iii) loading with *ElmConreq* elements (*Generation/Storage Units* table only). This will ensure that the worst case is considered. Overloadings of network equipment resulting from the VDE-AR-N 4110 connection request elements are reported. If no overloadings occur, the five highest loadings from all cases are displayed in the report.
  - **Admissible voltage changes.** Uses the defined generation in any VDE-AR-N 4110 connection request elements and executes a balanced load flow calculation. The same load flow calculation is then carried out excluding all generation. The voltage change at buses is calculated and violations are reported.
    - The assessment of the admissible voltage change requires that generation in the MV network is outaged. The elements that are considered as generation (for outaging) are: *ElmSym*, *ElmAsm*, *ElmGenstat*, *ElmPvsy*s, *ElmVsc*, *ElmVsmono* and *ElmConreq*. The criterion used to detect this generation is based on their calculated power flow. It should be noted that generation defined as part of an MV load *ElmLodmv*, for example, is not considered. Only generation which is connected to higher voltage levels (HV) is not outaged. Therefore, users may define any LV generation that should not be considered by the assessment as an MV load.
    - If a higher voltage level exists in the network, and any step-up transformers with tap-changing functionality are found (i.e. using voltage control on the LV side), their tap changer will be set to continuous for the assessment (and returned to their original setting upon completion of the assessment). If a higher voltage level exists and no such transformers are found, a warning is issued. This is to avoid any voltage changes in the HV network impacting on the assessment at lower voltage levels.
  - **Sudden voltage changes.** Calculates the sudden voltage change  $d$  for each unit at the junction point, and assesses these according to appropriate limits.
  - **Sudden voltage change due to switching of the entire plant.** For the load flow based sudden voltage change (i.e. switching of the whole plant), the worst-case results are obtained considering the following: (i) switching on/off the whole plant considering only the *Loads* table; and (ii) switching on/off the whole plant considering only the *Generation/Storage Units* table. This ensures that the worst case is considered.
    - When assessing the sudden voltage change due to the plant (as opposed to the individual units), a step-up transformer is required.
    - If no step-up transformer is found in the network, an assessment regarding the sudden voltage change will not be provided (i.e. it will be “N/A”). This is because the voltage drop across the transformer can not be considered.
    - An initial load flow is executed with the network as-is, and using the Load Flow command configured as-is. The tap positions of transformers and shunts are then saved. The outage of each connection request element in turn is then considered via a subsequent load flow, however this time configured with Load Flow options *Automatic tap adjustment of transformers* and *Automatic tap adjustment of shunts* deselected. Following the assessment, all transformer and shunt taps are set back to their original positions.
  - **Flicker.** Calculated the flicker severity  $Pst$  of the individual generation units as well as the whole plant and assesses the flicker disturbance factors according to appropriate limits. The calculation of  $Pst$  for loads uses the  $Pst=1$  curve as described in the D-A-CH-CZ guideline.

- **Harmonics, interharmonics and supraresonances.** Assesses the harmonic and interharmonic content based on user input in the connection request element, and makes an assessment according to appropriate limits. By activating the option *Simplified calculation of harmonic current limits* in the command, simplifying assumptions according to the standard are used. For the audio frequency ripple control it verifies if the level is within the permitted range.
  - The non-simplified assessment of all harmonic limits can only be done when the pre-defined k-factors in the connection request element are not zero. When the resonance factor  $k_v$  and/or the share factors  $k_B$ ,  $k_E$  and  $k_S$  are zero, the harmonic limits can't be calculated and a warning is printed into the output window.
  - When the k-factors can't be determined, it is recommended to use the *Simplified calculation of harmonic current limits*.
- **Commutation notches.** Calculates the relative short-circuit voltage of the commutation reactance,  $uk_{kom}$ .
- **Maximum short-circuit current.** Uses the defined short-circuit current injection in any VDE-AR-N 4110 connection request elements and executes a short-circuit calculation according to DIN 60909 for a balanced 3-phase fault. The command *Calculation* option *Max. short-circuit currents* is used. Short-circuit current limit violations (obtained from busbar and line types) are reported. The rated short-time thermal current limits of busbars and lines base on the *Fault Clearing Time* ( $I_{th}$ ) that is set in the short-circuit command.

#### **Agreed service voltage, $U_c$**

The supply voltage agreed between the system operator and the connectee. According to the standard it needs to be defined and is used to calculate the limits for the *Sudden voltage change*.

## **37.6 Connection Request Assessment Report**

Once a calculation is completed, the  icon can be used to generate reports. This will bring up the Report Generation (*ComReport*) command, where reports can be selected from a list of the reports that are applicable to the calculation that has been executed. For those reports that accept input parameters (such as an option to show detailed harmonic results), these can be supplied on the Parameters page of the command dialog. Reports can be generated as separate documents or combined into one. By default, reports are generated as PDFs and presented in *PowerFactory*'s inbuilt PDF viewer, but it is also possible to export reports from *PowerFactory* in various different formats.

For more information about reports and the Report Generation command, see Chapter 19: Reporting and Visualising Results, Section 19.5. It should be noted that reports viewed internally in *PowerFactory* are stored in the desktop of the study case, even after they have been closed, until they are actively deleted by the user.

The reports contain the following information:

- Basic Data of the connection request
- PCC Data
- Assessment status (overall, and per calculation)
- Detailed results per Calculation (depending on user selection of Calculations in Connection Request Assessment command dialog.)

# Chapter 38

## Flickermeter

### 38.1 Introduction

In terms of power quality, the term *flicker* is used to describe the phenomenon of unwanted, rapidly fluctuating light levels due to voltage fluctuations. The IEC 61000-4-15 standard specifies the function and design of apparatus for the measurement of flicker, termed the *Flickermeter*. This Flickermeter comprises five functional blocks which, via the use of multipliers, weighting filters, and smoothing and squaring operations, perform the tasks of simulating the *lamp-eye-brain* chain response, and statistically evaluating the flicker signal [12]. *PowerFactory* provides a Flickermeter command for the calculation of the short-term and long-term flicker according to IEC 61000-4-15.

The following sections explain the calculation of short- and long-term flicker by the Flickermeter command, as well as its configuration and handling.

### 38.2 Flickermeter (IEC 61000-4-15)

#### 38.2.1 Calculation of Short-Term Flicker

The short-term flicker value  $P_{st}$  calculated according to IEC 61000-4-15 is a measure of the severity of the flicker based on an observation period of 10 minutes. It is defined mathematically as follows [12]:

$$P_{st} = \sqrt[2]{(0,0314 \cdot P_{0,1}) + (0,0525 \cdot P_{1s}) + (0,0657 \cdot P_{3s})} \\ + (0,28 \cdot P_{10s}) + (0,08 \cdot P_{50s}) \quad (38.1)$$

where the percentiles  $P_{0,1}, P_1, P_3, P_{10}$  and  $P_{50}$  are the flicker levels exceeded for 0.1; 1; 3; 10; and 50% of the time during the observation period. The subscript  $s$  used in the above formula indicates that smoothed values should be used, which are defined as follows [12]:

$$P_{50s} = \frac{P_{30} + P_{50} + P_{80}}{3} \quad (38.2)$$

$$P_{10s} = \frac{P_6 + P_8 + P_{10} + P_{13} + P_{17}}{5} \quad (38.3)$$

$$P_{3s} = \frac{P_{2,2} + P_3 + P_4}{3} \quad (38.4)$$

$$P_{1s} = \frac{P_{0,7} + P_1 + P_{1,5}}{3} \quad (38.5)$$

### 38.2.2 Calculation of Long-Term Flicker

The calculation of the severity of long-term flicker,  $P_{lt}$ , considers the short-term flicker severity values over a longer period of time and is calculated according to the following equation [12]:

$$P_{lt} = \sqrt[3]{\frac{\sum_{i=1}^N P_{sti}^3}{N}} \quad (38.6)$$

where  $P_{sti}$  ( $i = 1, 2, 3, \dots$ ) are the consecutive  $P_{st}$  values and  $N$  is the number of observation periods. It can be seen from [12] that when  $N = 1$ ,  $P_{lt} = P_{st}$ .

## 38.3 Flickermeter Calculation

### 38.3.1 Flickermeter Command

This command is accessible via the *Flickermeter* button  in the Harmonics toolbar, which is accessible by selecting Harmonics/Power Quality when clicking on the button *Change Toolbox* .

### 38.3.2 Data Source

#### 38.3.2.1 File Input

**Import data from:** specifies the type of data file containing the input data. There are five file types available for selection as described in detail in Chapter 38.3.5.

**Filename:** the name of the input data file.

**Results File:** relevant to *Results File* input files only. The name of the *PowerFactory* results file.

**Configuration File:** relevant to *ComTrade* input files only. The name of the corresponding configuration file.

**Info:** a summary of information read from the file.

**Use System Separators:** relevant to comma-separated value (CSV) input files only. Tick the checkbox to use the same separators for parsing the file as those used by the operating system. When unchecked, separators are user-definable.

**Separator for columns:** in the case of a PowerFactory *Measurement File* as the input file type, this indicates the character used as a separator for the columns in the file. In the case of a *User Defined Text File* as the input file type, the separator may be selected as one of *Tab*, *Space* or *Other* (user-defined).

**Decimal Separator:** indicates the separator used for decimal numbers. This is user-definable for a *User Defined Text File* as the input file type

#### 38.3.2.2 Selection of Data for Calculation

This table allows the selection of input file data to be analysed. The leftmost column (with labels  $y_1, \dots, y_{24'}$ ) provides a naming convention for the output of results, indicating which time-series signals from the input file were analysed.

**Element:** relevant only to a *Results File* input file type. Used to specify the element from the results file for which a variable to analyse will be selected. This variable is then specified in the *Variable* column of the same table

**Variable:** relevant only to a *Results File* input file type. Used to specify the variable for the Flickermeter command to analyse. This variable is associated with the selected *Element* (see above).

**Column Number:** refers to the column/s in the input file which correspond to the time-series signal/s to be analysed.

**Variable Name:** for *ComTrade* files, the variable name is automatically read from the input file and displayed in the *Variable Name* column. No variable name is provided for other file types.

**Calculate Pst:** allows the user to select the signals in the input file for which to calculate the short-term flicker ( $P_{st}$ ). Valid for all input file types with the exception of results files.

### 38.3.3 Signal Settings

#### 38.3.3.1 Signal Settings

**Signal Type:** selection of either EMT or RMS input signal type. In EMT, RMS values are calculated by squaring the signal and this introduces a small ripple which contributes to the flicker. In RMS however, this ripple is not present. To calibrate the Flickermeter, one must use the signals defined in the standard (i.e. modulation, etc.) and the results must lie within the defined tolerance band. In order to achieve this, the minimum sampling frequency required for RMS signals is 800 Hz, because at 400 Hz the computed flicker would otherwise be too low.

**Specify start time:** user-defined start time at which data should be read from file. This is an absolute time value that exists within the input file, from which data will be read. If this value cannot be found in the file, the next time point after the specified start time will be used instead.

**Resample Data:** the input data will be resampled by the user-defined *New Sampling Rate*. If the time step of the data within the input file is not constant, the Flickermeter calculation will automatically resample the data at the average sampling rate taken from the input file.

**New Sampling Rate:** user-defined sampling rate at which data will be resampled if option *Resample Data* has been selected.

---

**Note:** The minimum sampling rate required for instantaneous input data is 400 Hz, and for RMS input data is 800 Hz.

---

#### 38.3.3.2 Calculation Settings

**Observation Period:** the time period over which the flicker will be analysed.

**Calculate Plt:** perform calculation of long-term flicker  $P_{lt}$ . When this option is checked, a results file is written.

**Observation Periods:** the number of successive observation periods (or “time windows”) to analyse.

### 38.3.4 Advanced Options

Input signals for Flickermeter can be either RMS or EMT signals. The algorithm treats both of these inputs the same, with the exception of the weight filter coefficients, scaling factor and the cut-off frequency used. The weight filter coefficients are preset (see Table 38.3.1), however the scaling factor and cut-off frequency are user-definable parameters and are described below.

#### 38.3.4.1 Parameter Definitions

**Cut-off Frequency** Cut-off frequency of Butterworth filter (Hz). When using an RMS input signal, the cut-off frequency is set to 50Hz; when using an EMT input signal, its default value is 35Hz but can be user-defined.

**Filter Offset** The offset (in seconds) for the filters to stabilise. A positive, non-zero offset should always be entered. When using an RMS input signal, the filter offset is set to 5s; when using an EMT input signal its default value is 5s but can be user-defined. A value of 5s is the minimum amount of time required to initialise the filters and to attenuate the initial transient.

**Scaling Factor** Calibration parameter. When using an RMS input signal, the scaling factor is set to 300469,4835 (defined as  $2 / (0.0025 * 0.0025) / 1.065$ ). When using an EMT input signal, its default value is 303317,5 but can be user-defined.

**Set to default** Resets the *Cut-off Frequency*, *Filter Offset* and *Scaling Factor* to default values.

#### 38.3.4.2 Constant Sampling Rate

**Tolerance:** tolerance for determining whether the sampling rate is constant or not. This tolerance is considered on the *Data Source* page in the *Info* frame when displaying the *Constant Sampling Rate* parameter.

#### 38.3.4.3 Report

Results of the Flickermeter calculation are displayed in *PowerFactory*'s output window provided that *Report* has been selected.

---

**Note:** When executing the Flickermeter command within DPL, the command option 'Report' must be disabled.

---

**Command:** displays the command used to output results. The Flickermeter command will write results to a results file provided that option *Calculate Plt* on the *Signal Settings* page has been selected. The results file used can be accessed via the dialog which opens when the *Command* button → is pressed.

Variable	EMT (from IEC 61000-4-15)	RMS
$\kappa$	1,74802	1,74
$\lambda$	$2 \cdot \pi \cdot 4,05981$	$2 \cdot \pi \cdot 4,1$
$\omega_1$	$2 \cdot \pi \cdot 9,15494$	$2 \cdot \pi \cdot 9,15$
$\omega_2$	$2 \cdot \pi \cdot 2,27979$	$2 \cdot \pi \cdot 2,27979$
$\omega_3$	$2 \cdot \pi \cdot 1,22535$	$2 \cdot \pi \cdot 1,22535$
$\omega_4$	$2 \cdot \pi \cdot 21,9$	$2 \cdot \pi \cdot 1000$

Table 38.3.1: Flickermeter Weight Filter Coefficients

Additionally, results of the Flickermeter command can be viewed within the Data Manager as *Flexible Data* of the Flickermeter command itself. The relevant variable names for selection when defining the Flexible Data are  $b : Pst\_y1, \dots, b : Pst\_y24$ , for short-term flicker values; and  $b : Plt\_y1, \dots, b : Plt\_y24$  for long-term flicker values). In this case, viewing the results of a Flickermeter calculation will appear similar to that illustrated in Figure 38.3.1. It should be noted that when multiple *Observation Periods* have been calculated, only the Plt results will be displayed (Pst results are shown as '0'); and for a single Observation Period the Pst results will be displayed. For further information on defining Flexible Data, refer to Chapter 11 (Data Manager and Network Model Manager), Section 11.2.4.

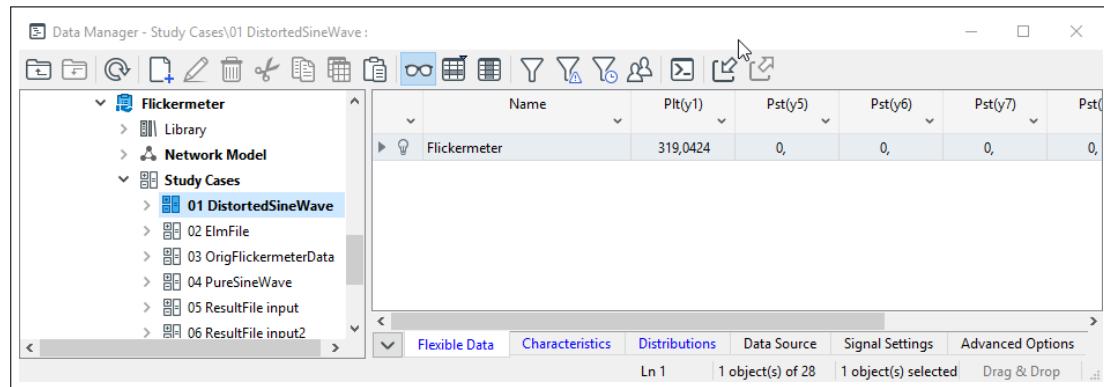


Figure 38.3.1: Using Flexible Data to Access Flickermeter Results

### 38.3.5 Input File Types

The Flickermeter command can handle five different input file types. The configuration of the Flickermeter command for each file type differs slightly, and is therefore described for each case in this section.

**Note:** The minimum sampling rate required for instantaneous input data is 400 Hz, and for RMS input data is 800 Hz.

#### 38.3.5.1 ComTrade

If a *ComTrade* file has been selected as input to the Flickermeter command, the *Configuration File* corresponding to the *ComTrade* data file is automatically displayed, as is the *Sampling Rate* as read from the *ComTrade* configuration file. The *Selection of Data for Calculation* table shows the column number and corresponding variable name as read from the *ComTrade* configuration file and also a user selection for which the short-term flicker value should be calculated (checkbox in the *Calculate Pst* column). In the example shown in Figure 38.3.2, a single variable has been selected for analysis. It can be read from this table that this variable corresponds to column 1 of recorded data in the *ComTrade* input data file. See Section 38.3.2 (Data Source) for information on other Flickermeter command options.

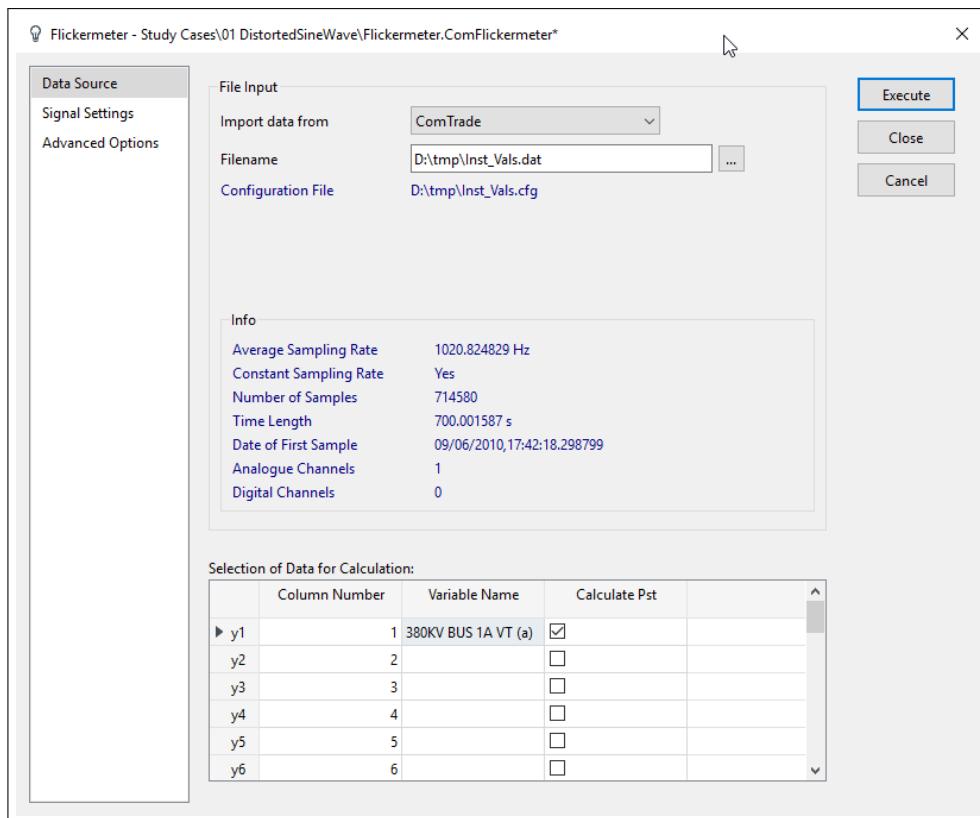


Figure 38.3.2: Configuration of Flickermeter Command for ComTrade Input File

### 38.3.5.2 Comma Separated Values and User Defined Text Files

For a CSV file or user defined text file, the *Selection of Data for Calculation* table shows that variables can be selected for analysis according to their corresponding column number in the input file. See Section 38.3.2 (Data Source) for information on other Flickermeter command options.

### 38.3.5.3 PowerFactory Measurement File

The *PowerFactory* Measurement File is a simple ASCII file containing a column of data for each variable recorded in it. The *PowerFactory* Measurement File can be used to record results from other *PowerFactory* calculations and then used as input to the Flickermeter command. For further information on the use of *PowerFactory* Measurement Files, refer to the technical references of the Measurement File (*ElmFile*). See Section 38.3.2 (Data Source) for information on other Flickermeter command options.

### 38.3.5.4 Results File

If a *Results File* has been selected as input to the Flickermeter command, the command dialog will look similar to that shown in Figure 38.3.3. Using a *PowerFactory* results file as the input file type is practical when the user wants to first record results from, for example, an EMS/RMS simulation in a results file, and then analyse the flicker contribution of one or more variables from this file. In the example in Figure 38.3.3, the specified *Element* in the *Selection of Data for Calculation* table is a terminal element (named “LV Busbar”) recorded in the results file, with its corresponding voltage selected as the *Variable* to analyse. See Section 38.3.2 (Data Source) for information on other Flickermeter command options.

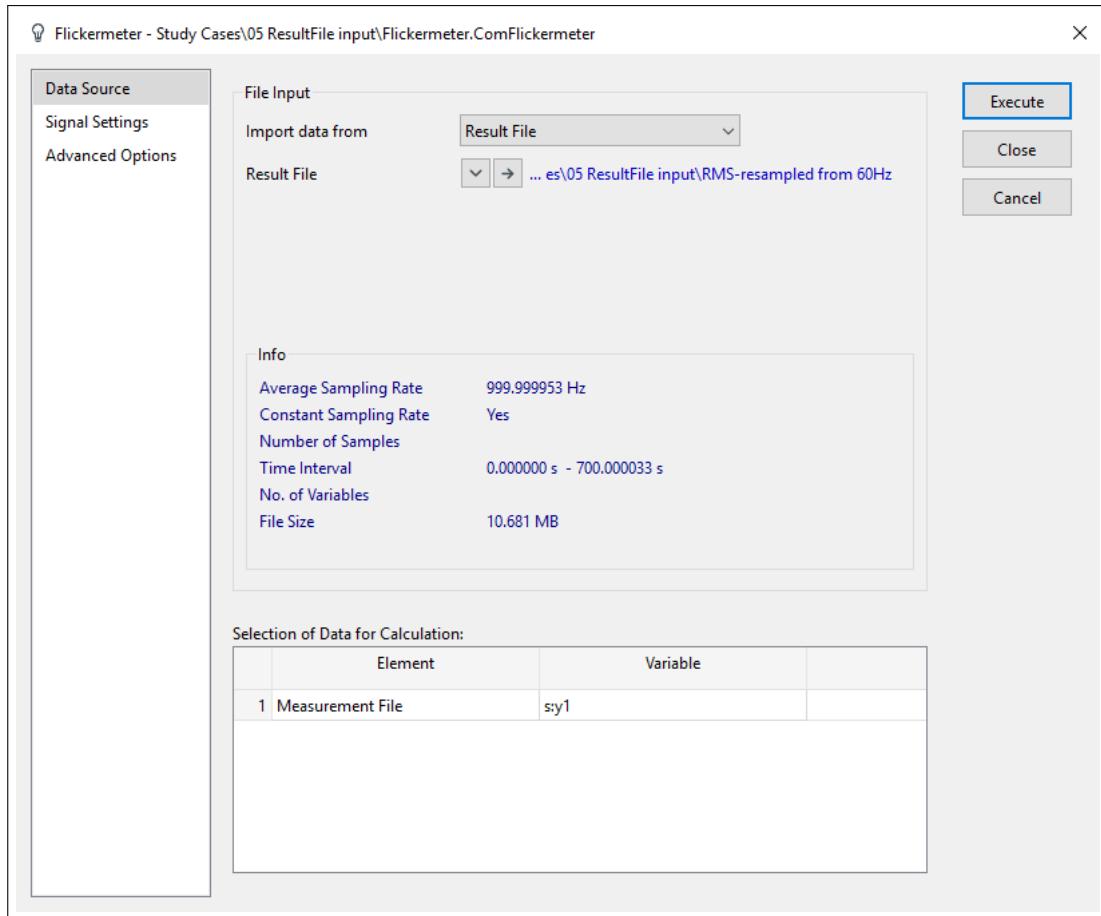


Figure 38.3.3: Configuration of Flickermeter Command for Results File

### 38.3.6 How to use the Flickermeter

The following steps present a small guideline on how to use the Flickermeter command:

1. Define the file type that is supposed to be analysed. The different types are described in Chapter [38.3.5](#).
2. Select the file which contains the measured/simulated data. Different options to configure the command are described in Chapter [38.3.2](#).
3. Depending on the data source selection, the *Selection of Data for Calculation* is done by selecting elements and variables (ElmRes) or by ticking the checkbox “Calculate Pst” for the corresponding column in the source file. This is generally necessary, not only when you want to calculate Pst.
4. Carry out the *Signal Settings* as described in Chapter [38.3.3](#).
  - Select whether the data are instantaneous values (EMT) or RMS values. Make sure that the input data have a minimum sampling rate in order to obtain reliable results. For EMT signals, we recommend a minimum sampling rate of 10 kHz.
5. Define the Observation Period as well as the number of periods that should be analysed as described in Chapter [38.3.3.2](#). Make sure that, depending on whether Pst and/or Plt should be determined, that the input data contains a sufficient time range.

# Chapter 39

## Optimal Power Flow

### 39.1 Introduction

The *Optimal Power Flow* (OPF) module in *PowerFactory* optimises a certain objective function in a network whilst fulfilling equality constraints (the load flow equations) and inequality constraints (e.g. generator reactive power limits). The user can choose between interior point and linear optimisation methods. In the case of linear optimisation, contingency constraints can also be enforced within OPF.

The available methods are:

- AC optimisation (interior point method) (Section 39.2)
- DC optimisation (linear programming (LP)) (Section 39.3)
- Contingency constraint DC optimisation (LP) (Section 39.4)

An OPF calculation in *PowerFactory* can be initiated by one of the following means:

- Via the main menu *Calculation* → *Optimal Power Flow / Unit Commitment* → *Optimal Power Flow...*; or
- By selecting *Optimal Power Flow / Unit Commitment* from the *Change Toolbox* button (▼) and then clicking on the *Calculate Optimal Power Flow* icon 

In both cases, the calculation is started by pressing the **Execute** button in the OPF command dialog.

The OPF functionality is licensed in two separate modules:

The **OPF (Reactive Power Optimisation)** module uses an AC load flow calculation and offers a range of controls and constraints for the optimisation of a selected objective function such as reactive power reserve.

The **OPF (Economic Dispatch)** module is an addition to the OPF (Reactive Power Optimisation) module and cannot be acquired as a stand alone module. The Economic Dispatch module adds the capability to optimise using cost-related controls such as generation active power dispatch. With this module the DC OPF can be used together with the possibility to consider contingencies.

### 39.2 AC Optimisation (Interior Point Method)

If the AC Optimisation method is selected, the OPF performs a non-linear optimisation based on a state-of-the-art interior point algorithm. The following sections explain the selection of objective function to be

optimised, the selection of control variables, and the definition of inequality constraints.

### Mathematical Background

The non-linear optimisation is implemented using an iterative interior-point algorithm based on the Newton-Lagrange method. Recall that the goal of the optimisation is to minimise an objective function  $\mathbf{f}$  subject to the equality constraints imposed by the load flow equations and also to the inequality constraints defined for various power system elements. This is summarised mathematically as follows:

$$\min = f(\vec{x})$$

subject to:

$$g(\vec{x}) = 0 \quad h(\vec{x}) \leq 0$$

where  $\mathbf{g}$  represents the load flow equations and  $\mathbf{h}$  is the set of inequality constraints. Introducing a slack variable for each inequality constraint, this can be reformulated as:

$$g(\vec{x}) = 0 \quad h(\vec{x}) + \vec{s} = 0 \quad \vec{s} \geq 0$$

We then incorporate logarithmic penalties and minimise the function:

$$\min = f(\vec{x}) - \mu \cdot \sum_i \log(s_i)$$

where  $\mu$  is the penalty weighting factor. In order to change the contribution of the penalty function:

$$f_{pen} = \sum_i \log(s_i)$$

to the overall minimisation, the penalty weighting factor  $\mu$  will be decreased from a user-defined initial value ( $\mu_{max}$ ) to a user-defined target value ( $\mu_{min}$ ).

The smaller the minimum penalty weighting factor, the less the applied penalty will be for a solution which is close to the constraint limits. This may result in a solution that is close to the limiting constraint bounds (if necessary). However, a smaller minimum penalty weighting factor will result in a higher number of iterations required.

## 39.2.1 AC Optimisation - Basic Options

### 39.2.1.1 Objective Function

The Optimal Power Flow command dialog, configured for AC optimisation, offers several objective functions. These are:

- Minimisation of losses (total)
- Minimisation of losses (selection)
- Minimisation of costs
- Minimisation of load shedding
- Maximisation of reactive power reserve
  - Based on rated power/control variable range
  - Based on redispatch-/penalty costs
- Minimisation of control variable deviations
  - Based on rated power/control variable range
  - Based on redispatch-/penalty costs

### Minimisation of losses (total)

When this objective function is selected, the goal of the optimisation is to find a power dispatch which minimises the overall **active** power losses.

### Minimisation of losses (selection)

The sum of active power losses of all the elements within the set is minimised. To use this option a set must be defined. The definition of the set is done by selecting the elements, right-clicking on them and selecting *Selections → Set - General → New...*. The newly created set is stored within the study case folder.

### Minimisation of costs

When this objective function is selected, the goal of the optimisation is to supply the system under optimal operating costs. More specifically, the aim is to minimise the cost of power dispatch based on non-linear operating cost functions for each generator and on tariff systems for each external grid.

Generator redispatch costs are not considered in this objective function.

For this purpose, it is necessary to introduce for each generator, a cost function for its power dispatch; and for each external grid, a tariff system.

- **Cost Functions for Generators:** imposing an operating cost function on a generator element is done as follows: on the *Operating Cost* tab of *Optimal Load Flow* page of each synchronous machine (*ElmSym*) element's dialog, it is possible to specify the operating costs of the unit with the aid of the *Operating Costs* table (which relates active power produced (in MW) to the corresponding cost (in \$/h)). This data is then represented graphically beside the *Operating Costs* table, for verification purposes. The number of rows that can be entered into the table is unlimited. To add or delete table rows, right-click on a row number in the table and select the appropriate command (i.e. *Copy, Paste, Select All; Insert Rows, Append Rows, Append n Rows, Delete Rows*, etc.). If there are more than two rows, spline or piecewise linear interpolation can be selected.
- **Tariff Systems for External Grids:** an external grid contributes to the overall cost function by a predefined tariff system. On the *Optimal Load Flow* page of each external grid (*ElmXnet*) element's dialog, the tariffs can be edited via the *Incremental Costs* table. This table relates the cost (in \$/MWh) over a certain range of active power exchange. The input data is represented graphically beneath the *Incremental Costs* table. In addition, the user can enter a monthly no load cost (in \$/month), which can be interpreted as a vertical shift of the cost function.

In contrast to a synchronous machine, where the cost curve is directly expressed in \$/h, the cost curve of an external grid is defined by means of a tariff which holds within certain intervals. Mathematically speaking, the cost curve of a synchronous machine is calculated as the interpolation of predefined cost points, whereas the cost curve of an external grid is a piecewise linear function with predefined slopes in each interval.

Note that this piecewise linear function is not differentiable at the interval limits. Since non-differentiable functions might cause problems within the optimisation routine, *PowerFactory* smooths the cost function slightly over a small range around the non-differentiable points. The width of this range can be defined by the user through the *Smoothing of Cost Function* factor. A value of 0% corresponds to no smoothing of the curve, whereas a value of 100% corresponds to full interpolation. The default value is 5%. It is recommended to leave this value at its default setting.

### Minimisation of load shedding

The goal of this objective function is to minimise the overall cost of load shedding, such that all constraints can be fulfilled. A typical application for this objective function is “Infeasibility Handling”. For the above mentioned objective functions, it may occur that the constraints imposed on the network are such that no feasible solution exists. This is evidenced by a lack of convergence of the optimisation. In

such cases, it is highly likely that not all loads can be supplied due to constraint restrictions. Hence it is recommended in these situations to firstly perform a *Minimisation of Load Shedding*.

In this (and only this) optimisation scenario, all load elements which have the option *Allow load shedding* enabled will act as controls. This option is enabled in the load (*ElmLod*) element's dialog on the *Optimal Load Flow* page in the *Controls* section. All loads without this option enabled will behave as they would in a conventional load flow calculation. In order to minimise the overall load shedding, for each individual load, the user must specify the cost of shedding (in \$ per shed MVA).

For each load that participates as a control in the optimisation, the scaling factor will be optimised. The optimisation is such that the overall cost of load shedding is minimised. Additionally, the user can specify the range over which the load may be scaled (options *Min. load shedding* and *Max. load shedding*).

### Maximisation of reactive power reserve

The objective of this function is to maximise the overall reactive power reserve of all participating generation units. There are three options to select how the maximisation is performed:

- **Min. deviations from Q target value:** if this option is selected the maximisation of reserve is done by keeping the reactive power of the generators as close as possible to a defined value “Q target”. The *Q target* value is set on the *Optimal Power Flow* page of the generator. It can be defined as a percentage of the reactive power limits or a percentage of the rated power.  
This approach is useful to reserve reactive power while monitoring both, critical voltage drop- and rise scenarios.
- **Min. deviations from min. Q:** if this option is selected the target reactive power of the generator is set to its lowest limit. This approach is particularly useful to reserve reactive power for a critical voltage drop scenario.
- **Min. deviations from max. Q:** if this option is selected the target reactive power of the generator is set to its highest limit. This concept is reserved for critical voltage rise scenarios.

*Weighting of deviations:* The weighting of the individual reserve of a generator among all participating generators can either be based on the rated power and the control variable range in relative values or in a more controllable format, based on the redispatch costs for reactive power change of the generators. Penalty cost of other network equipment can be used in both cases.

### Minimisation of control variable deviations

When this objective function is selected, the goal of the optimisation is to minimise changes to the control variables whilst satisfying all the constraints.

The *Weighting of deviations* can be on rated power and the control variable range or based on the redispatch costs in generators and the penalty cost in other network equipment. If the cost based optimisation is used, a redispatch calculation can be carried out whilst considering separate costs for active and reactive power change of generating units.

#### 39.2.1.2 Advanced

On the *Advanced* tab it can be selected, if *Penalty costs for control variable deviations* for transformer taps, shunt steps and HVDC power change shall be considered. The penalty costs for deviating from the load flow operating point can be entered in each respective network element. For HVDCs active and reactive redispatch costs can be entered separately.

## 39.2.2 Controls/Constraints

The global control and constraint settings determine which class of elements will be considered in the optimisation. The general rule is as follows: a parameter will be considered as a control if the corresponding flag is set on the *Optimal Load Flow* page of the element's dialog **and** if, in addition, the corresponding global parameter is set on the *Controls/Constraints* page the Optimal Power Flow command dialog.

### 39.2.2.1 General

#### Controls

The user can specify which parameters might serve as potential degrees of freedom for the OPF algorithm; i.e. which parameters will contribute as controls. The set of potential controls can be grouped into four categories:

1. Generator active power dispatch (*ElmSym*, *ElmXnet*, *ElmGenstat*, *ElmPvsys*, *ElmAsm* (only if DFIG and type is *TypAsmo*), *ElmVsc*, *ElmVsmono*)
2. Generator/SVS reactive power dispatch (*ElmSym*, *ElmSvs*, *ElmXnet*, *ElmGenstat*, *ElmAsm* (Only if DFIG and type is *TypAsmo*), *ElmPvsys*, *ElmVsc*, *ElmVsmono*)
3. Transformer tap positions:
  - 2-Winding Transformer (*ElmTr2*):
    - Tap position, Tap Changer 1 (continuous or discrete)
  - 3-Winding Transformer (*ElmTr3*):
    - Tap position HV-side (continuous or discrete)
    - Tap position MV-side (continuous or discrete)
    - Tap position LV-side (continuous or discrete)
  - 4-Winding Transformer (*ElmTr4*):
    - Tap position HV-side (continuous or discrete)
    - Tap position LV1-side (continuous or discrete)
    - Tap position LV2-side (continuous or discrete)
    - Tap position LV3-side (continuous or discrete)
  - Step-Voltage Regulator (*ElmVoltreg*):
    - S-Tap Position (continuous or discrete)
    - L-Tap Position (continuous or discrete)
  - Transformers can be combined into *External Tap Controllers* for the optimisation. This is beneficial for parallel transformers to guarantee that their tap positions are synchronised after the optimisation.
4. Switchable shunts (*ElmShnt*):
  - Number of steps (continuous or discrete)

It should be noted that the load shedding will only be taken into account for the *Minimisation of load shedding* objective function. In this case, all loads which allow load shedding are automatically used as controls.

If enabled, the above mentioned control parameters serve as variable setpoints during the OPF. However, if a parameter is not enabled as a control parameter, the OPF will treat this parameter according to the load flow settings. This could be a fixed position or a position found due to the option *Automatic tap adjustment of transformers* being selected in the Load Flow Calculation command. In this mode, the transformer tap position could be found in order to control the voltage of a certain node, or to be a slave that is externally controlled by some other transformer tap.

## Constraints

The user can formulate various inequality constraints for certain system parameters, such that the OPF solution lies within these defined limits. The inequality constraints can be defined as “hard constraints” or “soft constraints”. If “hard constraints” are considered in the OPF it may result in no feasible solution being found.

The handling of OPF constraints in *PowerFactory* is very flexible, and various categories of constraints exist. A constraint is considered in the OPF if and only if the individual constraint flag is checked in the element **and** the corresponding global flag is enabled in the OPF dialog.

The optimisation uses further constraints that are automatically imposed as soon as the corresponding parameter is used as a control. Examples of such constraints are tap position limits and the number of steps for switchable shunts.

Network elements and their available constraints are listed below:

- Synchronous Machine (*ElmSym*), Static Generator (*ElmGenstat*), Asynchronous Machine (*ElmAsm*) (only considered if its type is *TypAsmo*, it is set as a *Generator* and its Machine Type is *Double Fed Induction Machine*), PV System (*ElmPvsys*), External Grid (*ElmXnet*), PWM Converter (*ElmVsc*, *ElmVscmono*):
  - Minimum active power
  - Maximum active power
  - Minimum reactive power
  - Maximum reactive power
- Lines (*ElmLne*):
  - Maximum loading
- Transformers (*ElmTr2*, *ElmTr3*, *ElmTr4*, *ElmVoltreg*):
  - Maximum loading
  - Tap position range (if corresponding tap is a designated control parameter)
- Busbars and Terminals (*ElmTerm*):
  - Minimum voltage
  - Maximum voltage
- Bay (*ElmBay*):
  - Maximum loading (AC OPF only)
- Shunts (*ElmShnt*):
  - Controller steps range (if switchable steps are designated control parameters)
- Boundary (*ElmBoundary*):
  - Minimum active boundary flow
  - Maximum active boundary flow
  - Minimum reactive boundary flow
  - Maximum reactive boundary flow
- Static Var System (*ElmSvs*):
  - Minimum reactive power
  - Maximum reactive power

## Active and reactive power limits of generators and external grids

For generators and external grids, the user may impose up to four inequality constraints: namely a minimum and maximum value for active power generation; and a minimum and maximum value for reactive power generation. Active power limits are specified as MW values; reactive power limits may be specified as either absolute values or as per unit values (i.e. referred to the type's rated apparent power). Alternatively, it is possible to directly use the reactive power limits specified in the synchronous machine's type (*TypSym*). Again, the user is free to select any number and combination of the available constraints.

### Branch flow limits (max. loading)

Branch flow limits formulate an upper bound on the loading of any branch (*ElmLne*, *ElmTr2*, *ElmTr3*, *ElmTr4*). The user has to specify a maximum value for the loading on the element's *Optimal Load Flow* page. If specified, this constraint is only taken into consideration if the corresponding flag (*Branch flow limits (max. loading)*) in the OPF dialog is also ticked. Loading limits are supported for lines and 2- 3- and 4-winding transformers.

### Voltage limits of busbars/terminals

The maximum and minimum allowable voltages for each terminal or busbar element (*ElmTerm*) can be specified in the corresponding element's dialog. Therefore, each terminal or busbar may contribute at most two inequality constraints to the OPF. Maximum and minimum voltage limits may be imposed individually; i.e. it is possible to specify an upper limit without specifying a lower limit.

### Boundary flow limits

*PowerFactory* boundary elements (*ElmBoundary*, icon  ) define topological regions in a power system by a user-specified topological cut through the network. Constraints can be defined for the flow of active and reactive power in a network (over a defined boundary or between internal and external regions of a boundary), and this constraint can then be enforced in OPF. For detailed information on defining boundaries, refer to Section 15.4: Boundaries.

#### 39.2.2.2 Soft Constraints

In the individual network elements, it can be specified whether a constraint shall be used as hard or soft constraint. If the soft constraint option is selected, the optimisation may violate the set constraint threshold, which is associated with certain costs/penalties. The penalty used is determined with the *Penalty factor for soft constraints*, the default value of the penalty factor for soft constraints is 1000, multiplied by a *Scaling factor for soft constraint penalty*, which can be specified in the individual network element. This allows the violation of constraints to be weighted differently.

With the setting *Enforce soft...* it is possible to overwrite the setting in the network elements whether a limit shall be interpreted as soft or hard. So for the selected constraints on the *General* tab all the limits of a chosen class are interpreted as soft constraints, if the option *Enforce soft...* is selected.

#### 39.2.2.3 Advanced

The power flow over an HVDC link can be controlled in the PWM Converter similar to a generator in the OPF. HVDC controls are included in the generator active and reactive power control category. If the controls or constraints for the PWM Converter sub category shall be enabled or disabled, this can be done via the *Additional controls* and *Additional constraints* settings for active and reactive power for HVDCs separately.

### 39.2.3 AC Optimisation - Initialisation

The non-linear optimisation requires initialisation to generate an initial starting condition. The *Initialisation* page of the OPF dialog allows the user to select the initialisation method. The user may choose whether the initialisation is performed by using previous OPF results, load flow results or by a flat start.

#### Initialise by flat-start

If this option is selected, the OPF will start with all node voltages set to 1 per unit. Note, that starting from saved results or the load flow results is usually beneficial for the performance and the convergence of the OPF.

#### Start from last calculated results if available

If this option is selected, the OPF checks whether previous OPF results are available and if so, uses the results as starting point for the new calculation. This usually leads to a better and faster convergence of the OPF. If no previous results are available, the fallback is *Start with recalculation of load flow results*.

#### Start with recalculation of load flow results

If this option is selected, the OPF checks whether an “OPF-initialising” load flow result has been calculated prior to the OPF. Here, “OPF-initialising” means that the flag *Use this load flow for initialisation of OPF* was enabled in the load flow command dialog before execution. This flag can be found on the *Advanced* tab of the *Advanced Options* page in the load flow command dialog. The result of this load flow is then used as a starting point for the iterative OPF interior-point algorithm. If no valid OPF-initialising load flow result is found, the OPF will recalculate a new load flow.

#### Start from saved results

If this option is selected, the OPF uses saved results from a previous saved calculation (see Section 39.2.3.1). There are two source options:

- **from memory:** one set of OPF results can be saved to memory. The results are kept in memory as long as the current study case stays active.
- **from a file:** any AC OPF results can be saved to a “.opfs” file. If this option is selected, The file to be used can be defined under *File*. Saved files can also be used after a *PowerFactory* restart.

#### 39.2.3.1 AC Optimisation - Saving results

The *Save Optimal Power Flow Results* command will be shown in both cases and the user can specify whether the current results shall be saved to memory or to a file. Furthermore the user can select, if the object identification shall be done via *object ID* (internal ID that will work even if a parent folder of a network element is renamed) or via *full path name* (full path in the database).

If the option **To a file** is selected the file path must be specified. If the option **To memory** is selected, previously saved results in the memory will be overwritten. The command *Save Optimal Power Flow Results* offers the option to **Adapt the initialisation settings in the Optimal Power Flow command** directly, in order to use the currently saved results. The saved result can be used as a starting point for a new OPF calculation.

### 39.2.4 AC Optimisation - Algorithm

**Solver** The AC OPF offers two solver options the build in *Standard* solver and the open source *Ipopt* solver.

**Penalty Weighting Factor** settings are available if the *Standard* solver is selected.

The penalty weighting factor determines the amount by which the penalty is applied. For example, the smaller the specified penalty weighting factor, the less the penalty will be applied for solutions which are close to constraint limits.

- *Initial value*: initial value of the penalty weighting factor.
- *Target value*: target value of the penalty weighting factor.
- *Reduction factor*: a factor by which the current penalty weighting factor will be divided by between the iterations.
- *Adaption based on complementary slackness*: the penalty weighting factor is adapted during the optimisation. This leads to less iterations and better convergence of the OPF.
- *Start outer loops with initial penalty factor*: the penalty weighting factor is reset to the initial value for each outer loop iteration.

### 39.2.5 AC Optimisation - Iteration Control

*PowerFactory* offers the user flexibility in configuring of the number of iterations and the convergence criteria for OPF. The available options on the *Iteration Control* page of the OPF dialog are described below.

The implementation of the Lagrange-Newton method means that the OPF will internally minimise the resulting Lagrange function:

$$L(\vec{x}, \vec{s}, \vec{\lambda}) = f(\vec{x}) - \mu \cdot \sum_i \log(s_i) + \vec{\lambda}^T \cdot [g(\vec{x}) + h(\vec{x}) + s] \quad (39.1)$$

with the Lagrange multipliers ( $\vec{\lambda}$ ).

The following parameters can be used to alter the stopping criteria for this iterative process. The algorithm stops successfully if the following three criteria are fulfilled:

1. The maximum number of iterations has not yet been reached.
2. All load flow constraint equations  $g(x)=0$  are fulfilled to a predefined degree of exactness (i.e. within an allowable tolerance), which means:
  - all nodal equations are fulfilled
  - all model equations are fulfilled
3. The Lagrange function L converges. This can be achieved if:
  - either the objective function itself converges to a stationary point, or the gradient of the objective function converges to zero.

The following parameters are used to configure these stopping criteria. The alteration of the default values for these parameters is recommended only for advanced users.

#### Maximum number of iterations

- *Interior-point algorithm (inner loop)*: maximum number of iterations for the interior-point algorithm.
- *Control loop (outer loop)*: maximum number of iterations of the outer loop.

### Convergence of objective function

Options relating to the convergence criteria for the Lagrangian function: either the value of the function itself is required to converge to a stationary point, or the gradient of the Lagrangian is required to converge, as described below.

- *Values of objective function become constant*: if this option is selected, the user is asked to enter a value for the *Max. Change of Objective Function*. If the change in value between two consecutive iterations falls below this value, the Lagrangian is considered to have converged. The *Max. Change of Objective Function* is the value(in %), below which the Lagrangian is considered to have converged.
- *Gradient of objective function converges to zero*: if this option is selected, the user is asked to enter a value for the *Max. Value for Gradient of Objective Function*. If the gradient falls below this value, the Lagrangian is considered to have converged. The *Max. Value for Gradient of Objective Function* is the absolute value, below which the Lagrangian is considered to have converged.

For reasons of mathematical exactness, it is strongly recommended to select the latter option, *gradient of objective function converges to zero*. If the underlying Jacobian matrix is numerically unstable, this often results in oscillatory behaviour in the last iterations. Therefore, the latter method provides assurance that the result is in fact a minimum.

### Max. acceptable error for load flow equations

- *Max. acceptable error for nodes*: the maximum allowable error for the nodal equations (in kVA) can be entered for high, medium and low voltage nodes as in the load flow command. The thresholds can be defined in the project settings.
- *Max. acceptable error for model equations*: the maximum allowable error for the model equations (in %).

## 39.2.6 AC Optimisation - Results/Output

### 39.2.6.1 Outputs

Prior to the non-linear optimisation, the OPF informs the user (in the output window) of the total number of constraints and controls that will be considered in the subsequent calculation. This information is detailed such that the imposed constraints and the participating controls are counted for each constraint and control categories separately.

An outer loop is wrapped around the central non-linear optimisation algorithm. This outer loop is required to perform rounding and optimisation of the evaluated tap and shunt positions to discrete values (if desired by the user). The maximum number of outer loops is defined on the *Iteration Control* page of the dialog. However, if no convergence is reached with the defined number of outer loops, the user will be informed via a message in the output window that further outer loop iterations are required.

Three options are available to select the level of detail contained in output messages. These options are available on the *Output* page of the OPF dialog and are described below.

#### Show convergence progress report

If this flag is checked on the *Output* page of the OPF command dialog, the user will get a detailed report on the convergence of the non-linear optimisation. For each step of the iteration, the following figures are displayed in the output window (actual displayed names are shown parenthesised in italics):

- The current iteration if the interior point algorithm (*Iter*);
- The current value of the penalty factor  $\mu$  (*PenFac*).

- The current values of the relaxation factors (*Rlx1*, *Rlx2*) for the primal and dual variables;
- The current error of the constraint nodal equations (in VA) (*Error Nodes*);
- The current error of the constraint model equations (*Error Model Equations*);
- The current error of the inequality constraints (*Error Inequalities*);
- The current value of the gradient of the Lagrangian function (*GradLag*);
- The current value of the objective function *f* to be minimised (*ObjFunc*);

#### Show detailed max. nodal and model equation errors

If this flag is checked, the algorithm outputs per iteration, the components which have the largest error in the equality constraints (i.e. mismatch in the load flow equations).

#### Number of reported marginal cost factors

If this flag is checked, the marginal costs of inequality constraints with respect to the objective function can be issued. Then, the benefit on the objective of releasing a particular constraint of an element is shown in the output window.

This information is issued per iteration and can be used to determine where limiting constraints of the OPF are or which constraint is responsible for a possible infeasibility. The user can also define the number of reported cost factors per iteration.

#### Show full load flow output

In case the OPF is initialised with a recalculation of load flow results, this option gives the possibility to analyse the convergence process within the output window. The level of detail is taken from the *Outputs* page of the Load Flow command.

#### 39.2.6.2 Results

The presentation of OPF results is integrated into the user interface, in that the OPF solution is available via the complete set of variables available for conventional load flow calculations. These can be viewed in the single line diagram or through a data browser.

The complete set of variables from conventional load flow calculations is available. For further information on defining Flexible Data in *PowerFactory*, refer to Chapter 11 (Data Manager and Network Model Manager), Section 11.2.4.

A number of text reports are available and can be selected via the *Optimal Power Flow Report* icon (Report) in the *Optimal Power Flow* toolbar. These reports are generated as text in the output window.

If an AC Optimal Power Flow calculation fails, the final results shown for the models are typically not helpful to detect the reason for the failure. The user can choose which results processed during the iterations of the algorithm should be restored and shown in case of failure, by selecting the desired option from the *Shown results* drop-down menu. The available options are:

- *Last iteration results, if feasible*: the results of last iteration are shown if the OPF converged. It could be the case, that better solution was found in the optimisation process, but with this option the results from the last iteration are shown.
- *Last iteration results*: the results of last iteration are shown regardless whether the optimisation converged or not. A feasible solution could have been found in the optimisation process, but will not be shown if this option is selected.

- *Best found feasible results (Default)*: this option will show the results of a successful optimisation with the best value of the objective function. If there is no feasible solution, no results are shown. A solution is feasible, if the “Errors of nodes” and the “Errors of the model equations” are below the specified thresholds on the *Iteration control* page and the “Errors of inequalities” are resolved (no constraint violation).
- *Results with smallest error*: if the optimisation finds a feasible solution, this solution is displayed as with the default option *Best found feasible results*, but in case of non-convergence, the iteration results with the smallest error are shown. The smallest error is defined as the sum of the “Error nodes”, the “Error model equations” and “Error inequalities”. This can also be the solution of the initial load flow.

## 39.3 DC Optimisation (Linear Programming)

The following describes the configuration of the DC optimisation formulation of OPF in *PowerFactory*.

Internally, from the settings provided, a linear programming (LP) formulation of the problem is derived. The load flow is calculated using the linear DC load flow method. For general information regarding DC load flow, refer to Chapter 25 (Load Flow Analysis).

The result of the linear optimisation tool includes calculated results for control variables, such that all imposed constraints are fulfilled and the objective function is optimised.

Provided that a feasible solution exists, the optimal solution will be available as a calculation result. That is, the algorithm will provide a DC load flow solution where all generator injections and tap positions are set to optimal values.

### 39.3.1 DC Optimisation - Basic Options

The OPF calculation is initialised by a load flow, which is displayed by the *Load Flow* parameter on the *Initialisation* page of the OPF dialog. The user can inspect the load flow settings by clicking on the → button. The load flow command contained in the current study case is set here automatically. Within the load flow command, the *Calculation Method* will be automatically set to *DC Load Flow (linear)* for use by OPF (when *Method* is set to one of the LP variants).

#### 39.3.1.1 Objective Function

The following objective functions are available when executing a DC Optimisation:

##### Feasibility Check

Performs a feasibility check of the network considering the specified controls and constraints (i.e. performs a constrained load flow).

##### Minimisation of Costs

The objective is to minimise generation costs. To perform a cost minimisation calculation for each generator, a cost factor needs to be entered:

*Cost curve \$/MWh per generator element (ElmSym)*: the (linear) algorithm uses a fixed cost-factor [\$/MWh] per generator. This cost factor is the average cost considering the costs at the generator's active power limits. The selection of this objective function provides the option of calculating the Locational Marginal Prices (LMPs). For further information on this option refer to: Shadow Prices and Locational Marginal Prices (LMPs) (See Section 39.3.1.3).

### Minimisation of generator redispatch

Minimises the change in generator redispatch from the generators' initial value. The *Weighting of redispatch powers* can be done *For all generators equally* or *Based on redispatch costs*.

#### 39.3.1.2 Advanced

##### Costs for load shedding

If *Allow Load Shedding* is among the selected Controls (see Section 39.3.2) the cost for load shedding can be enabled or disabled with this setting. If enabled, the individual cost terms for shedding from the loads with enabled load shedding are used and multiplied with the global *Scaling factor for load shedding costs*.

##### Penalty costs for deviations from load flow transformer tap positions

If tap positions are to be optimised, different solutions can yield the same optimal value for the objective function. One can therefore impose a term to the objective function, which forces the solution to be as close as possible to the initial transformer tap positions. The options are:

- *Off*: no penalty costs used.
- *Use global penalty costs*: applying the global cost term to all controlled transformers.
- *Use penalty costs from transformers*: using the individual cost term from controlled. transformers

##### Redispatch costs for deviations from load flow generator dispatches

If the generator dispatch is to be optimised, different solutions can yield the same optimal value for the objective function. One can therefore impose a term to the objective function, which forces the solution to be as close as possible to the initial generators dispatch tap positions. The options are:

- *Off*: no penalty costs used.
- *Use global redispatch costs*: applying the global cost term to all controlled generators.
- *Use penalty costs from generators*: using the individual cost term from controlled generators.

This setting is available for all DC and DC contingency constraint objective functions, except for the *Minimisation of generator redispatch*.

#### 39.3.1.3 Shadow Prices and Locational Marginal Prices (LMPs)

If the option *Calculate locational marginal prices (LMPs)* is selected, the Locational Marginal Price (LMP) is calculated. The Shadow Price is always calculated. The LMP represents the change in the system's total production costs based on a unit change of load at the bus. The calculation of LMP takes into account the network constraints.

The system lambda represents the change in the system's total production costs based on a unit change of any load in the absence of network constraints.

With the *Calculate Locational Marginal Prices (LMPs)* option ticked, the execution of the OPF will (on the fly) calculate the LMP for each busbar. The following quantities (current, voltage and powers) are available for all busbars (i.e. *ElmTerm* elements with *Usage* set to *Busbar*):

- LMP in \$/MWh (Locational marginal price)
- SysLambda in \$/MWh (System lambda)

In addition to the LMPs, the DC Optimisation always computes the shadow prices. These quantities are available per component, which introduces a constraint to the system. The shadow price then represents the change in the objective function if the constraint is released by a unit change.

### 39.3.2 DC Optimisation - Controls/Constraints

#### 39.3.2.1 General

##### Controls

The basic role of each control for the DC Optimisation method is as described for the AC optimisation method in Section [39.2.2.1](#).

The user can select from the following control variables (the names of the associated *PowerFactory* elements are provided in parentheses):

- **Generator active power dispatch** (*ElmSym*)

In generator optimisation, for each selected generator a single control variable is introduced to the system. The total number of generator controls in this case equals the number of selected generators.

- **Transformer tap positions** (*ElmTr2*, *ElmTr3*, *ElmTr4*)

In tap optimisation, for each selected transformer a single control variable is introduced to the system. The total number of tap controls in this case equals the number of selected transformers.

- **Allow load shedding** (*ElmLod*)

A separate control variable is introduced to the system for each selected load. The total number of load controls in this case equals the number of selected loads. This control variable can be selected in conjunction with any objective function.

---

**Note:** At least one type of control variable in the Controls section of the OPF dialog must be selected.

---

##### Constraints

The three constraints are as described for the AC optimisation method in Section [39.2.2.1](#).

The available constraint classes are:

- *Active power limits of generators*: Active power dispatch constraints can be chosen on an individual basis (via a checkbox) per generator. The minimum and maximum constraints for generators are set on the *Optimal Power Flow* page of the generators. It should be noted that generator constraints are **not** implicitly imposed when active power dispatch is selected as a control. Tap position constraints will be implicitly imposed whenever the corresponding tap is a designated control variable.
- *Branch flow limits (max. loading)*: Loading constraints can be chosen on an individual basis (via a checkbox) per line element (*ElmLne*). If loading constraints are included, the maximum loading limits will be calculated with respect to the type of the element, or with respect to a thermal rating object (*IntThrating*). If a thermal rating object is selected, the limits will be calculated with respect to the *Continuous Rating* value.
- *Boundary flow limits*: Boundary flow constraints can be chosen on an individual basis per boundary element (*ElmBoundary*).

For DC optimisation the following constraints are also imposed:

*Transformer tap constraints (implicitly imposed)*

Minimum and maximum tap positions for transformers (*ElmTr2*, *ElmTr3*, *ElmTr4*) are considered. These constraints are implicitly imposed when transformer tap positions are specified as controls in the *Controls* section of the dialog. This means that two constraints are introduced to the LP for the base case tap position calculation.

*Load shedding limits (implicitly imposed)*

The minimum and maximum limit for load shedding for each applicable load (selected as control) are considered if load shedding is used.

### 39.3.2.2 Advanced

#### Calculation of Transformer Tap Positions

- *Discrete controls (using direct method)*: this method calculates discrete tap position values within the LP (known as the “direct method”). This method may provide better accuracy, however will yield fewer solutions.
- *Continuous controls (Using outer loop rounding)*: this method calculates continuous tap position values and then rounds these values to discrete values in the outer loop of the calculation. This method may be faster but the values may not be optimal.

#### Calculation method for transformer tap sensitivities

- *Linearisation of transformer tap changes*: uses linearised load flow equations around the operating point to derive sensitivities to transformer tap positions.
- *Discrete transformer tap assessment*: provides a more accurate assessment in cases where there is a strong dependence of the impedance on the current tap position (e.g. through use of a measurement report). If the degree of dependence between the impedance and the current tap position is not significant, the (faster) linearisation algorithm will be used.

A minimal *Threshold for power flow constraints* is offered. The value represents the minimum change of active power flow in MW per control variable deviation (e.g. change of one tap for transformers and infeed change of 1 MW from a generator) for a control to be considered for a respective constraint.

### 39.3.3 DC Optimisation - Algorithm

The Optimal Power Flow offers different solver choices for solving linear programs. There are build in options available as well as the possibility to connect external commercial solvers. The solver specific configuration parameters can be modified. For details of solver selection and configuration please refer to the according section in the Unit Commitment chapter ([40.3.7.2](#)).

### 39.3.4 DC Optimisation - Iteration Control

Two outer loop settings are available: (i) control of the number of iterations of the algorithm; and (ii) definition of a constraint tolerance. These settings are described below.

#### Outer Loop

Following the solution of the LP problem, it may be the case that loading constraints are not within their boundaries. The reason is that for taps, the algorithm uses tap sensitivities which assume a linear change in MW flow per tap step. Since these tap sensitivities depend on the initial tap position, the result becomes inaccurate if the optimal tap position is far from the initial tap position. This inaccuracy

can be remedied by an additional outer loop. At each iteration, this outer loop starts with the optimised tap positions which were calculated in the previous loop. The following *Outer Loop* settings can be entered on this tab:

- *Max. number of iterations*: maximum number of outer loop iterations until all constraints are fulfilled (within a defined tolerance). It should be noted that when *Max. Number of Iterations* is set to '1', the LP is solved without outer loops.
- *Max. acceptable error for constraints*: maximum relative error (in %) by which a constraint can be violated while still being considered a feasible solution.
- *Check for constraint violations after optimisation*: if this option is selected the violated constraints are written to the output window after the OPF.

### Limitation of branch flow constraints

This option is useful for avoiding long calculation times for large systems. If selected, the LP is solved via an iterative procedure which iterates until no further constraint violations are found (with respect to the *Max. Acceptable Error for Constraints* parameter). It should be noted that the option *Check for Constraint Violations after Optimisation* on the *Advanced Options* page must be selected in order to utilise this iterative procedure. An initial set of branch flow constraints must be selected by the user, as described below.

- *Initial set of branch flow constraints*: the set of branch flow constraints to be considered can either be the *set of N most highly loaded components* or a *user-defined set*. In the case of the *set of N most highly loaded components*, the program finds these automatically either by using a contingency analysis calculation (in the case of a contingency constrained DC OPF) or by using the initial load flow (for the other OPF methods). In the case of a *user-defined set*, the user must define and assign a set of components. A set of components can be defined either via the single line graphic or Data Manager, by multi-selecting the desired components, right-clicking and selecting *Define... → General Set...*. This set can then be selected and assigned via the  button.
- *Max. number of additional constraints per iteration*: after solving the LP with an initial set of constraints, the solution is checked against all loading constraints and overloaded components are added to the LP. The parameter *Max. number of additional constraints per iteration* specifies the maximal number of added components.

## 39.4 Contingency Constrained DC Optimisation (LP Method)

The Contingency Constrained DC Optimisation performs an OPF using DC optimisation (as described in Section 39.3: DC Optimisation (Linear Programming)), subject to various user defined constraints and subject also to the constraints imposed by a set of selected contingencies.

The Contingency Constrained DC Optimisation also considers user-defined post-fault actions. That is, the optimisation can be carried out using contingency cases that include any specified post-fault action. These actions include switch events, generator redispatch events, load shedding events and tap change events.

In order for the OPF to consider post-fault actions, the contingency analysis command that is assigned to the OPF must be set to "Multiple Time Phases". The contingency cases can then be defined to contain post-fault actions. For further information on defining contingency cases with post-fault actions, see Chapter 27: Contingency Analysis.

In addition to the result variables available for DC optimisation, the contingency constrained OPF provides result variables for the individual contingency cases.

## 39.4.1 Contingency Constrained DC Optimisation - Basic Options

### 39.4.1.1 Contingency Analysis

This is a reference to the *Contingency Analysis (ComSimoutage)* command to be used during the contingency constrained OPF. The user can select and set this contingency analysis command via the

▼ button, and view or edit the contingency analysis command settings using the arrow button → . If the user would like the contingency cases to use post-fault actions, the *Method* used by the contingency analysis command must be set to *Multiple Time Phases*. See Chapter 27: Contingency Analysis.

### 39.4.1.2 Objective Function

The selection of objective function for Contingency Constrained DC Optimisation includes the same objective functions as those provided for DC Optimisation (see Section 39.3.1: Basic Options). Two additional objective functions are provided:

#### Min. generator dispatch change (pre-to-postfault)

Minimises the sum of the generator dispatch changes between the base case and each contingency case.

#### Min. transformer tap change (pre-to-postfault)

Minimises the sum of the tap position changes between the base case and each contingency case.

The **Weighting of dispatch changes** for both additional objective functions can be *Based on redispatch costs* or *For all generators equally*. If the base case does not fulfill the selected constraints, the base case will be adapted with the minimal required adjustments.

## 39.4.2 Contingency Constrained DC Optimisation - Controls/Constraints

### 39.4.2.1 Controls

The definition of control variables for the contingency constrained DC optimisation method differs slightly from the DC optimisation method, however the basic fundamental role of each control is as described for the AC optimisation method in Section 39.2.2.

The user can select how to handle the selected controls in the contingency cases, by enabling the *Separate contingency control* for *Generator active power dispatch* and *Transformer tap positions*.

- *Yes*: for each selected control, a control variable is introduced for the base case and for each contingency case. This will result in different setpoint for the base case and the contingency cases.
- *No*: a single control variable is introduced to the system for each selected control. One common setpoint is calculated for the base case and all contingency cases.

The separate control for contingency cases for *Allow load shedding* is always enabled. For the *pre-to-postfault* objective functions the separate contingency controls have to be enabled for the corresponding control class.

### 39.4.2.2 Constraints

This formulation of OPF performs a contingency analysis for a predefined set of contingencies (*ComOutage* objects; i.e. a set of interrupted components per contingency case). The *Max. Loading* (parameter name: *maxload*) for lines and transformers (*ElmLne*, *ElmTr2*, *ElmTr3*; (one constraint per bus)) for each contingency case is considered in the calculation. For each loading constraint, the number of constraints added to the LP will be: 2\*(number of contingencies).

In addition to the constraints provided for DC optimisation (for further information see Section 39.3.2), the contingency constrained DC optimisation method offers additional constraints:

#### Maximum number of tap changes per contingency

If this checkbox is ticked, then for each contingency, no more than the maximum tap position change steps from the base case to the contingency case are allowed over all transformers (i.e. for a given contingency, a constraint is enforced on the sum of all maximum difference of base case to contingency case taps, over all transformers).

#### Transformer tap constraints (implicitly imposed)

Minimum and maximum tap positions for transformers (*ElmTr2*, *ElmTr3*, *ElmTr4*) are considered. These constraints are implicitly imposed when transformer tap positions are specified as controls in the *Controls* section of the OPF command dialog.

## 39.4.3 Contingency Constrained DC Optimisation - Algorithm

As described for DC optimisation. Refer to Section 39.3.3.

## 39.4.4 Contingency Constrained DC Optimisation - Iteration Control

As described for DC optimisation. Refer to Section 39.3.4.

## 39.4.5 Contingency Constrained DC Optimisation - Results/Output

For contingency constrained DC OPF, results can be optionally recorded for those branches which exceed a selected limit value. This can be done for both the non-optimised results and the optimised results. For each recording of results (i.e. with optimised or non-optimised values) a separate results file must be chosen.

### Contingency Analysis results

Allows the selection of results files for the contingency analysis results with and/or without optimised controls.

- *Results (before optimisation)*: the results file in which to store the non-optimised results.
- *Results (after optimisation)*: the results file in which to store the calculated (optimised) results.

### Limits for Recording

The limits displayed here are set in the selected *Contingency Analysis* command on the *Basic Options* page of the contingency analysis command dialog. They define the limits outside of which results

will be written to the results file(s). See Chapter 27: Contingency Analysis, Section 27.4.1 for further information.

## Reports

Following a contingency constrained DC OPF calculation, a number of text reports are available and can be selected via the *Optimal Power Flow Report* icon ( in the *Optimal Power Flow* toolbar. These reports are generated as text in the output window.

The following reports are offered:

- *Optimal Solution*: prints a detailed report to the output window, showing all optimal settings for generators, transformers and loads, component-wise, for all contingencies. An additional flag (*Report only Contingency with max. Deviations*) can be checked to show only the settings for the contingency where the maximum deviation occurs.
- *Optimal Solution (per Contingency)*: prints a detailed report to the output window, showing all optimal settings, on a per-contingency basis.
- *Maximum Loadings*: prints a detailed report to the output window showing the maximum loadings of components against the relevant contingency. The user may define the loading limit for which to report violations, and may select whether to report only the highest loadings for branch components. Moreover, this report facilitates the display of results before and after the optimisation.
- *Loading Violations*: prints a report to the output window showing components with loading violations, against the relevant contingency. The user may define the loading limit for which to report violations, and may select whether to report only the highest loadings for branch components. Additionally, the reporting of violations in contingency cases may be suppressed if violations already exist in the base case.
- *Violations per Case*: prints a report to the output window showing components with loading violations, on a per-contingency case basis. The user may define the loading limit for which to report violations, and may select whether to report only the highest loadings for branch components. Additionally, the reporting of violations in contingency cases may be suppressed if violations already exist in the base case.

## 39.5 Troubleshooting Optimal Power Flow Problems

In general, if a solution can be found, i.e. the optimisation is mathematically solvable, *PowerFactory* will find a solution. In some cases the number of controls and constraints, the bandwidth for the target values and the cost definition of the generators, might lead to non-convergence.

It should be noted that executing an Optimal Power Flow calculation on a large network generally requires careful adjustment of the controlled elements.

This section explains a typical approach to achieve convergence of an (AC) Optimal Power Flow calculation.

### 39.5.1 Verification of Load Flow Options and Results

Although the solution of the load flow calculation is only used as starting point of the optimisation if the corresponding option is selected from the *Initialisation* page of the command, it should be noted that it is always recommended to have a working load flow before performing an optimisation (see Section 25.6: Troubleshooting Load Flow Calculation).

Since the load flow solution is important to the optimal power flow calculation, the first step when troubleshooting a non-convergent optimal load flow is to check the load flow results and options.

Some of the options for active power control and balancing are not supported by the OPF command. The optimal power flow is solved with the following load flow options:

- Active power control set to *As Dispatched*
- Balancing set to *by reference machine*

Therefore, if these are not the options used by the load flow and the OPF is not set to flat start, the *Update Database* command can be used to save the dispatch of the machines to a new operation scenario. It is recommended to compare the results of the load flow with the original active power control options against the results of the load flow after updating the database and the active power control set *As Dispatched* and *by reference machine*.

### 39.5.2 Verifications of OPF Constraints

The introduction of constraints into the optimisation function has a big impact on the convergence. For example, if no constraints are selected, the OPF might find a mathematical solution that is not physically feasible (e.g. infinite active power on a generator); on the other hand having too many constraints selected complicates the convergence process.

In this sense, the regulation of active and reactive power options during the load flow, i.e. power limits and tap changers, is also important for OPF. It is less likely that an OPF will converge if the constraints to be considered during the OPF are violated during the load flow calculation, regardless of the initialisation option selected; being too close to the limit might also make convergence difficult.

A good approach to verify this is to execute a load flow calculation and use the flexible data page of the elements to check if the limits are violated, or nearly violated.

It is also important is to check whether the limits are “reasonable”. Sometimes limits are entered via scripts or an import, and experience is that it is easy for the comma-separator of some limit to be in the wrong place. Note that very broad limits might disturb the OPF since the problem might become unbounded. Reasonable restrictions generally help. For the external grid, the default Q-limits are  $\pm 9999$ , which is not helpful for OPF convergence.

Once this is done, it can be decided which elements should be considered in the OPF, by selecting the corresponding check box of each element.

For the voltage limits, the OPF voltage constraint should not be set on terminals with *internal* or *junction node* usage. Terminals with this usage belonging to a substation automatically take the value of the substation.

It is generally beneficial to consider the respective limits for a selected control, for example the active power limits of a controlled generator, to avoid unreasonable results for this control. On the other hand it is possible in the OPF to select limits of non-controlled unit as constraint, for example the reactive power limits of a generator without reactive power control enabled. This might be a valid approach for some special cases, but generally a negative impact can be expected for the convergence behaviour.

### 39.5.3 Verification of the OPF Controls

The elements selected as controls should also be verified when troubleshooting an OPF. It is important to note that if the elements selected for the control are also participating in station controllers, they won't be able to assist in maintaining a corresponding voltage setpoint during the optimisation.

### 39.5.4 Step-by-Step Approach

In a first step, the OPF should feature simple control variables which are not being used for control by the Load Flow. Below is a step-by-step procedure for an OPF with *Maximisation of Reactive Power Reserve* as objective function; the procedure is similar for other objectives.

1. Select all non-controlling units without capability curves (PQ-generators, not in station-controller, not PV) from the set of potential control machines and mark the *Reactive Power-control* flag. Do not select any limits yet!
2. Configure the OPF command as follows:
  - Select *Reactive Power* controls.
  - Initialise with *Load Flow*.
  - Penalty weighting factor should start at 50 and terminate at 1.0e-05. In this way, we initially give more weight to satisfying the constraints (at penalty 50) and finally give very little weight to this, helping to find the optimal solution.
  - Gradient value for convergence should be set to 1.0e-05 (helps to find the optimal solution).
3. Calculate the OPF and check the deviations from the Q-target values. If it does not converge, add generators one-by-one to find the problematic control.
4. Configure the OPF command to include Reactive Power-constraints. Activate the Q-limits of the currently participating units and run OPF again. If it does not converge, add constraint-pairs (min/max) one-by-one to find the problematic constraints.
5. Include all non-controlling units with capability curves in the tab *Controls and Constraints* on the OPF page and optimise again. Generally, capability curves can cause difficulties in convergence when they are voltage dependent or not differentiable. If it does not converge, add controls and constraint-pairs (min/max) one-by-one to find the problematic ones.

In the next step, we can add generating units that are contained in a station controller or PV-generators to the OPF:

1. Configure the OPF command to include voltage constraints.
2. If Q-setpoints are controlled by a station controller, the corresponding generation units can be added to the OPF directly (controls / constraints). If the OPF shows convergence problems, a boundary can be defined at the Q-setpoint cubicle of the station controller. Then, add Q-limits (only) around that setpoint to the OPF in the boundary and activate *Boundary Flow limits* in the OPF command.
3. For voltage controlling units: Find the controlled terminal of such a generation unit. Check the calculated voltage in the Load Flow and define the upper and lower limit around this value with some bandwidth. Activate the voltage constraints in the terminal.
4. Add the generation units (control/Q-constraints) to the OPF and run it. Do this one-by-one to recognise problematic control variables and limits. Sometimes one may hit an essential controller for the Load Flow.

If tap positions are added to the OPF, note the effect on station controllers: transformers which are selected to participate in the optimisation will no longer be used by the station controllers, which can lead to voltage set points not being met.

When further constraints (voltage or reactive power) are added, one must keep in mind that the selected controls actually need to have an impact on these constraints. This can again be evaluated step-by-step.

## Chapter 40

# Unit Commitment and Dispatch Optimisation

### 40.1 Introduction

The **Unit Commitment and Dispatch Optimisation**  allows the user to complement the electrical calculations with market simulations, without any need for an external tool. It solves the unit commitment linear-programming problem over a predefined period of time, while optimising the operating point of the dispatched generators, for instance to minimise the overall operating costs. The module combines functionalities of Quasi Dynamic Simulation with the Optimal Power Flow and the Contingency Analysis. The Unit Commitment and Dispatch Optimisation functionalities are placed in the toolbar “Optimal Power Flow / Unit Commitment”.

The Unit Commitment and Dispatch Optimisation can be executed based on a balanced AC load flow or a linear DC load flow. A time range and step size for the simulation are defined, and the user can specify whether contingencies should be considered for the optimisation.

If contingencies are defined, curative control actions can be taken into consideration.

The tool offers a wide range of configuration possibility and allows selective configurations to the objective functions, the controls and constraints to match specific requirements.

The Unit Commitment and Dispatch Optimisation is referred to as Unit Commitment in a short form in several places in *PowerFactory* and in the User Manual.

### 40.2 Application Cases for the Unit Commitment

The Unit Commitment and Dispatch optimisation module can be used in various setups, ranging from a basic market simulation to a pure redispatch calculation. This can be achieved by selectively enabling the constraints and controls in the network and differentiating in an AC or DC load flow based calculation. In general the controls and constraints can be selectively enabled for single units or globally in the Unit Commitment command. Thus, it is possible to optimise and investigate the whole system, only certain network parts or even individual units.

### 40.2.1 Full Unit Commitment

The main application is to execute a full Unit Commitment calculation for a network. Unit Commitment optimises various control variables in an electrical network, such as the power dispatch of generators and tap positions of switchable equipment, in order to achieve a common target. This can be done for a single point in time or for a time range. The dependencies between the time steps can be considered via minimum up and down times and ramp limits. Unit Commitment considers the network limitations via definable constraints for equipment loading, power flow limitations and voltage ranges. The goal is to match demand and generation at the most economical operating point by modifying the generator dispatch or by load shedding. Each control variable has assigned costs as a basis for the optimisation. In *PowerFactory*, the Unit Commitment function can directly provide a solution, without the need of a later redispatch, since all the network constraints can be directly taken into account for the base case and selected contingency cases in each time step. Usually, all costs in the system are minimised whilst taking into account all network constraints directly in one execution of the command. If the Unit Commitment is to be independent from the previous dispatch, the additional redispatch costs should not be considered.

### 40.2.2 Market Simulation

The Unit Commitment and Dispatch Optimisation module can also be used to execute a market simulation without considering any (or only selected) network constraints. Thus, the resulting dispatch represents the solution of a free, unconstrained market in which the cheapest units generate the required power regardless of the feed-in point. A market simulation may be needed to determine the power plant schedule of the unregulated market and derive the required redispatch in a second iteration.

The constraints of the generating units such as ramps and minimum downtimes can optionally be taken into account, so as to have a feasible solution for the individual units. Also cross border flows can be considered via boundary constraints. Usually the DC load flow based Unit Commitment is used when executing a market simulation and the additional redispatch costs are not taken into account.

### 40.2.3 Redispatch Calculation

A pure redispatch calculation can be carried out with the Unit Commitment module by first executing a market simulation without considering network constraints and then running a Unit Commitment while regarding network constraints such as line loadings, and considering contingencies. The results from the market simulation can also be imported into *PowerFactory* via characteristics, if the market simulation is executed in an external program which might not be able to consider network constraints for n-x cases. It might be a good option to disregard the operational costs in the objective function for a redispatch calculation and work with positive and negative additional redispatch costs. Since a redispatch calculation is strongly focussed on the network limitations, the AC load flow based Unit Commitment should be used to consider the influence of the reactive power on the critical loaded elements as well.

Individual curative redispatch actions for each contingency can be computed in addition to the base case solution if the curative redispatch option is enabled.

## 40.3 Unit Commitment Command

### 40.3.1 Basic Options

#### 40.3.1.1 Load flow

The Unit Commitment and Dispatch Optimisation can be executed based on an **AC balanced load flow** calculation or a linearised **DC load flow**. If the AC load flow is selected, the Unit Commitment can optimise the active and reactive power dispatch and can additionally include voltage and reactive power constraints. For the DC-based simulation only the active power dispatch is optimised. The load flow method is automatically set in the load flow command. Other load flow settings can be changed via the pointer to the load flow command. Keep in mind that the optimisation is always linear. Therefore the control parameters for an AC Load Flow based Unit Commitment are also set bases on the linearised optimisation and can lead to deviations in set constraint values or even non-convergence.

#### 40.3.1.2 Consider Contingencies

When ticked, one common dispatch is determined respecting base- and outage situations simultaneously. Only network constraints such as loading-, boundary- and voltage constraints are formulated with the base case control variables for the contingencies. The constraint group can be individually selected to feature contingency constraints. It is also possible to determine an optimal dispatch with this tool incorporating all base case constraints only. In a second step, the algorithm can then write the corresponding results for all contingencies to result files. The contingency command can be accessed and modified via the pointer to the **Contingency Analysis**. Please note that only a very restricted part of the contingency Analysis features is supported.

- Only Single Time Phase is supported
- Contingency screening and Remedial Actions are not supported

Contingency filters (see Section 40.3.4.4) can be defined to restrict the consideration of the individual constraints to the most relevant ones.

If the option **Use linearised calculation** is selected, the contingency analysis will execute feasible contingencies with a fast linearisation method (for AC or DC).

#### 40.3.1.3 Time Period

The calculation time period can be defined via default time periods like a day a month or a year or as a user-defined range. The calculation is then executed for certain time points over the defined time period. These calculation points can either be equidistant with a specified step size or, if “User-defined study times” is selected, individually defined.

## 40.3.2 Objective Function

### 40.3.2.1 General

The objective function of the minimisation can be defined by the user, who chooses either “**Minimisation of total costs**” or “**User-defined**”. “Minimisation of total costs” defines the complete objective function with all function components, which then minimises the overall network operating costs. This means all costs selected in the models will be used. If the “User-defined” objective function is chosen

the function components can be selected individually. If a function component is disabled, it disregards the settings/costs in the corresponding network elements. The available function components are:

- **Minimise generator active power operating costs:** The generator operating costs include the fuel costs, emission costs and other fixed and variable costs. The operating costs can be defined via a table directly in the generator or in a cost curve model for generators (*ElmSym*), external grids (*ElmXnet*), static generators (*ElmGenstat*), PV-systems (*ElmPvsys*) or doubly-fed induction machines (certain *ElmAsm* models), whenever they are not selected to be variable renewable energy sources (VREs).
- **Minimise reactive power operating costs:** The generator reactive power operating costs can be defined via a table in the generators (*ElmSym*), external grids (*ElmXnet*), static generators (*ElmGenstat*), PV-systems (*ElmPvsys*) or doubly-fed induction machines (certain *ElmAsm* models), independent of their generator usage.
- **Minimise additional generator active power redispatch costs:** The objective function participation tries to keep generator active power values as close as possible to the values prior to the optimisation and can be defined for the same elements as above.
- **Minimise additional reactive power redispatch costs:** The objective function component tries to keep the generator reactive power respectively the voltage setpoints as close as possible to the defined target values or the values prior to the optimisation. The additional reactive power redispatch costs can be defined based on the reactive power deviation or the voltage setpoint deviation for the same element classes as mentioned under *Minimise reactive power operating cost*. The deviation can be minimised either from the reference value of the current time (default, also for the option *Minimisation of total costs*) or from the optimised value of the previous time. The corresponding setting can be defined individually for thermal generating units, VREs, generators coupled with storage model and HVDCs (reactive power) on the *Time Coupling* tab (see Section 40.3.2.2)
- **Minimise generator start-up/shut-down costs:** Start-up/shut-down costs can be defined for the same elements as operating costs. The costs can be entered via a constant cost value, a simple model with cold and warm-start-up costs or a step-function (via a table) depending on the off-line time.
- **Minimise curtailment of variable renewable energy sources (VREs):** Curtailable elements where the flag Variable Renewable energy source is ticked on the Unit commitment page can reduce their injection. This part of the objective minimises the curtailment of these VREs. All generator elements that feature production costs above can be selected to be VREs, but typically they would be of class *ElmGenstat* or *ElmPvsys*.
- **Minimise load shedding:** Load shedding is a last emergency step for network operators. This is possible for loads where the flag “Allow load shedding” is ticked on the Unit Commitment page. This part of the objective minimises the shedding of loads based on the Load Flow values.
- **Minimise penalty costs of transformers, shunts and HVDCs (active power):** The objective function component tries to keep taps and the active power of HVDC-controls as close as possible to the values prior to the optimisation of the current time step, e.g. the load flow tap/control. Costs for deviations are defined within the individual elements, which are *ElmTr2*, *ElmTr3*, *ElmTr4*, *ElmVolreg*, *ElmShnt*, *ElmVsc*, *ElmVsmono*. The deviation can be minimised either from the load flow value of the current time (default, also for the option *Minimisation of total costs*) or from the optimised value of the previous time. The corresponding setting can be defined individually for transformers, shunts and HVDCs (active power) on the *Time Coupling* tab (see Section 40.3.2.2).

#### 40.3.2.2 Time Coupling

If the objective function *Minimisation of total costs* is selected, no time coupling options are available. In this case, the deviation from the load flow / reference value is minimised. In order to define time couplings for additional reactive power redispatch costs or penalty costs for transformers, shunts and

HVDCs, the objective function must be set to *User-defined* and one of the following objective function components must be selected for minimisation:

- *Additional reactive power redispatch costs*
- *Penalty costs of transformers, shunts and HVDCs (active power)*

For *Thermal generation units*, *VREs*, generators *Coupled with storage model*, *HVDCs*, *Transformers* and *Shunts*, the user has two options to determine from which starting point the deviation is to be minimised:

- **Reference / Load flow value of current time:** Means that the positions are kept as close as possible to the values prior to the optimisation, i.e. the load flow tap/controls (default if *Minimisation of total costs* is selected)
- **Optimised value of previous time:** Means that the positions are kept as close as possible to the optimised values of the previous time step. If this option is selected, the settings for *Control variable values outside of calculation time period* become relevant (see Section 40.3.3.3).

### 40.3.3 Controls

#### 40.3.3.1 General

##### Controls

The controls offered depend upon the selected load flow method and the controls linked to reactive power are not available if DC load flow initialisation is used. The supported controls are:

- **Active power dispatch:** Synchronous generators (ElmSym), external grids (ElmXnet), static generators (ElmGenstat), PV-systems (ElmPvsy) or doubly-fed induction machines (certain ElmAsm models) may contribute such a control variable. HVDCs (ElmVsc, ElmVscmono) can also be selected as control variables, if they are AC active power controllers in the load flow as well. The slack or reference machine should be included as an active power control.
- **Transformer tap positions:** Quad-Boosters in particular can be very effective in reducing loading violations. Elements that may participate here are ElmTr2, ElmTr3, ElmTr4, ElmVoltreg. The general representation as continuous or discrete controls is configured on the Advanced page. Please note that this only applies to controls that are selected to be discrete controls in the element itself. Tap controllers for parallel transformers can be used.
- **Load shedding:** Loads can be shed to eliminate overloads, when this option is selected. The priority is based on the cost in the element.
- **Reactive power dispatch (only AC):** Synchronous generators (ElmSym), external grids (ElmXnet), static generators (ElmGenstat), PV-systems (ElmPvsy) or doubly-fed induction machines (certain ElmAsm models), HVDCs (ElmVsc, ElmVscmono) may contribute such a control variable. The relation between reactive power and constraints such as bus voltages and loadings is usually highly non-linear, especially for larger networks. But the optimisation itself is linear and uses the linear sensitivities from the operating point before the optimisation. Thus, deviations and mismatches can regularly occur when optimising the reactive power dispatch.
- **Switchable shunts (only AC):** This control may assist when it comes to voltage constraints / stability.

**Control variable limits:** It can be selected whether the limits (which are defined in the network elements) should be considered or not via list boxes on the right side of the control variables.

It is recommended to include the control variable limits to obtain realistic results. The available limits are:

- Active power limits

- Transformer tap positions limits
- Load shedding limits
- Reactive power limits (only AC)
- Shunt tap positions limits (only AC)

**Additional controls** are available if the active or reactive dispatch controls are enabled, in order to specify whether certain classes of generators/converters shall participate as controls in the Unit Commitment.

- **Active/reactive power controls for virtual power plants:** The participation of virtual power plants can be enabled or disabled for the unit commitment with this setting.
- **HVDC active/reactive power controls** are important for the redirection of power flows in order to reduce overloads. This setting can be changed when active or reactive power controls are selected.
- **Allow curtailment of variable renewable energy sources:** Generators marked as VRE can be shed to eliminate overloads, when this option is selected. It is active and can be changed when active or reactive power controls are selected. The priority is based on the cost in the element.

#### 40.3.3.2 Curative Controls

Curative controls actions are only applicable if contingencies are considered in the Unit Commitment. The curative capabilities of each control has to be specified individually in the network element. Under the global setting **Allow curative control variables** the curative control actions can be enabled or disabled for a corresponding control class (*Active power dispatch, Transformer tap positions, ...*) by setting the **Allowed control usage**:

- *Preventive only*: Only the preventive controls of the corresponding class are used. The curative controls are disregarded.
- *Curative only*: Only the curative controls of the corresponding class are used. The preventive controls are disregarded.
- *Preventive and Curative*: Each control of the corresponding class is used as specified in the network element.

The possible effect of curative actions is considered for the base case in the optimisation and thus a more relaxed base case solution can be achieved. The costs of curative actions are neglected for the base case solution, since the probability of an actual activation is very low. The cost of curative actions are used to determine which actions are to be applied for each contingency and each point in time.

The usage of curative actions strongly increases the size of the optimisation problem, thus longer solving and execution times have to be expected compared to only using preventive control actions.

#### 40.3.3.3 Advanced

**Power balance:** The option “Enforce control variable power balance” will force the Unit Commitment to do a redispatch if required to keep the same power balance (sum of output of generators selected as control) before and after the optimisation. If this option is not selected and the slack not included in the Unit Commitment as a control, the optimisation would use this degree of freedom and reduce the infeed of power with the controlled machines.

**Modelling of Discrete Controls:** Transformer tap positions and switchable shunts can either be modelled as discrete variables in the optimisation, or as continuous variables which are rounded at the end of the optimisation. This approach will be applied whenever there is an element in the system where the control type of the tap/shunt is discrete.

**Control variable values outside calculation time period:** The historical and future dispatch situations are important in a Unit Commitment calculation if time-dependent constraints are to be considered. If the option “Not considered” is selected before or after the time period, it means that the Unit Commitment can move the controls freely to any setpoint without the consideration of ramps or start up/shut down costs. For example: If a generator is offline in the load flow the Unit Commitment can start it in the first hour without any start up costs to any setpoint point without the limitations by ramps, since no history is considered. If the variable values outside the calculation period should be considered, the options “Non-optimised values of first calculation time” and “Non-optimised values of last calculation time” can be selected. In this case, the load flow value from the first/last point in time is taken and a generator that is for example offline at the start of the calculation period will have start up costs in the first time step if it is started. Also the ramp limitations will apply for this first calculation step. If any further history of control variables shall be considered, characteristics have to be used (e.g. on the “Must run”-flag or directly on the setpoints).

## 40.3.4 Constraints

### 40.3.4.1 General

Constraints can be defined in the network elements and selected for consideration in the Unit Commitment dialog. A constraint will only be active if it is specified in the network element and globally in the Unit Commitment / Dispatch Optimisation command. Some constraints can be defined as soft or hard constraints in the network elements.

**Power flow / Voltage constraints** are network constraints which can be considered for the Unit commitment simulation. The constraints can be defined as hard or soft constraints

- **Branch flow constraints (max. loading)** are available for transformers and lines.
- **Branch flow constraints (max. power) (AC only)** are an addition to the branch flow constraints based on max. loading and are available for transformers and lines if the max. loading option is selected in an AC Unit Commitment. The max. loading option calculates the sensitivities as a delta loading per control action. For the max. power option, an additional constraint is formulated which is based on the sensitivity of a power change per control action. The power representation can be adapted in the Advanced Algorithm settings (40.3.7.3). This option is beneficial if the direction of the flow on a constraint branch is changing.
- **Boundary flow constraints** can be defined for boundaries to limit the active and/or reactive power exchange over boundaries. A use case would be a limitation of the power exchange between neighbouring countries.
- **Voltage constraints of busbars (AC only):** This constraint keeps the voltage at busbars between their upper and lower voltage limit.

**Generator Constraints** comprise further limitations of generators besides the power limits. The generator constraints can be defined in the generators which can be controlled by the Unit Commitment.

- **Ramp rate constraints** define the speed that the injected active power of a generator is able to change over time (in MW/h). Moreover, there can be max. start-up ramps and shut-down ramps defined. This constraint is always considered as a hard constraint.
- **Minimum up-/down-times** specify how long a unit must run, once switched on, or remain off, once switched off. These are generally constraints with a high number of additional variables and involving time-coupling. It is advised to add them with care. This constraint is always considered as a hard constraint.
- **Spinning reserve constraints** are defined for regions (grids (ElmNet), zones (ElmZone) and areas (ElmArea)) and define the minimum value for the spinning reserve. Contributing generators to the spinning reserve are defined on the generator “Load Flow” page via the checkmark “Consider for region spinning reserve” on the “Automatic Dispatch” tab. The constraint value for the spinning

reserve is entered in MW and the available spinning reserve for a region can be determined by checking its contents. This constraint can be defined as a hard or soft constraint in the network elements.

- **Energy/Storage constraints** indicate whether the energy limits and target value at the end of a unit commitment for storage models (ElmStorage) shall be considered (see Section 40.6).

#### 40.3.4.2 Soft Constraints

Soft constraints are constraints that need not necessarily be satisfied by the optimiser. So the constraint can be violated, but this results in a cost penalty. Whether a constraint is soft or hard has to be defined in the network elements on the Unit Commitment page.

Also on the Advanced page of some network elements a different constraint type and different constraint limit can be specified for contingencies, as well as a cost scaling factor that will be multiplied, for each element, by the global costs for soft constraints.

The option **Enforce type for active constraints** offers a global control for the different constraint classes (Branch flows, boundary flows, voltages, others) on whether the specified constraint type shall be taken from each component/element or globally enforced as soft or hard constraint.

**Penalty cost type for constraint violations:** This option allows the user to force all costs for constraint violations to be linear for all soft constraints. If “Linear for all soft constraints” is selected, this overrides the selections in the network elements as to whether the cost for constraint violation shall be linear or non-linear.

**Linear penalty cost for constraint violations:** Penalty costs for elements with linear soft constraints. The value is multiplied by the individual scaling factor that can be specified in each network element.

**Non-linear penalty costs:** The use of non-linear soft constraints can be configured in this table if the non-linear option enabled. Instead of constant costs as in the linear option, the costs change for discrete violation ranges as specified in the table. The idea is to have increasing costs for more severe violations, so that a solution with a larger number of slight violations is preferred over a solution with just a few very severe violations.

#### 40.3.4.3 General Filters

The number of constraints has a big impact on the performance when executing a Unit Commitment calculation. Hence, it is beneficial to reduce the number of constraints to be considered.

The margin filters are an option to reduce the number of constraints depending on the dispatch for each time step.

**Filter for power flow constraints:** When loading/boundary constraints are formulated, the margin to the max/min allowed values before the optimisation is calculated. This value indicates the likelihood of the constraint being violated.

The setting “**Do not consider, if constraint margin is greater than**” allows the user to specify a threshold that constraints with a higher margin are not used in the formulation of the optimisation problem. This allows a leaner formulation for the solver and thus better performance, but there is a risk that some filtered constraint may be violated after the optimisation. If a filtered constraint is violated in the recalculation, a warning will be issued in the output window.

The **Filter for voltage constraints** option works in the same way as the filter for power flow constraints and is available if voltage constraints are used in a AC based Unit Commitment.

#### 40.3.4.4 Contingency Filters

The size of the optimisation problem increases significantly for each considered contingency. Therefore, reducing the number of contingencies has a big impact on the performance. There are several filters available to filter redundant and similar contingencies.

The options on this page will only be shown if contingencies are considered.

**Filter by outage distribution factor:** The outage distribution factor describes the change in power flow on a circuit as a result of the (fault) outage of another circuit. The minimum absolute and relative outage effect thresholds for consideration of contingency constraints can be entered.

**Restrict number of critical constraints:** The maximum number of contingency constraints which are formulated in the MILP for a constraint can be limited separately for branch flows, boundaries and busbar voltages. The selection of the most critical constraints is based on the effect of a contingency on the element and its margin.

**Filter equal constraints:** Several constraints can be very similar in the Unit Commitment. It can be sufficient to only considerer one of these similar constraints. Constraints are regarded as similar if the *Max. constraint value deviation* and the *Max. effectiveness deviation* are under a specified threshold. In the first step, groups of contingencies are assembled for each constraint based on the *Max. constraint value deviation* and then the effectiveness values are compared within a group. If the option *Use fast comparison heuristic* is enabled, the comparison of the effectiveness deviations are not compared one by one for each contingency but with some heuristics to increase performance.

### 40.3.5 Results/Output

#### 40.3.5.1 Results

The Unit Commitment and Dispatch Optimisation simulation writes three result files during execution.

These three files are contained as sub result files within one main result file (“Unit Commitment DC/AC”). The sub result file “Unit Commitment (summary)” stores the overall simulation results and thus contains only one row of data and is not editable.

The “Unit Commitment (before optimisation)” result file saves the load flow results of each time step before the optimisation and is equivalent to the results from a quasi dynamic simulation. The “Unit Commitment (after optimisation)” result file saves the results after the optimisation.

The control variables, such as the power setpoints of controlled machines and tap positions of controlled transformers, will automatically be recorded in the before and after optimisation result files for each time step. Additionally, the settings on the *Result*-page offer the possibility to record other results of interest in a convenient way. Besides this, the user can add other variables for recording for both result files or in the main result file as in other calculations.

The available settings for the **After optimisation** results are:

- **Record redispatch costs per time step:** In addition to the automatically recorded control value solutions, the respective costs can be written to the result file directly without the recalculation of the optimal solution.
- **Calculate and record load flow results with optimal controls:** The optimal controls are calculated by the solver and automatically recorded. If this options is selected, a second load flow sweep will be calculated, verifying the solution by applying the optimal controls to the network with this option. Only with this setting enabled, the following options for recording are available and user defined variables can be recorded.
- **Record all constraint result variables:** If this option is enabled the results of all constraints are

recorded.

The option **Record contingency results (after optimisation)** is only available if the contingency option on the basic options page is selected. If this option is enabled the full contingency result files are recorded. If curative controls are enabled, the suboption **Record curative control variables only** allows the user to limit the recording of results to the curative control variables.

The control variables are also automatically recorded for the **Before optimisation** results. In addition with the option **Record all constraint result variables** all constraints will be recorded.

#### 40.3.5.2 Output

**Output:** The user can choose whether there should be only a start and end message for the ComUc, more details such as the current calculation step (time sweep, optimisation, critical contingency filtering, etc.) or even the full output of the contingency time-sweep and optimisation.

**Max acceptable error for constraints** defines a threshold for which the optimal solution is chosen to be verified on the Results page. When the solution is verified and violations are detected, there will be sets of violation elements reported in the output window.

#### 40.3.6 Maintenance

**Planned outages** can be defined on the maintenance page of the Unit Commitment dialog. The button “**Show all**” opens the outage library folder and shows all available planned outages (IntPlannedout) of the project. The button “**Show used ones**” shows a browser with the selected planned outages for the simulation.

#### 40.3.7 Algorithm

##### 40.3.7.1 General

**Restrict simultaneous optimisation period (rolling horizon):** If this option is selected, the optimisation is separated in time periods of length number of study times. These time periods are optimised individually, but using the optimisation results of the prior periods. The overlap is not used after the optimisation; it is only there to consider an outlook on the variable behaviour for the next period. The linear problem can therefore be reduced significantly in size by solving several sub problems. So using this rolling horizon leads to a significant decrease in memory usage, an increase in performance and can help to improve solvability. Keep in mind the minimisation can only find the optimum for each single time period and the found solutions are therefore local optima. But if the time period is significantly longer than the minimum up and down times of generating units the found local optimum will be very close to the global optimum.

**Update of effectiveness for power flow/voltage constraints:** Calculating the sensitivities/effectiveness of the control variables on power flow and voltage constraints can be demanding. A recalculation is always needed when the topology of the network changes. Optionally, this can also be triggered after a fixed number of calculation steps.

There is only a minor impact of dispatch changes to the effectiveness in DC calculations. Therefore it is usually sufficient to do the recalculation only when the topology changes. For AC based calculations the effectiveness is more dependent on the dispatch and it may be beneficial for the accuracy to recalculate the effectiveness values more frequently or even for every time step. The recalculation of the effectiveness values will impact the calculation times.

**Thresholds for power flow/voltage constraints:** When loading/boundary/voltage constraints are

formulated, the effectiveness (Sensitivity) of generators, transformers, loads, HVDCs and shunts is calculated in order to estimate the constraint values with varying controls.

In the optimisation all effective controls are considered for each constraint. The effectiveness thresholds for determining which controls are to be considered can be specified relative in percent and as an absolute sensitivity. The two thresholds are linked with an “AND”-condition. Setting a low threshold will lead to a larger optimisation problem since more controls are considered for each constraint. A higher effectiveness threshold may result in more uncertainties and potential overloads after the optimisation, since some controls with small impacts are not considered but still change the resulting outcome of the constraint element.

#### 40.3.7.2 Solver

*PowerFactory* offers different possibilities to solve the mixed-integer linear program (MILP) given by the Unit Commitment and Dispatch Optimisation. There are two internal MILP solvers available, that allow the simulation of small to medium size optimisation problems:

- *PowerFactory* comes with two built in linear programming solvers, the “ip\_solve”-solver and the “cbc”-solver. The “cbc”-solver is the default and recommended, if no commercial solvers are available.
- It is also possible to use commercial external MILP solvers for optimising large-scale Unit Commitment problems. The supported external solvers are the IBM CPLEX-solver and the GUROBI-solver. The solvers are not included in the purchase of the Unit Commitment and Dispatch Optimisation module and have to be acquired separately. But the external solvers can fully integrated in *PowerFactory* and only the link to the “.dll” has to be added as administrator in the Linear Programming Configuration. The supported version for CPLEX are starting from 12.8 and for GUROBI starting from 8.0.x. With new versions of *PowerFactory*, later versions of the external solvers are supported on a regular basis. Also, a GUROBI server solution is supported to separate the solving process from the set up and recalculation of the Unit Commitment.

The solver is selected on the algorithm page of the Unit Commitment / Dispatch Optimisation command. In order to configure the solvers the User needs to log into *PowerFactory* as Administrator and access the Linear Programming Configuration. It is placed in *Configuration* → *Optimisation* → *Linear Programming Configuration* and can also be accessed via *Administration* → *Calculation Settings* → *Linear Programming Configuration*. The built in solvers are included in the *PowerFactory* installation and need no further configuration. But they can be enabled or disabled. The link to the “.dll” of the external solvers has to be added here (If the solver is marked the list of its dialog becomes editable). After configuring the solvers centrally, the solvers have to be specified for each user in the user edit dialog on the Optimisation page. This allows to restrict the available solvers for each individual user. When the problem is transferred to the solver the Unit Commitment and Dispatch Optimisation can not be interrupted since this is an external process. If the simulation shall be interrupted the user has to close *PowerFactory*.

**Basic parameters:** A time limit for the solving time of the solver can be set to avoid very long run times.

The optimisation problem is scaled by default to avoid numerical problems in the solving process.

**Solver parameters configuration:** Each MILP-solver has some input parameters which are preconfigured with defaults. The inputs are different for each solver and can change or be extended as new solver versions are linked. If the *Advanced* option is selected, a table with the available input parameters of the selected solver is offered and the values can be modified, for example the relative optimality gap (*mipGapRel*) and the absolute optimality gap (*mipGapAbs*).

### 40.3.7.3 Advanced

**Restriction of branch flows (max. power):** The constraints “Branch flow constraints (max. power)” can be based on different implementations:

- Max. active and reactive power flow: The impact of power change is considered separately for active and reactive power.
- Max. apparent power flow: The impact of power change is considered for active and reactive power combined as apparent power.
- Max. active power flow only: Only the impact of active power change is considered. The reactive power is assumed as constant.
- Max. reactive power flow only: Only the impact of reactive power change is considered. The active power is assumed as constant.

**Ignore violated constraints without effective controls:** Some violated constraints cannot be cured by the Unit Commitment and would lead to an infeasibility. An example would be an overloaded transformer that is supplying a network area with no controls and a fixed power demand.

**Handling of errors during load flow time sweep before optimisation:** A valid load flow calculation is needed for each considered point in time to initialise the Unit Commitment. If this is not the case, the network model should be adapted to achieve a valid pre-optimisation result. However, if this cannot be achieved, there are two possible options to handle points in time with no load flow solution:

- Ignore and continue: use results of last time point for time-coupling constraints (ramps, downtimes, etc.)
- Stop calculation

Errors during the load flow time sweep after the optimisation are always ignored and the calculation is continued. If there is a divergence in a contingency, the contingency will be ignored.

### 40.3.8 Parallel Computing

**Parallel computation of contingencies** is supported in Unit Commitment. In addition to a link to the user's *Parallel computation* settings, the following options are offered:

- The *Additional objects to transfer* option may be needed to transfer some external objects which are not stored in the project or the global library itself, since not all external references are automatically resolved by the sub-processes.
- The *Minimum number of contingencies* to specify the threshold number of contingencies for the use of parallel computation.

## 40.4 Handling of results

### 40.4.1 Reports

The Unit Commitment and Dispatch Optimisation toolbar includes a Report command . The available reports are:

- Optimisation results:
  - **Optimal solution report:** This report shows the resulting values for the control variables and the corresponding costs for each network element participating in the Unit Commitment.

- **Grid summary report:** The resulting costs and redispatch amounts for each time step and the complete simulation are displayed for each control element class. Different grouping elements like grid, zones and areas can be selected as base for the report.
  - **Storage report:** This report provides an overview over the stored, charged and discharged energy of storage elements and the participation if the individual generating unit is connected to the storage.
  - **Curative controls:** This report shows the deployment of curative controls over the optimisation period. Curative actions can be summarized or filtered for each time step, contingency and/or control.
- Time sweep load flow results:
    - **Loading Ranges:** This report displays the elements loaded over a threshold within a specified time range.
    - **Voltage Ranges:** This report displays the terminals violating a defined voltage band within a specified time range.
    - **Non-convergent Cases:** This report lists the points in time for which the load flow does not converge before and after the optimisation.

An extra **contingency report**  is available in the toolbar. This report opens the standard report from the Contingency Analysis with time sweep and summary results for the contingencies considered in the Unit Commitment and Dispatch Optimisation.

#### 40.4.2 Plots

The create plot button  is also included in the “Optimal Power Flow / Unit Commitment” toolbar. Typical plots for the Unit Commitment and Dispatch Optimisation are On/Off curve of generators and the redispatch over the simulation time.

The energy plots for time sweep calculations like the aggregated generation per plant category and fossil vs renewable generation are also available like in the Quasi-Dynamic Simulation.

#### 40.4.3 Colouring Mode

*PowerFactory* offers a colouring mode for the Unit Commitment and Dispatch Optimisation. The generation units and the substations can be coloured separately depending on the absolute or average (per hour) redispatch cost or the absolute (MWh) or average/relative redispatch (in percent). The relative redispatch is calculated as absolute redispatch in MWh per energy generation of a unit/in a station before the optimisation in MWh over the entire simulation time frame.

The use of colouring in network diagrams is described in Section 10.3.10.1, and more detail about the use of colours and colour palettes can be found in Section 4.7.8.

#### 40.4.4 Load Unit Commitment Results

The *Load Unit Commitment Results* functionality  allows to investigate the results of a Unit Commitment directly in the network model based on the result file. The *Load results of single point in time* option reloads the results of a specific point in time from the result file to the network elements. This allows the user, for example, to efficiently investigate the results of the critical hours before the optimisation in the network model without performing a load flow calculation. The results after the optimisation can also be loaded for a specific time. The user can select the type of results to be loaded under *Time sweep load flow results*, where the link to the corresponding result object is also displayed. The date and time for which the results are to be loaded can be specified in the corresponding fields. The available time interval is also displayed.

The *Assign time characteristics with optimal controls* functionality enables the user to write back the resulting optimal controls from the result file to time characteristics in a *new study case and variation* or directly to the *existing network*.

This allows the user to execute continuing investigations and calculations based on the Unit Commitment results directly. This offers also the possibility to execute a market simulation without network constraints in a first Unit Commitment and then carry out a second Unit Commitment with enabled network constraints to do a redispatch calculation. The old characteristics and variations are replaced by the unit commitment results. Please be aware that the applied time characteristics are therefore only valid for the time range of the unit commitment.

#### 40.4.5 Flexible Data Page

There are statistics available for variables that have been recorded for each point in time of certain elements after the execution of the Unit Commitment and Dispatch Optimisation, which can be selected in the Variable Selection on the Flexible Data page. Statistics such as average, minimum, maximum and variance are available for recorded parameters, which are based on the values of the complete simulation time.

---

**Note:** The summary result file also contains statistical result parameters from the entire simulation period.

---

### 40.5 Generating Units

It is essential for the simulation to specify which generating units shall participate in the Unit Commitment and Dispatch Optimisation and assign costs to these units. The “Operating Costs”, “Redispatch costs”, “Start-Up/Shut-down Costs” can only be defined on the Unit Commitment page of the element if the generator (ElmSym, ElmGenstat, ElmPvsys, ElmAsm, ElmXnet) is participating as a control in the Unit Commitment. Characteristics are generally supported by the Unit Commitment and Dispatch Optimisation. So all costs and related variables can be adapted during a simulation.

Generating units can be categorised in different types:

- Single thermal generating unit
- Variable renewable energy source (VRE)
- Coupled with storage model
- Part of Virtual Power Plant

#### 40.5.1 Controls and Limits

If the generating unit is specified as **Variable renewable energy source** (VRE) it belongs to the separate class of control variables. These can be curtailed to reduce overloads in the system.

If the generator is not a VRE, it can be selected whether the generator shall be part of the optimisation as a **control** variable by varying its active and/or reactive power output. This can be defined via a characteristic as well. The global setting for this control variable type is in the Unit Commitment / Dispatch Optimisation command (ComUc).

If the active or reactive power controls is enabled, the **operational limits** of the generating unit can be specified.

If the generating unit is coupled with a storage unit, either direct or via a virtual power plant, an extra control can be enabled as **Separate consumption mode** for this unit (e.g. Battery or pump storage) and an additional set of operational limits for the consumption mode can be entered. Also **Efficiency curves** can be defined for the generation and consumption mode (see Section 40.6.1)

The **Control usage** of a generating unit defines if the control is used in the *Preventive only*, *Curative only* or *Preventive and curative* modes. If curative actions are enabled, a **Max. curative power deviation** is to be specified for the control. The deviation describes the possible curative action in contingencies of the generating unit for each point in time based on the base case (load flow) value after the optimisation.

## 40.5.2 Operating Costs

The **Operating costs** for the generating units and external grids can be entered on the operational cost tab directly as “local cost definition” or specified via a “Generator Cost Curve” from the operational library.

In the **Local cost definition** the Operating/Fuel costs are entered via a table with different active power nodes. The associated cost values are in USD/h, except for the external grid and the *ElmPvsys*, where these costs are in USD/MWh and in USD/kWh respectively. This means that the slope of the curve is defined for the external grid, rather than the actual cost values (similar to the OPF). The entered list for the active power has to be in an increasing order and are the variable costs of the generating unit. The **Fixed costs** can be specified separately and are added to the variable costs in the table. Additional **Penalty costs** in USD per MWh can be assigned to a generator. An **approximation** curve, such as polynomial, spline, etc., will be used to generate a cost curve from the entered data.

---

**Note:** The default currency unit is USD. However, this can be changed on the *Project Settings* dialog (see Section 9.1.3).

---

The **Generator Cost Curve** is described in 40.5.2.1 The resulting cost curve from the entered operation costs has to be linearised in both cases for the linear programme/solver. This can be done via the **Piecewise linearisation for LP** by using the resulting average costs between the operating limits of the generating unit or linearisation of the cost curve by equidistant or user specified breakpoints. If a linearised cost curve is used the size of the optimisation problem increases with the number of break points. Therefore it is important to keep the number of breakpoints as low as possible. The **Operating costs plot** displays the cost curve from the entered operational data with the specified approximation and the linearised cost curve used in the solver.

### 40.5.2.1 Generator Cost Curves

Generator Cost Curves  offer a flexible way to enter the cost data of a generating unit. The entered costs in a cost curve can be modified during the Unit Commitment and Dispatch Optimisation simulation by characteristics. The following components can be specified:

- **Primary fuel costs** (*FuelC* in USD/MWh\_th), describe the cost for the primary energy source.
- **Fixed costs** (*FixC* in USD/h), are the fixed cost of the generating unit.
- **Emission costs** (*EmisC* in USD/t), specify the cost for the generating units emissions. The emission amount is calculated via the **Emission intensity** (*EmisInt* in t/MWh\_th) and the rated efficiency of the generating unit. The emission cost are intended to represent the costs for CO2 certificates.
- **Other thermal costs** (*OtThC* in USD/MWh\_th), are additional costs that can be added depending on the fuel amount.

- **Other electrical costs** (OtEIC in USD/MWh), are additional costs that can be added depending on the electrical generation of the generating unit.
- **Rated efficiency** (Eff in p.u.): The rated efficiency is based on the rated power of the generator and thus specified in per unit at certain percentages of the active power. A Spline, Piecewise linear, Polynomial or Hermite **Approximation** is used for the cost curve. The cost curve is displayed in USD per hour over the rated power of the machine in percent.

Based on the entered data the generator costs are calculated. The variable costs for each operation point  $i$  of the generator can be calculated as follows from the individual components based on the active power:

$$\text{VariableCosts}_i = \left( \frac{\text{FuelC} + \text{OtThC} + \text{EmisC} * \text{EmisInt}_i}{\text{Eff}_i} + \text{OtElC} \right) * \text{ActivePower}_i \quad (40.1)$$

So the resulting online costs for each operating point  $i$  of the generating unit are:

$$\text{OnlineCosts}_i = \text{FixC} + \text{VariableCosts}_i \quad (40.2)$$

The operating cost plot in the Generator Cost Curve dialog is based on a 100 MW generating unit and displays the resulting operating costs per hour over the rated efficiency. The same **Approximations** are available as in the local cost curve definition (40.5.2).

### 40.5.3 Reactive Power Costs

The **Reactive power operating costs** for the generating units and external grids can be activated and entered on the Reactive Power Costs tab (except they are part of a virtual power plant). The **Operating costs** for reactive power are entered via a table with different reactive power nodes. The associated cost values are in USD/h. The entered list for the reactive power has to be in an increasing order and are the variable costs of the generating unit. Additional **Penalty costs** in USD per Mvarh can be assigned to a generator. An **approximation** curve, such as polynomial, spline, etc., will be used to generate a cost curve from the entered data.

Note that the piecewise linear function is not differentiable at the interval limits. Since non-differentiable functions might cause problems within the optimisation routine, *PowerFactory* smooths the cost function slightly over a small range around the non-differentiable points. The width of this range can be defined by the user through the *Smoothing Factor*. A value of 0 % corresponds to no smoothing of the curve, whereas a value of 100 % corresponds to full interpolation.

---

**Note:** The default currency unit is USD. However, this can be changed on the *Project Settings* dialog (see Section 9.1.3).

---

The resulting cost curve from the entered operation costs has to be linearised for the linear programme/solver. This can be done via the **Piecewise linearisation for LP** by using the resulting average costs between the operating limits of the generating unit or linearisation of the cost curve by equidistant or user specified breakpoints. If a linearised cost curve is used the size of the optimisation problem increases with the number of break points. Therefore it is important to keep the number of breakpoints as low as possible.

The **Reactive power operating costs plot** displays the cost curve from the entered operational data with the specified approximation and the linearised cost curve used in the solver.

#### 40.5.4 Redispatch Costs

**Additional active power redispatch costs** can only be entered if the generating unit is not a VRE and the *Active Power* control is activate. The upward and downward redispatch cost can be defined separately for the active power. If the generating unit is flagged as a VRE only the **Costs for curtailment** can be specified.

**Additional reactive power redispatch costs** can be selected for all available generator usages if the *Reactive power* control of the generating unit is activate. Three different options are available:

- **Off:** Additional redispatch costs for reactive power are not considered
- **Based on reactive power deviations:** The *Costs for reactive power deviation* are entered in USD/Mvarh and have no differentiation between capacitive or inductive reactive power. The *reference value for the redispatch costs* can be defined as:
  - **Q load flow value** (default): Deviations from the reactive power resulting from the load flow calculation prior to the optimisation are penalised. Depending on the setting on the *Load Flow* page of the generator, the Q value can be either a user-defined operating point (e.g. *Const. Q*) or the result of a voltage control.
  - **Q target value:** Deviations from the reactive power target value that can be specified in the field *Q target value* in Mvar are penalised.
- **Based on voltage setpoint deviations of controller:** In reality (e.g. for grid operators), it is common practice to specify voltage setpoints for the generators that cause changes in the reactive power instead of specifying Q setpoints directly. With this option, the Unit Commitment minimises voltage setpoint deviations of the machine. The *Costs for voltage setpoint deviation* are entered in USD/p.u.. The *reference value for the redispatch costs* can be defined as:
  - **Controller voltage setpoint** (default): Deviations from the controller voltage setpoint defined in the generator or in the assigned Station Control are penalised. Depending on the setting on the *Load Flow* page of the generator, the voltage setpoint can be either the user-defined setpoint (if *Const. V, Voltage Q-droop, Voltage Iq-droop* is selected) or the voltage of the local terminal of the generator resulting from the load flow calculation prior to the optimisation.
  - **Target voltage setpoint:** Deviations from the voltage setpoint target value that can be specified in the field *Target voltage setpoint* in p.u. are penalised.

---

**Note:** This option does not support generators which are part of a Virtual Power Plant or a Station Controller that does not specify a voltage setpoint or controls more than one machine.

---

**Note:** The target reference value is only considered, if the *Time Coupling* for the corresponding generator usage on the *Objective Function* page of the Unit Commitment command is set to *Reference value of current time* (see Section 40.3.2.2). If the other option is selected, the *Optimised value of previous time* will be the target reference value. In case the option *Control variable values outside calculation time period* for *Before time period* is set to *Non-optimised values of first calculation time* (see Section 40.3.3.3), the load flow calculation result (e.g. Q) of the first time step prior to the optimisation is used as the target reference.

---

**Fixed costs per setpoint deviation** can be specified to represent the effort for each setpoint change, no matter how small. This can be useful to reduce the total number of measures, because the required reactive power is shifted to a generator, which has to change its setpoint anyway.

#### 40.5.5 Start-Up/Shut-down Costs

The **Shut-down costs** of a generating unit can be specified as a fixed amount in USD. There are different possibilities to define the **Start-up costs**:

- **Constant:** The cost can be entered as a fixed amount equal to the Shut-down costs.
- **Warm-/Cold start-up costs:** The Start-up cost can be entered for a cold start and a warm start. The cold start time specifies the downtime of the generating unit after which the start up costs are switched from warm- to cold start.
- **User-defined discretisation:** The start up costs for different down times can be specified via a table. Typically, the temperature dependence is exponential in time which can be modelled via a step function here. Please note that these costs might drastically affect the performance of the solver.

The cost for the Start-up over the down time of the generating unit are displayed in the diagram on the right side of the tab.

#### 40.5.6 Constraints

The constraints of a generating unit have to be defined in the unit itself. In the Unit Commitment / Dispatch Optimisation command it can be specified whether a specific constraint should be regarded in the simulation. The available limitations of generating units are:

- **Ramp rate constraints** The limitation for the ramp rate is distinct between ramp rates when the generator is on-line and start-up/shut down ramp limits. The up and down ramp limits are defined in MW/h or in p.u./h. The start up and shut down ramp limit can be entered in MW or p.u. This means the first/last step of the generator is limited to a certain value.
- **Start-up/shut-down time constraints** When the optimiser decides to switch a machine off, it will remain off for at least this specified time period. Similarly, a minimum up time can be defined.

When a generator is set as a control variable, the optimisation might decide to switch it off at some point in time. This can be prevented with the **Must run** flag. It will prevent “Switch off” actions and can be set via a characteristic. Note: Switch-on actions can be prevented by using planned outages.

**Fix controls to Load Flow values** is intended to be controlled by a characteristic and it allows the user to specify the simulation points for which the generation unit is not participating in the Unit Commitment and Dispatch Optimisation and so the operation point is kept at the load flow value.

The option **Restriction to discrete active power values** offers the specification of allowed discrete operating points in percent. Herewith, the range for active power setpoints of controlled generators or Virtual Power Plants in a Unit Commitment calculation can be restricted to a number of discrete values defined in terms of the rated power. New rows can be added by right clicking in the table and selecting the corresponding option from the context menu. The current operating point is added to the list with the option *Allow current active power value* by default.

## 40.6 Storage Units

Storage models (*ElmStorage*) can be linked with generators and allow the generated and consumed energy of the generators to be restricted.

Storage models can be linked directly with one generating unit (40.5) or with several generators via a virtual power plant (40.7). A storage model can only be linked to one generating unit or one virtual power plant. There are three storage types available:

- **Battery:** This represents storage without any inflow but with a self discharge. The **Capacity** can only be entered as energy in MWh.
- **Hydropower:** This represents storage with the possibility to have a natural inflow. Also all **Energy** figures can be entered directly in MWh or as **Water volumes** which are then converted with the **Head of water**.

- **General Storage:** This option allows the user to enter all storage properties in MW and MWh, without specifying the storage technology.

The available constraints for the unit commitment are:

- Operational energy limits
- Energy constraint at study period end
- Energy at study period start
- Water spillage/self discharge
- Inflow/Additional charge

The rolling horizon ([40.3.7](#)) functionality will cause some drawbacks in the optimisation of storage units, since the storage is then only optimised for the individual time range.

#### 40.6.1 Efficiency curves

Storage models are linked to generators via efficiency curves. The efficiency curves can be distinguished between generation and consumption mode. If no efficiency curve is defined the efficiency of 1 p.u. or 100 percent is used. The **rated efficiency** of the conversion between stored energy and the electrical infeed of a generating unit can be entered via a table and is then approximated. The efficiency curve is then linearised for the optimisation. The options are:

- Use average efficiency w.r.t. operational limits
- Linearise efficiency between operational limits
- Use efficiency at max. operational limit
- Use efficiency at min. operational limit
- Use user-defined breakpoints

The recalculation is done exactly with the determined control values and small mismatches due to linearisation imprecision can occur.

### 40.7 Virtual Power Plants

The basic functionality and handling of virtual power plants is described in [15](#). This section focusses only on the aspects of a virtual power plant for a unit commitment. The idea of a virtual power plant in the unit commitment is to combine several generating units into one unit for the optimisation. Possible examples are the aggregation of all turbines of an wind park or the combination of all generators in one region. Also virtual power plants can be used to link several generator/turbines of a hydro power plant to one storage model (see Section [40.6](#)). The available types of power plant usage of the virtual power plant are:

- Thermal generation unit
- Variable renewable energy source (VRE)
- Coupled with storage model

The usage types and the available constraints are, thus very similar to a single generating unit (see Section [40.5](#)). In the virtual power plant the single limits and costs can either be aggregated from the single units or entered globally in the virtual power plant. When using a global constraint the aggregated value for the limits of the individual units should be respected, otherwise this may lead to infeasibility.

The unit commitment returns only one value for a control for the virtual power plant. This figure, for example the active power contribution of the virtual power plant, has then to be distributed to the single units within the virtual power plant. The options for the **Power distribution** to the single generating units are:

- According to rated power
- According to dispatched power
- According to generation shift keys

If the reactive power is controlled, the same distribution is used.

---

**Note:** If the option *Redispatch costs based on voltage setpoint deviations* is used in a generator, which is part of a virtual power plant, additional reactive power redispatch costs for this generator are ignored.

---

The **Respect individual operational machine limits** flag will ensure that the individual limits of the machines are respected in the distribution after the optimisation.

The virtual power plant is always a simplification for the optimisation in unit commitment.

## 40.8 Other Network Elements

### 40.8.1 Transformers, voltage regulators and shunts

The tap positions of transformers, voltage regulators and shunts can be defined as **control** variables to govern active/reactive power flows. This control variable can be modelled as a discrete variable directly or as a continuous variable, which simplifies the algorithm.

The **Penalty costs per Tap deviation** can be specified in the element as well. The participation as a control can be specified separately for each tap for 3- and 4-winding transformers

The **maximum loading** of transformers and voltage regulators can be defined as a hard or soft constraint.

#### 40.8.1.1 Shunts

In addition to the above described settings, two further options are available for shunts:

- **Allow control to be switched on/off:** If this option is enabled two additional fields appear, in which *Switch-on costs* and *Switch-off costs* can be defined. The cost optimisation can then switch the shunt off at a high tap position and switch it on again later if this minimises the costs, instead of reducing the tap position by several positions and increasing it again to the original position.
  - **Tap limits:** Allows to specify a flexible range of tap positions to be used in the Unit Commitment.
- 

**Note:** The Unit Commitment is not allowed to switch a shunt if there are ambiguously associated switches.

---

### 40.8.2 Tap Controllers

If several transformers are connected in parallel or should be handled as a group, a tap controller can be used. The tap controller aggregates all selected transformer as one control for the unit commitment and guarantees coherent tapping of the transformers.

### 40.8.3 Loads

Loads (ElmLod, ElmLodlv, ElmLodmv) can also be used as control variable for the Unit Commitment and Dispatch Optimisation by **allowing load shedding**. **Costs for load shedding** can be assigned to each load in costs per MWh. The limits for the shedding of the load can be specified in percent of the load's apparent power. The Unit Commitment then sheds the load based on the before optimisation load flow value. Note: For DC Unit Commitment this is only an active power value.

### 40.8.4 Boundaries

The minimal and maximal active and reactive power flow over a boundary can be defined for the Unit commitment. Each component can be activated separately and it can be defined whether the entered active and/or reactive constraints should be soft constraints for the Unit Commitment and Dispatch Optimisation.

The limits for the boundary flow can be specified as absolute limits or relative to the pre optimisation load flow value.

### 40.8.5 Terminals

The voltages on the terminals can be limited for the Unit Commitment and Dispatch Optimisation. These constraints can be entered as upper and lower limit in p.u. and specified as hard or soft constraints.

### 40.8.6 Lines and Bays

The **maximum loading** of lines and bays can be defined as a hard or soft constraint.

### 40.8.7 Regions: Grids, Zones and Areas

A minimum value for the spinning reserve in MW can be defined for regions (grids (ElmNet), zones (ElmZone) and areas (ElmArea)). This can be defined as a hard or soft constraint. The available reserve can be determined via the summation of the active power of the machines of the region.

### 40.8.8 HVDC Converters

The active and reactive power flow over HVDC branches can be controlled by the Unit Commitment. The controls on the *General* tab on the *Unit Commitment* page of the converter (*ElmVsc*, *ElmVscmono*) can only be activated if the *Control mode* (see *Load Flow* page) allows control of P and/or Q (e.g. *P-Q*). The available settings on this page are very similar to those of a generator. For more information refer to Section 40.5.1. More information about the settings on the *Reactive Power Costs* and *Redispatch Costs* tabs can be found in Section 40.5.3 and Section 40.5.4.

## 40.8.9 Advanced constraint settings

On the “Advanced” tab on the “Unit Commitment” page of network elements which represent network constraints in the unit commitment, such as lines, transformers, bays, boundaries and terminals (main page), some additional settings are available:

### 40.8.9.1 Max. loading constraint for contingencies

Separate constraint types and limits can be specified for each element for contingency situations.

### 40.8.9.2 Penalty cost for soft constraints

The cost for soft constraint violation is globally specified in the Unit Commitment command. The cost can be specified as **linear** or **non-linear**, and for each constraint element it is possible to select which option shall be used. The **Cost scaling factor** gives an additional possibility to scale the soft constraint penalty for the element individually.

### 40.8.9.3 Constraint filtering

In order to improve performance, it is possible to disregard certain less critical constraints by using the filters of the Unit Commitment command. If certain constraints have to be considered in a certain setup, the filtering can be disabled in for each network element by deselecting the **Allow filtering by constraint margin** and/or **Allow contingency filtering by number of critical constraints**.

## 40.9 Troubleshooting

### 40.9.1 Solver Selection

The solver selection for the optimisation problems has not only an influence on performance but also in the solvability itself. Some optimisation problems might not be solvable with the included solvers and using a commercial solver might be beneficial. This can be the case for large and complex problems with a lot of time dependencies.

### 40.9.2 Soft Constraints

The optimisation can become infeasible if a set constraint cannot be kept or cured with the given controls. To avoid this soft constraints can be used. To influence the effect the price for violating soft constraints can be set in the Unit Commitment command.

### 40.9.3 Performance

The total calculation time for a Unit Commitment execution in *PowerFactory* can be split in several parts, with different possibilities for impact on the performance:

- Calculation of the original load flow results for each point in time to initialise the Unit Commitment: The only possibility to increase performance in this step is to gain time in the load flow calculation itself. Generally the DC load flow should be faster than the AC load flow. If AC load flow is

selected, improving the convergence behaviour can help; if possible, reduce the number of outer loop iterations required.

- Calculation of sensitivities for the selected controls on the selected constraints: When formulating the MILP the sensitivities are needed. Every topology change requires the calculation of new sensitivities. If the topology does not change, the sensitivities can be assumed as constant. If this is done the number of sensitivity calculations can be decreased (see Section [40.3.7.1](#)).
- If contingencies are considered it is possible to use the linearised method which is generally faster. Also parallel computation of contingencies is supported.
- Solving the MILP using an external solver: The time for solving the MILP itself can be influenced by the solver selection and the adjustment of advanced solver parameters. Generally the commercial solvers (CPLEX and GUROBI) are faster and are beneficial for large Unit Commitment problems. The solving time can increase strongly (non-linearly) for more complex problems. Thus it is generally recommended to use as few controls and formulate as few constraints as possible when setting up a Unit Commitment, if performance is important. In addition, discrete controls and time dependencies are expensive for the performance of the MILP. The rolling horizon also offers the possibility to split up a large problem in several smaller problems which can help with performance and also solvability.
- Recalculation with optimal controls is an optional feature to check the validity of the solution and obtain the corresponding results. This step is optional and can be skipped to gain performance.

#### 40.9.4 Voltage Controlling Elements

If the network includes a lot of voltage controlling elements with narrow limits it can be the case that certain setpoints cannot be kept after the optimisation and thus lead to non-convergent cases when using an AC load flow.

#### 40.9.5 Reference Machine

If the reference machine is not included as a control in the Unit Commitment the optimisation as a degree of freedom will reduce the power of the controlled units. This delta will then be compensated for by the reference machine. If the reference machine shall not actively participate but the described power shift shall be avoided, the additional redispatch costs of the reference machine can be set to a very high value.

The active power control can also be crucial for the load flow convergence after the optimisation. A single slack might not be able to balance the active power especially if contingencies are considered. An option to improve the convergence behaviour can be to select a distributed slack by generators.

#### 40.9.6 Not regarded Constraints

In order to reduce the optimisation problem the Unit Commitment offers a constraint filter on the Algorithm page. Some constraints may be neglected in the optimisation because of this filter. Also the optimisation is always executed on the linearised system. This can result in deviations when recalculating the system with a non-linear AC load flow. In general warnings will be raised if a constraint is violated.

#### 40.9.7 Rolling horizon and time dependencies

The rolling horizon ([40.3.7](#)) function will impact the optimisation of time depended constraints strongly. This means constraints for max. downtimes and ramps can only be optimised for the specified time

range given by the number of simultaneous time steps and the step size. The overlapping time steps with next period allows to consider these dependencies. The optimisation of the storage units is also just done for the time specified with the rolling horizon and not globally if the rolling horizon is used. It is recommended not using the rolling horizon if working with storage units.

## 40.10 Redundant Constraint Filter

The “Redundant Constraint Filter”  is an extension tool of the Unit Commitment toolbox and offers a powerful possibility to determine the relevant and limiting network constraints for a selection of controls. The filter compares all the selected constraints and flags the limiting ones for the Unit Commitment. Each constraint which is not flagged by the filter is therefore redundant and can be neglected for this particular Unit Commitment setup.

In the Constraint Filter, contingencies are fully supported. Hence, it is possible to determine the Critical Branch Critical Outage combinations (CBCOs or also called CNECs for Critical Network Element and Contingency) for a single point in time or a specified time period. The *Redundant Constraint Filter* also offers an option to replicate the mechanism of the flow based market coupling (FBMC) in Central Western Europe (CWE).

### 40.10.1 Basic Options

The *Redundant Constraint Filter* contains a link to the **Unit Commitment** command. The selected settings in the Unit Commitment command determine which controls and constraints shall be regarded and which contingency cases are considered. Also the constraint filters from the Unit Commitment command can be used.

The individual **Filters** are available and can be applied selectively:

- **Apply constraint filters from Unit Commitment:** In the Unit Commitment command, constraint filters and filters for contingency constraints can be specified ([40.3.4.3](#) and [40.3.4.4](#)). These constraint filters are applied if this option is selected.
- **Apply filter w.r.t control variable bounds:** If several controls have an effect on a constraint, the filter checks whether the constraint would be violated if the respective controls were on their variable bounds. If there is no possibility for a violation, the constraint is redundant.
- **Check redundancy w.r.t. contingency constraints:** All contingencies are compared to find the critical constraints for each constraint element. This filter supports parallel computing.
- **Check redundancy w.r.t. whole system:** Each constraint is checked against the entire system regarding all controls and the remaining constraints to identify further redundancies.

### 40.10.2 Algorithm

#### 40.10.2.1 General

**Additional settings** for the redundant constraint filter are:

- **Relax constraint violations:** If a constraint is already violated, either in the base load flow or a contingency, this constraint will always be limiting in the MILP. If these violation are relaxed, the overloads will be ignored and not be resolved. This means technically, that the loading limit of each overloaded element is set to maximum loading value of this element in any contingency.
- **Detailed constraint redundancy detection:** This is only relevant for loading constraints. If a branch element has several ends, the loadings are different for each end. Thus only the highest

loading of an element is considered. If this setting is active the loadings from all sides are considered.

- **Tolerated violation for redundant constraints:** If the violation difference on a branch is smaller than the specified threshold it is regarded as redundant.

The **Modelling of discrete controls** is possible as discrete or continuous.

**Soft constraints** can be ignored or considered as hard constraints.

#### 40.10.2.2 Advanced

##### Advanced settings:

- **Apply multi-threading for redundancy check:** Multi-threading can be used to increase performance. The multi-threading can only be applied in the “Check redundancy w.r.t. contingency constraints” filter
- **Apply filter w.r.t control variable effectiveness:** This filter is designed to detect constraints which have a relevant sensitivity to the selected control variables.

The **Initialisation of redundancy checks** enables a selection between two algorithmic approaches. Either starting with **No constraints** and adding constraints to the list of non-redundant constraints or starting with **All constraints** and removing redundant ones accordingly. The result should be quite similar but there can be differences since different sets of constraints can constrain the same set of controls. The default is to start with “No constraints”, for better performance.

#### 40.10.2.3 Advanced - Flow Based Marked Coupling

The setting **Consider minimum RAM and net positions** enables the replication of the flow based market coupling (FBMC) mechanisms of Central Western Europe (CWE) in order to determine the relevant critical branch / critical outage combinations (CBCOs). This option is only available if the Unit Commitment command is set up for DC load flow calculation and usually only applied for one point in time.

In the FBMC, a minimum availability margin (minRAM) for each considered branch has to be reserved for the market from the branch capacity. Thus, the **Min. RAM** is equivalent to a loading percentage of a branch that has to be available for the market. The branches to be considered as potential critical branches have to be selected as constraints on the Unit Commitment page of each branch element and the branch flow constraints have to be selected globally in the Unit Commitment command.

As a basis for the FBMC, an initial guess of the load flow situation for the hour of interest is needed to determine the flows without trades in this situation. This is usually calculated with a prior market simulation based on Net Transfer Capacities (NTCs) between the market regions and estimates for load and renewable generation.

In order to calculate the fictive flow on each considered branch without market flows, it is assumed that there will be no market flows between regions, if each region supplies itself. Therefore the **net positions**, the difference between generation and demand of a region, is used to fictively correct the branch flows from the load flow solution of the initial guess.

The net positions are shifted by a **Delta net position** for each region so that the region is supplying itself (the generation is increased by the given value). However, special cases may apply for setting the delta net position differently (for example because of fixed interchanges via HVDCs). The sensitivities for the regions are calculated for each constraint branch and the resulting matrix is multiplied by the vector of the *Delta net positions* in order to get the branch flows/loadings without market flows.

The region itself cannot be selected directly in the net position table, because the Redundant Constraint

Filter can only work with elements that are potential controls in the Unit Commitment. Therefore all the relevant power plants of a region, which are active in the market, have to be added to one Virtual Power Plant (ElmBmu) with their respective generation shift keys (GSKs). For the Virtual Power Plant the respective delta net position of the region has to be specified. The Virtual Power Plants have to be configured as for a Unit Commitment simulation: The active power control flag has to be enabled for the Virtual Power Plant and the contained generators. Additionally, the contained generators should have *Generator usage* set to "Part of Virtual Power Plant".

The **Fixed trading volume** in the net positions table has to be set for the regions which have a fixed trading volume and are not considered to be flexible in the later flow based market simulation. An example of this would be a fixed trading volume from a neighbouring region that is not part of the regions used in the later flow-based market simulation. This has to be specified for the filter since the fixed trading volume is not controllable, but is part of the market flows and thus included as such in the minRAM evaluation. To consider this accordingly, the Virtual Power Plant has to be configured identically to the controllable ones and a delta net position has to be specified. The filter uses the *Fixed trading volume* information to exclude the virtual power plant as an active control.

The base flows (without market flows) are then calculated with the delta net positions and the corresponding sensitivities for the virtual power plants. Then, the allowed maximum loading is determined by checking if the specified minRAM is available on each branch. If not, the maximum loading limit is relaxed to meet the minRAM requirement. From this point the Redundant Constraint Filter applies the selected filters described in the previous subsections.

The "Redundant Constraint Filter" marks the relevant/critical CBCOs in the result file (40.10.3). If information about the corresponding sensitivities and margins of the CBCOs is needed, post-processing (via script) in combination with the Sensitivity Analysis is required.

### 40.10.3 Results

The **Results** are written to a result file. The result file is a matrix where the columns are all the regarded constraints in the Unit Commitment and the rows correspond to the selected contingencies (The first row represents the base case). The non-redundant Critical Branch Critical Outage (CBCO) combinations are indicated with a "1"

The option **Write results to output window** complements the result file and gives the information about the non-redundant constraints to the output window.

The **Verification w.r.t. final system** can be used to evaluate the selection made by the contingency filter and the whole system filter. Due to numerical reasons it can happen that a constraint is invalidly flagged as redundant. The verification goes over all the filtered constraints and compares them with the final solution. If one of the constraints is not redundant, taking into account the **Add. tolerated violation for redundant constraints**, it can be reported and/or added to the list of non-redundant constraints. The following options are available for the verification:

- Off
- Report non-redundant constraints only
- Reactivate non-redundant constraints only
- Reactivate and report non-redundant constraints

#### 40.10.3.1 Report

The dedicated report  can also be used to visualise the non-redundant constraints found by the filter.

# Chapter 41

## Transmission Network Tools

### 41.1 Introduction

The chapter presents the transmission networks tools available in *PowerFactory*:

- PV Curves Calculation (Section 41.2.1)
- PV Curves Plot (Section 41.2.2)
- QV Curves Calculation (Section 41.3.1)
- QV Curves Plot (Section 41.3.2)
- PTDF Calculation (Section 41.4)
- Transfer Capacity Analysis (Section 41.5)
- Flow Decomposition (Section 41.6)

These tools are accessible as illustrated in Figure 41.1.1, and this chapter introduces each respective tool, providing general descriptions and details pertaining to the command dialogs.

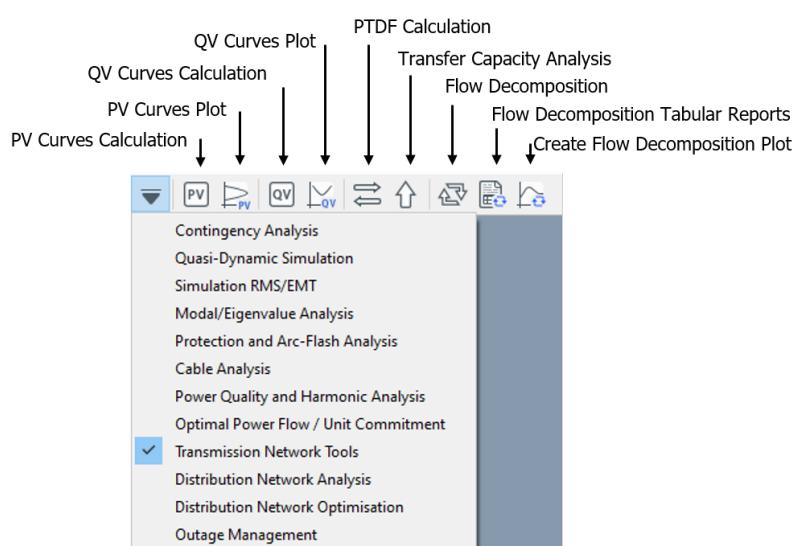


Figure 41.1.1: Accessing Transmission Network tools

## 41.2 PV Curves

PV curves are essential for analysing the voltage stability of power systems. The PV Curves Calculation finds the critical point of voltage instability by increasing the power demand of user-selected loads until the load flow calculation no longer converges; i.e. until the stability limit is reached. The critical demand is reported in the output window, and the voltage drop (V), and increasing power (P), can then be plotted using the PV Curves Plot command, as illustrated in Figure 41.2.1.

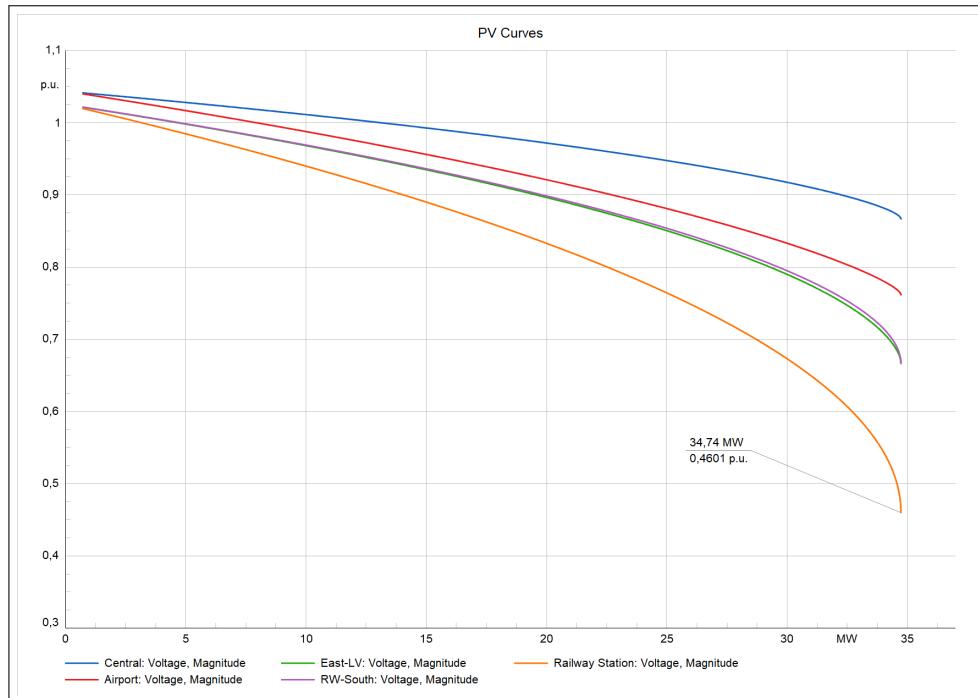


Figure 41.2.1: Network PV curve

The calculation and the corresponding plots are made available via two separate commands, both of which are available either via the *Calculation* main menu, under *Transmission Network Tools*, or the main toolbar using the following icons:

- PV Curves Calculation (calculates the critical demand); see Section 41.2.1.
- PV Curves Plot (plots the PV curve); see Section 41.2.2.

In addition, the PV Curves Calculation is accessible via the single line diagram or Data Manager, when right-clicking on network element/s and selecting *Calculation → PV Curves...*

### 41.2.1 PV Curves Calculation

The different pages options of the PV Curves Calculation command are explained below.

#### 41.2.1.1 Basic Options

##### Calculation

Selection of either **AC load flow, balanced** or **AC load flow, unbalanced, 3-phase (ABC)**. The **Load Flow** button provides access to the Load Flow Calculation command settings. It should be noted that some

load flow settings, such as *Automatic Tap Adjust of Transformers*, have a strong influence on the PV Curves Calculation results. Decreasing the setting *Break if no progress in X iterations* (e.g. from 10 to 5) may significantly improve performance for some large systems. This setting is available in the Load Flow Calculation command, *Iteration Control* page, *Advanced Settings* tab.

### Consider contingencies

If this option is ticked, an existing Contingency Analysis command can be selected. The PV Curve Calculation will apply each defined contingency and calculate the resulting PV curves accordingly.

Multiple time phase and single time phase contingencies are considered. The post-fault time of the contingency is considered in the calculation when applying the contingencies, and must already be defined in the Contingency Analysis command.

Regardless of whether contingencies are considered, the beginning of the PV Curve Calculation is always executed with no contingencies in order to calculate the base case.

### Scale loads

The loads that are to be increased can be those contained in either the *Whole system* or a *User-defined selection*.

Loads with a value of zero as operating point are neglected. In order to consider negative loads and motors, the corresponding checkboxes should be ticked.

### Record terminal results

Selection of terminals for the PV curves may be either *All busbars in system* or a *User-defined selection*. The latter may include terminals defined as internal nodes. The voltage and the gradient of the voltage for each terminal are saved at each iteration in the file specified by *Results*. For each contingency a separate sub-results file is created. These sub-results files can be accessed via the primary results file (i.e. that specified by *Results*).

#### 41.2.1.2 Iteration Control

##### Step size algorithm

Two step size algorithms are available for selection:

- Binomial search: doubles the step size up to the maximum step size (until the point of divergence) and then continuously halves it until the minimum step size is reached.
- Adaptive step size (default): reduces the step size before reaching the stability limit.

##### Step size definition

Definition of the step size and the number of iterations, and therefore influences the accuracy, calculation speed and smoothness of the plotted curves.

The *Minimum step size* defines the threshold of the iteration: it stops when the iteration step gets smaller than this value, which ends the PV Curve Calculation of this case. The smaller this value is, the closer the calculation gets to the stability limit, however more iteration steps are required which increases the calculation time. Adjustment of the *Maximum iterations* is not recommended, rather adjustment of the maximum step size (or the *Multiplication factor*, depending on what is of interest). The units of the initial-, maximum- and minimum step size are in percent of the initial power demand of each load.

### **Initial load scaling**

Defines the starting point for the load scaling and therefore the starting point of the iteration. The *Multiplication factor* provides the user with the means to choose a different starting point for the calculation. When set to '1', the PV Curves Calculation will start from the current operating point of the selected loads.

### **Show detailed output**

Provides information about each iteration, allowing the user to fine-tune the step size. For further information refer to Section [41.2.3.1](#).

## **41.2.2 PV Curves Plot**

By default, the PV Curves Plot command plots the critical busbar of the critical contingency. If the calculation has been made with no consideration of contingencies, the base case will automatically be selected. The PV Curves Plot options, outputs and results are explained in Section [41.2.2.1](#) - Section [41.2.3.4](#).

### **41.2.2.1 Data input**

The results file specified in, and created by, the PV Curves Calculation command (see Section [41.2.1](#)).

### **41.2.2.2 Busbars**

The selection of busbar/s for which PV curves are to be plotted. This can be set to *Only plot critical busbar* or *User-defined selection*. If the PV Curves Calculation has been carried out with no consideration of contingencies, the base case will be selected by default. If the calculation has been carried out with consideration of contingencies, the critical busbar of the critical contingency will be plotted.

### **41.2.2.3 Contingencies**

If the PV Curves Calculation has been executed with consideration of contingencies, the user can either choose to plot the busbar of the critical contingency (default), or may select the busbar/s to be plotted from the list of calculated contingencies. In the latter case, the contingencies should be selected from the *Contingencies* table before selecting the associated busbars. For a selection of multiple contingencies, the option *Plot each contingency in a separate sub-plot* causes the selected busbars to be plotted for each contingency.

## **41.2.3 Outputs and Results**

### **41.2.3.1 Output Window**

During the calculation, the initial-, scalable- and critical demand are printed out for each contingency. If the detailed output option is selected, the number of successfully converged load flows and the total number of load flows for each contingency are displayed, as well as the load flow convergence, current scaling factor and demand for each iteration-step (per contingency).

Following the calculation, a 'PV Curve Study Summary' is printed in the output window, summarising every contingency by showing the limiting bus and the power demand at the critical point of instability.

The contingencies are sorted according to the power demand at the critical point. The most critical contingency is reported; it is the contingency where the critical point has the smallest power demand.

#### 41.2.3.2 Single Line Diagram

After the calculation, the single line diagram displays the results of the load flow calculation for the most critical contingency at the critical point of calculation.

#### 41.2.3.3 Results File

For every contingency a results file is created which contains the name of the contingency and a result matrix which is organised as follows:

- The rows of the matrix represent each iteration step of the calculated power demand. The last row is the maximum power demand and thus the critical point of the system.
- There are two columns for every busbar in the calculation: the first contains the voltage of each iteration-step, and the second displays the gradient of the voltage (compared to the previous row/iteration-step). After the calculation has finished, this gradient is also available on the Flexible Data page for busbars, by selecting the calculation variable *b:uGradient* in the Variable Selection editor.

#### 41.2.3.4 Estimated Critical Busbar

The estimated critical busbar is the one with the highest gradient in the last converging iteration-step (i.e. in the last row of the results file). If the total voltage drop from the first to the last iteration of this bus is less than half the drop of the bus with the maximum drop, then the bus with the maximum drop is considered to be the critical bus. The total drop of the busbar is checked in case the final gradient is positive. This occurs when the load flow solver finds the wrong solution (e.g. in the iteration before the last step), which might happen when the load scaling is too large near the stability limit.

### 41.3 QV Curves

QV curves are very useful when analysing voltage stability of power systems. The QV Curves show the sensitivity and variation of bus voltages with respect to injected reactive power. Since all reactive power control devices are designed to operate satisfactorily when a increase of reactive power (Q) is accompanied by an increase in voltage (V), operation on the right side of the curve is stable and on the left side unstable. The bottom of the curve, where the derivative  $dV/dQ$  is equal to zero, represents not only the voltage stability limit but also the minimum amount of reactive power required for stable operation.

When executing a QV Curves Calculation, the critical reactive power and voltage are reported in the output window. The reactive power (Q) and corresponding voltage drop (V), can then be plotted in a QV curve, as shown in Figure 41.3.1.

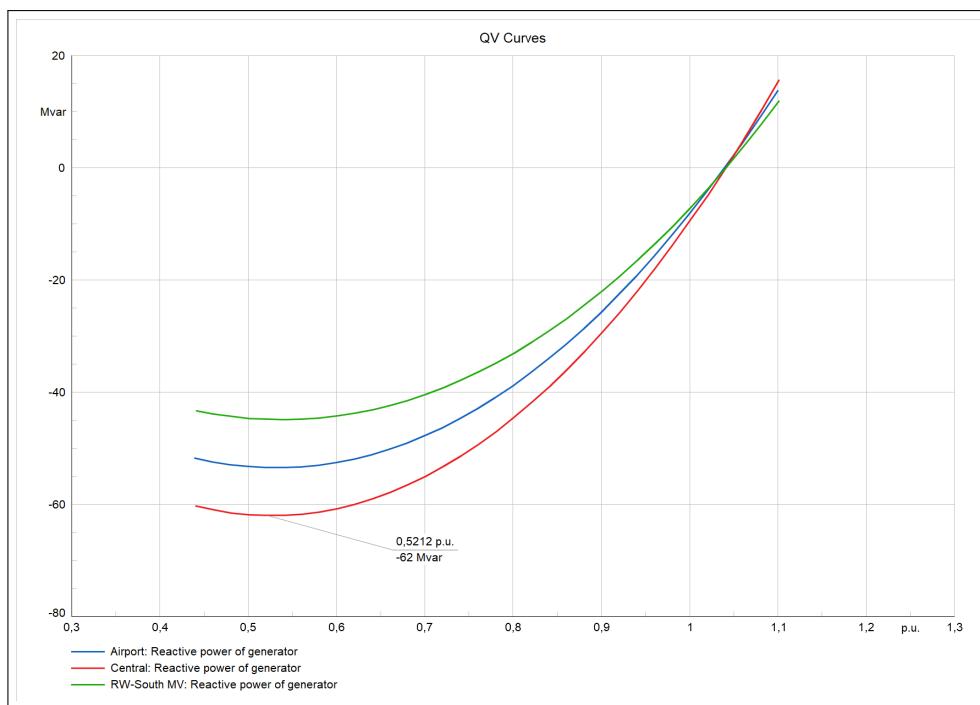


Figure 41.3.1: Network QV curves

To execute a QV Curve calculation, at least one busbar should be selected. If there is more than one study busbar, they will be calculated iteratively and not simultaneously.

The calculation and the corresponding plots are made available via two separate commands, both of which are available either via the *Calculation* main menu, under *Transmission Network Tools*, or the main toolbar using the following icons:

- QV Curves Calculation  (executes the calculation); see Section 41.3.1.
- QV Curves Plot  (plots the QV curve); see Section 41.3.2.

In addition, the QV Curves Calculation is accessible via the single line diagram or Data Manager, when right-clicking on network element/s and selecting *Calculation* → *QV Curves...*.

## 41.3.1 QV Curves Calculation

The different pages options of the QV Curves Calculation command are explained below.

### 41.3.1.1 Basic Options

#### Calculation

Selection of either *AC load flow, balanced* or *AC load flow, unbalanced, 3-phase (ABC)*. The *Load Flow* button provides access to the Load Flow Calculation command settings. It should be noted that some load flow settings, such as *Automatic Shunts Adjustment*, have a strong influence on the QV Curves Calculation results.

### Consider contingencies

If this option is ticked, an existing Contingency Analysis command can be selected. The QV Curve Calculation will apply each defined contingency and calculate the resulting QV curves for each study busbar accordingly.

Multiple time phase and single time phase contingencies are considered. The post-fault time of the contingency is considered in the calculation when applying the contingencies, and must already be defined in the Contingency Analysis command.

Regardless of whether contingencies are considered, the beginning of the QV Curve Calculation is always executed with no contingencies in order to calculate the base case.

### Analysed nodes

The terminals where the reactive power will be injected. When clicking on the select button (▼) the list of busbars relevant for calculation is displayed. When a group of busbars is selected, a set named “QV Curves Set” is automatically created inside the active study case.

---

**Note:** The analysis on 1-phase and 2-phase nodes is not supported.

---

### Results

The voltage for each terminal is saved at each iteration in the file specified by *Results*. For each contingency a separate sub-results file is created. These sub-results files can be accessed via the primary results file (i.e. that specified by *Results*).

#### 41.3.1.2 Voltage Iteration

##### Voltage range

Two options are available for selection:

- Global voltage range for each terminal: a global maximum and minimum voltage will be set for all the busbars.
- Range around base case voltage of each terminal: the solution of the load flow will be used as base voltage for each terminal, afterwards a range around this voltage can be defined for all the selected busbars.

##### Voltage iteration

On this part it is possible to select the start (maximum) and end (minimum) of the voltage iteration, as well as the step size, which will stay constant for each iteration.

Additionally, there is an option to continue the calculation if the starting (maximum) voltage load flow does not converge. *PowerFactory* then reduces the starting voltage until the load flow converges.

#### 41.3.1.3 Active Power Injection

By default the injection of active power at the study busbar(s) is set to zero. In the *Active Power Injection* page it is possible to define a list or a range of additional active power injections on the busbar(s).

If the *User defined range* option is selected, a minimum, maximum and step size for the active power injection can be defined.

If the *User defined list* option is selected, a list of active power values can be defined. Additional rows can be added by right-clicking and selecting *Append Row*.

For each set point a QV-Curve will be calculated.

#### 41.3.1.4 Output

Normally the critical voltage and the amount of reactive power injected at the study busbar(s) are printed in the output window; however additional output options can be printed in the output window, in case the user is interested to see the iteration process. The following options are available in the *Output* page of the QV Curves Calculation command:

- **Voltage iterations:** the reactive power and voltage values for each voltage iteration are printed.
- **Messages of initial Load Flow of each curve:** self explanatory.
- **Deactivated voltage controllers:** since the QV calculation controls the voltage at the selected busbar, all the elements that might affect the voltage control (e.g. station controllers, tap changers, voltage control of generators, etc.) are set to the base case value, for example, for a synchronous machine with voltage control, the controller mode will be changed from const.V to const.Q, fixing the dispatched reactive power value to the one obtained from the base case. Activating this option, all the disabled controller devices are printed in the output window.

#### 41.3.2 QV Curves Plot

By default, the QV Curves Plot command plots the critical busbar of the critical contingency. If the calculation has been made with no consideration of contingencies, the base case will automatically be selected. The QV Curves Plot options are explained below.

##### 41.3.2.1 Basic Options

**Data Input:** the results file specified in, and created by, the QV Curves Calculation command (see Section 41.3.1).

**QV-Curves:** the following options are available:

- Only plot critical case: this is the default option, it plots the critical contingency and busbar per active power injection (if more than one active power injection is calculated).
- User defined selection: the user can select the contingencies and busbars to be plotted.
- Critical node for selected contingencies: only the critical busbars for the selected contingencies are plotted. When this option is selected it is also possible to plot as many plots as contingencies.

##### 41.3.2.2 Active Power Injections

If several active power injections are calculated when executing the QV Curves Calculation command (see Section 41.3.1.3), it is possible to select between plotting all the active power injections or select only some cases.

### 41.3.2.3 Capacitors

In this page it is possible to define additional capacitors in every plotted QV curve. These additional capacitors are useful to determine the behaviour of the system when compensated.

## 41.4 Power Transfer Distribution Factors (PTDF)

Power Transfer Distribution Factors (PTDFs) are used to evaluate the change in power flow across a set of branches, while changing production and consumption, respectively, in two predefined regions. Knowledge of these factors assists greatly when analysing the impact of commercial transactions on cross-border transmission capacities. The PTDF command is available under the *Transmission Network Tools*, and the following sections explain how to use it.

Since *PowerFactory 2020*, the earlier Load Flow sensitivities function has been extended and additional distribution factor calculations developed. These include a more flexible and potentially faster PTDF calculation. For details, please see Chapter 50.

### 41.4.1 Calculation Options

#### 41.4.1.1 Basic Data

##### Calculation Method

Selection of either *AC load flow, balanced* or *DC load flow (linear)*. The *Load Flow* button provides access to the Load Flow command settings.

##### Region data

The Region Data defines the exporting and importing regions. The interchange region can be a zone (*ElmZone*), grid (*ElmNet*), area (*ElmArea*) or boundary (*ElmBoundary*). The selection here should be kept consistent; i.e. if areas are selected as 'Exporting regions', then areas should also be selected as 'Importing regions'.

If more than one element is selected, a PTDF set is automatically created. If a PTDF - Export or Import Set is available, then the user can select it by using the Select Set option.

##### Flowgates

The Flowgates are the interconnections between the exporting and importing regions; these can either be automatically determined according to adjacent regions, or can be user-defined by selecting one or more boundaries.

##### Scaling elements

These are the elements that are going to be scaled, and a selection of generators or loads is available. It is possible to either use all the elements available in the regions or select a *User defined* set of 'Scaling elements'.

- *Generators*: all synchronous machines (*ElmSym*) and Static Generators (*ElmGenstat*) that are in-service and connected are considered.
- *Loads*: all loads (*ElmLod*) that are in-service and connected are considered.

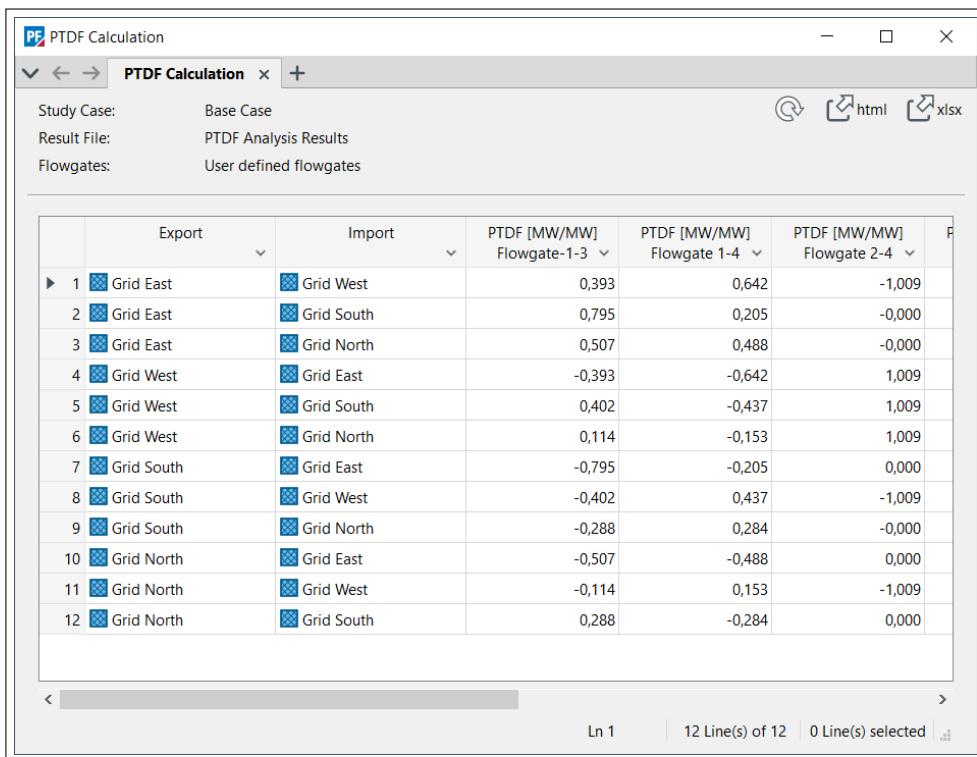
### Generators scaling mode

If generators are selected as scaling elements, the user can select between different scaling modes:

- *Rated active power*: generation shift is proportionally scaled for the generators according to their rated active power.
- *Remaining active power*: generation shift is scaled for the generators according to the “remaining active power”. The definition of “remaining active power” depends on the location of the generators: for generators in the exporting region, the value equals the difference between the maximum active power and operating active power; for generators in the importing region, the value equals the difference between the operating active power and minimum active power.
- *Merit-order table*: generation shift is scaled for the generators according to the merit-order table. When the *Merit-order table* option is selected, it is possible to define the power transfer between the two regions.

#### 41.4.1.2 Output of Results

The result of this calculation is the increase or decrease of power flow through the Flowgates according to the change of power in both the exporting and importing regions. All possible combinations of exporting and importing regions are calculated. An example result overview can be seen in Figure 41.4.1. In this example, the regions are different grids and the Flowgates are user-defined boundaries. In the tabular report each row represents the transaction between the exporting and importing regions, and the PTDFs are provided in the corresponding cell of the Flowgate column.



The screenshot shows a software window titled "PTDF Calculation". The top bar includes standard window controls and tabs for "PTDF Calculation" and "+". Below the tabs, there are three status fields: "Study Case: Base Case", "Result File: PTDF Analysis Results", and "Flowgates: User defined flowgates". To the right of these fields are icons for "html" and "xlsx" file formats. The main area is a table with the following data:

	Export	Import	PTDF [MW/MW] Flowgate-1-3	PTDF [MW/MW] Flowgate 1-4	PTDF [MW/MW] Flowgate 2-4	P
► 1	Grid East	Grid West	0,393	0,642	-1,009	
2	Grid East	Grid South	0,795	0,205	-0,000	
3	Grid East	Grid North	0,507	0,488	-0,000	
4	Grid West	Grid East	-0,393	-0,642	1,009	
5	Grid West	Grid South	0,402	-0,437	1,009	
6	Grid West	Grid North	0,114	-0,153	1,009	
7	Grid South	Grid East	-0,795	-0,205	0,000	
8	Grid South	Grid West	-0,402	0,437	-1,009	
9	Grid South	Grid North	-0,288	0,284	-0,000	
10	Grid North	Grid East	-0,507	-0,488	0,000	
11	Grid North	Grid West	-0,114	0,153	-1,009	
12	Grid North	Grid South	0,288	-0,284	0,000	

At the bottom of the table, there are navigation arrows and text indicating "Ln 1", "12 Line(s) of 12", "0 Line(s) selected", and a zoom icon.

Figure 41.4.1: PTDF result table

The results can be output in either ASCII or tabular format, selected on the Output page. There is a link to the actual reporting command, which is the same reporting command used for the more general Sensitivities / Distribution Factors calculation (see Chapter 50).

#### 41.4.1.3 Advanced Page

On the *Advanced* page, the user can define a set of scenarios (*IntScenario*) to calculate the PTDFs. If no set is provided, PTDFs are only calculated for the current operation point.

## 41.5 Transfer Capacity Analysis

The Transfer Capacity Analysis determines the maximum power transfer capacity between two regions, or between two sets of generators and/or loads, by scaling generation or load until a limit is reached.

If the region-based option is used, an importing and an exporting region are defined, and the capacity is calculated either for the lines connecting the regions or across a user-defined boundary.

If the element-based option is used, the user has to specify a boundary for which the capacity will be calculated.

### 41.5.1 Basic Data

#### 41.5.1.1 General

##### Calculation Method

Selection of either *AC load flow, balanced* or *DC load flow (linear)*. The *Load Flow* button provides access to the Load Flow command settings.

##### Contingency constrained

The analysis is carried out considering the contingencies and settings defined in the Contingency Analysis command. This command is accessible by clicking on the *Edit* arrow (→).

##### Transfer capacity definition

Here, the user specifies how the elements for scaling are selected. The choice here determines which other options are offered on this page.

- *Region Based*: The user will specify one importing and one exporting region, which can be grids or grouping objects such as Areas.
- *Element selection*: The user will provide two sets of elements, Exporting and Importing. The sets must be mutually exclusive.

##### Measured transfer capacity - for Region-based calculation

If the transfer capacity is calculated using Regions, there are two options for where the final transfer should be measured.

- *Interconnection lines*: In this case the automatically-determined interconnection lines are used as the interconnection border across which the transfer is measured and reported as the capacity.
- *Boundary*: This option can be selected if the user is in fact interested in the transfer across a different border, which will be specified as a Boundary (*ElmBoundary*).

##### Interconnection regions - for Region-based calculation

Two interconnection regions are selected; an interconnecting region can be a zone (*ElmZone*), grid (*ElmNet*), area (*ElmAra*) or boundary (*ElmBoundary*). The regions must be adjacent, i.e. have

interconnecting lines, but must not overlap. The interconnecting lines are determined automatically and can be shown via the button **Show interconnection lines**.

#### 41.5.1.2 Scaling elements

In region-based calculation, the user may select between generators, loads or both. By default, all relevant elements are scaled, but there are options for restricting the scaling to user-defined sets of elements.

- *Generators*: all synchronous machines (*ElmSym*) and static generators (*ElmGenstat*) that are in-service and connected are considered.
- *Loads*: all loads (*ElmLod*) that are in-service and connected are considered. The load shift is proportionally scaled for the loads according to their active power at the operating point.
- *Both generators and loads*: all the generators and loads in-service and connected are scaled.

#### User defined elements - for Element-based calculation

Here, the user specifies the elements that will form the Exporting and Importing sets. Using the down-arrow, the user can select elements or select a previously-defined set of elements. Generation or loads, or a combination can be used, but the two sets must be mutually exclusive.

A boundary (*ElmBoundary*), across which the transfer is to be measured, is also selected.

### 41.5.2 Constraints

#### 41.5.2.1 Thermal Limits

If the option *Consider thermal constraints* is selected, the thermal constraints for branch elements are considered. The available options are:

- Global constraint for all components: the limit defined in the *Maximum thermal loading of components* field is used for all branch elements.
- Individual constraint per component: the limit defined on the *Load Flow* page of every branch element is used.

Additionally, on the *Advanced* tab, a set of elements to consider can be defined.

The flag *Continue calculation if base case is overloaded* allows execution of the Transfer Capacity Analysis even if relevant constraints are not complied within the base calculation. If this option is selected, the user must specify a relaxation factor to be applied to the limits for those elements which are overloaded in the base calculation.

#### 41.5.2.2 Voltage Limits

If the option *Consider voltage limits* is selected, the voltage limits of all the terminals are considered for the Transfer Capacity analysis. The available options are:

- Global constraint for all terminals: the limits defined in the *Upper and Lower limit of allowed voltage* fields are used for all terminals.
- Individual constraint per terminal: the limits defined on the *Load Flow* page of the terminals are used.

Additionally, on the *Advanced* tab, a set of elements to consider can be defined.

The flag *Continue calculation if base case has voltage violations* allows execution of the Transfer Capacity Analysis even if relevant constraints are not complied with in the base calculation. If this option is selected, the user must specify a relaxation factor to be applied to the limits for those elements where limits are violated in the base calculation.

#### 41.5.2.3 Power Flow Limits

If one or several boundaries are selected as interconnection, this boundary can also be used as a limit. On the *Power Flow Limits* page the minimum and maximum power flow across a boundaries is defined via limits. The limit value can either be set globally or individually per boundary. If the option “individual constraint per boundary” is selected, the active power limits of the boundary is defined within the boundary on the *Transfer Capacity Analysis* page.

#### 41.5.2.4 Advanced

- *Consider constraints only for selected objects*: with this option, a specific selection, for which constraints will be applied, can be defined.
- *Ignore all constraints for nominal voltage below or equal to*: this option can be used to ignore the constraints on the terminals under a certain voltage level.
- *Consider active power limits*: the active power limits of the generators are considered in the analysis. The limit used is the minimum of the *Operational limit* and the *Rating limit* defined on the *Load Flow* page of the generator.

### 41.5.3 Output

When the Transfer Capacity Analysis is executed, the operational point of the network is modified in order to get a better solution. These modifications are not saved by default. However, the user can choose to save the data for the last feasible solution by using the following options available on the *Output* page:

- Do not save
- Save to a new operation scenario
- Save to a user defined operation scenario

If one of the options that creates a new operation scenario is selected, additional information about the created scenario (e.g. Total transfer capacity for the last feasible solution) will be stored on the *Description* page of the corresponding scenario.

The results of the Transfer Capacity Analysis are written in the output window as shown in Figure 41.5.1. In the example shown, the scaling mode was selected for all generators. *PowerFactory* generates a status report for each iteration while the power scaling process is running.

Transfer capacity analysis started...					
Generator SGI with active power set to zero is treated as the synchronous compensator and cannot be scaled.					
Generator dispatch status for generator SG2 is set to fixed. It will be ignored.					
...from Grid North to Grid West.					
Consider active power limits is enabled.					
Iteration Number	Exporting Region Generation (MW)	Importing Region Generation (MW)	Transfer (MW)	Iteration Status	
0	3741,00	1900,77	921,28	OK	
1	3787,06	1854,71	968,73	OK	
2	3879,19	1762,58	1064,12	OK	
3	3971,77	1670,00	1160,39	thermal violation, generation minimum reached (import)	
4	3971,32	1670,45	1159,92	thermal violation	
5	3925,26	1716,51	1111,98	thermal violation	
6	3902,22	1739,55	1088,04	OK	
7	3913,74	1728,03	1100,00	OK	
8	3919,50	1722,27	1105,99	thermal violation	
9	3916,62	1725,15	1103,00	thermal violation	
10	3915,18	1726,59	1101,50	OK	
Generation in the Grid West region reached limiting value. Calculation process stopped.					
Minimum step size reached. Calculation process stopped.					
The total transfer capacity (ITC) for the last feasible solution is 1101,50 (MW).					
Transfer capacity analysis successfully calculated.					

Figure 41.5.1: Transfer Capacity Analysis output

#### 41.5.4 Iteration Control

In the *Initial Conditions* the *Initial scaling factor* and the *Initial step size* for the analysis are defined. In the *Convergence Criteria* the *Min. step size* and the *Max. number of iterations* are defined.

These factors can be used to speed up the calculation by starting closer to the limit of the transfer capacity, and to increase the accuracy of the solution.

The *initial scaling factor* is used to define the starting point for the transfer capacity analysis. The procedure is as follows:

- A load flow calculation is executed and the actual power flow between regions or across the border is multiplied by the *initial scaling factor*,
- the selected loads and generators are then scaled accordingly,
- this dispatch is the starting point for the iterative calculation of the maximum power transfer capacity.

In case the actual power flow is smaller than 1 kW, the dispatch of the exporting region is scaled, to get the initial transferred power.

The *initial step size* is applied to the initial dispatch to calculate the first step. The *Min. step size* defines the convergence step of the algorithm and therefore the accuracy.

#### 41.5.5 Advanced

##### Use fictitious border network

This option is by default not selected and for most networks is not needed. It is provided for use with networks where adjacent regions, normally modelling adjacent countries, are not connected directly but via a so-called “fictitious border grid”, as shown in Figure 41.5.2 below. In order that the calculation recognises such configurations as interconnections, the option *Use fictitious border network* should be checked, and the border grid selected.

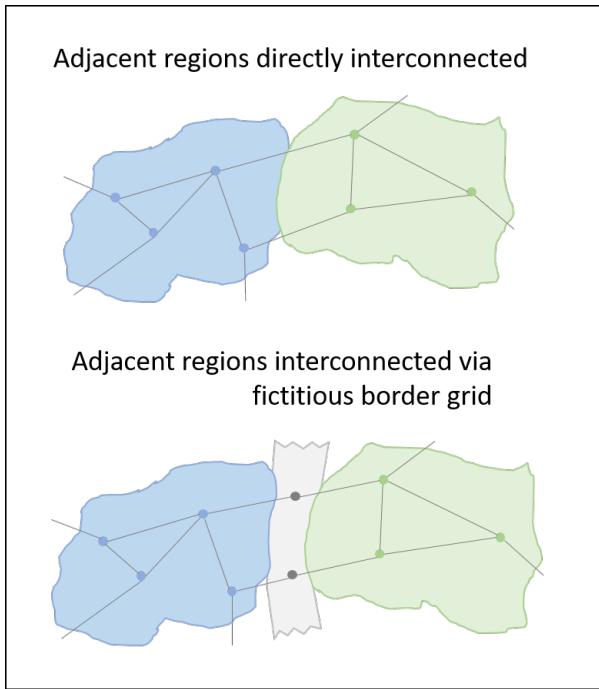


Figure 41.5.2: Network regions without and with fictitious border network

### Generators scaling mode

If generators are selected as scaling elements, the user can select between different scaling modes:

- *Rated active power*: generation shift is proportionally scaled for the generators according to their rated active power.
- *Remaining active power*: generation shift is scaled for the generators according to the “remaining active power”. The definition of “remaining active power” depends on the location of the generators: for generators in the exporting region, the value equals the difference between the maximum active power and operating active power; for generators in the importing region, the value equals the difference between the operating active power and minimum active power.
- *Merit-order table*: generation shift is scaled for the generators according to the merit-order table.
- *Generation shift key*: this scaling mode allows the assignment of individual participation factors for each power generating unit. Via their specification, it is possible to define how the regional change in power generation has to be mapped to each of the generating units within the exporting and the importing region.

The default set value of this attribute is 100% and it can be entered on the *Advanced* tab on the *Load Flow* page.

The actual shift factors of the generators that assume the power transfer changes between the exporting and importing regions, i.e. those set as “Dispatchable” on the field *Generator Dispatch* (*Automatic Dispatch* tab), are generally calculated as follows:

$$GSK_{i,act} = \frac{GSK_{i,ini}}{\sum_{i=1}^n GSK_{i,ini}} \quad (41.1)$$

## 41.6 Flow Decomposition

Power flows on a branch located somewhere in the power system can be classified by the locations of the sources and sinks causing these power flows. The flow decomposition gives the possibility to trace an active power flow on a branch element within the system, by analysing the sources (e.g. generators) and the sinks (e.g. loads) causing this flow. The branch element can be either an internal or a cross-border element connecting zones, areas or grids. Using this classification and a regional grouping, e.g. zones, following flow types can be defined:

**Loop Flow:** A loop flow is a flow where the source and sink are in the same zone but the branch element or even part of it is in another zone.

**Transit Flow:** A transit flow is a flow where the source, sink and the branch element or even part of it are all located in different zones, e.g. an active power flow from a source in zone A to a sink in zone C on a line in zone B.

**Export Flow:** An export flow is a flow on a branch element located in the same zone as the source with a sink in another zone. This flow is just defined for internal elements.

**Import Flow:** An import flow is a flow where the sink is in the same zone as a part of the branch element and the source is in another. Import/export flows on cross-border lines are always classified as import flows.

**Internal Flow:** An internal flow is an active power flow where the source, the sink and the branch element are located in the same zonal allocation.

The different flows are also visualised in Figure 41.6.1. It is important to note that the flow definition depends on the location of the branch element, e.g. a loop flow becomes an internal flow when it reaches the zone of the source and the sink.

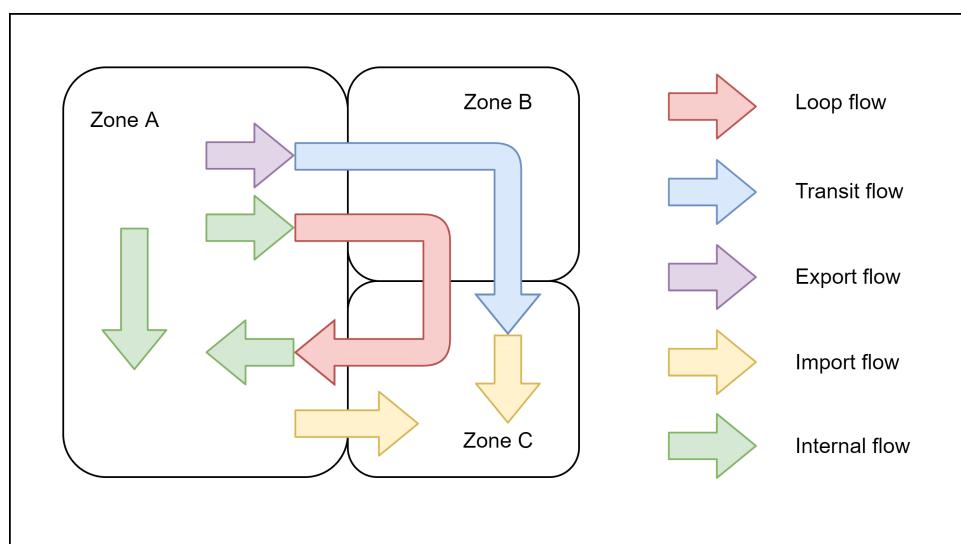


Figure 41.6.1: Definition of flow types according to zone

The flow decomposition is important for network planning to ensure sufficient free line capacity for the market or to analyse potential measures to reduce e.g. loop flows.

It is also used for the allocation of loop and transmission flows for the cost sharing between different system operators.

### 41.6.1 Technical Background

Tracing electricity can be seen as a transportation problem of determining how the power injected by generators is distributed between the lines and loads of the meshed network. The goal is to trace electricity from a particular generator to a particular supplier. There are different calculation methods that can be used for the flow decomposition. In *PowerFactory* the flow decomposition is performed according to the *Full Line Decomposition* (FLD) method [40]. The FLD method is based on the calculation of a power exchange matrix as described in detail by Bialek [40]. Further information can be found in [54].

The used FLD method is composed of the following main steps:

1. Calculation of a DC load flow.
2. Obtain nodal power transfer distribution factor matrix (PTDF matrix). The nodal PTDF matrix describes the linear relation between the power injections by generators and the active power flows in the network elements.
3. Calculate power exchange matrix (PEX matrix). The PEX matrix contains the power that is exchanged between each generator node and each load node. That means  $PEX_{ij}$  is the power produced in node  $i$  for the load in node  $j$ .
4. With the PTDF and the PEX matrix, the power flow partitioning matrix (PFP matrix) is calculated.
5. In the last step the flow types are calculated by filtering and summing up the cells of the PFP matrix.

The *Flow Decomposition* in *PowerFactory* supports the decomposition of the active power flow for DC load flow calculation.

#### 41.6.1.1 HVDC cycle flows

The HVDC cycle flows are calculated according to the method presented in [47]. Only HVDC-links with two connections can be considered. The flow decomposition can neither be used for Multi-Terminal HVDCs nor for HVDCs with generators or loads within the DC network. For all HVDCs in the system the condition  $P_{AC\ inflow} = P_{AC\ outflow}$  needs to be satisfied.

The HVDC-link is analysed regarding the effect on the AC network. The flow on the DC line itself is not decomposed.

#### 41.6.1.2 PST cycle flows

For the calculation of the PTDF matrix, non-linear phase-depending impedances of the PST need to be linearised. The linearisation is performed around the current operating point.

This may lead to a mismatch between the decomposed flows and the actual active power flow because of linearisation errors.

#### 41.6.1.3 Power Exchange Analysis

The *Power Exchange Analysis* gives information about which generator nodes and load nodes are responsible for a specific flow, e.g. which generator-load pair is causing the whole or parts of the loop flow. For this the PFP matrix is analysed to get all relevant elements causing this flow. Participations below a user-definable threshold are neglected.

## 41.6.2 Flow Decomposition Command

### 41.6.2.1 Basic Options

#### Method

The *Flow Decomposition* command offers two different calculation methods:

- **Flow Decomposition:** The *Flow Decomposition* decomposes the power flow on the selected network components.
- **Power Exchange Analysis:** In addition to the *Flow Decomposition*, the *Power Exchange Analysis* also calculates the corresponding sources and sinks for each flow types. Thus, not only the different flow types on a line can be analysed, but also the cause for each specific flow. The information can be recorded (see Section 41.6.2.3) and represented either for each element or grouping wise (see Section 41.6.3.2).

#### Analysed network components

There are two different options to select the analysed network components:

- All cross-border branch elements
- User-defined branch elements

Only branch elements, e.g. lines, transformers or switches, can be analysed. In case of a user-defined selection either one specific element or a set (\*.SetSelect) can be selected.

#### Groupings

Groupings are important for the flow decomposition as they define the flow classification of the decomposed flow. There are three available options:

- Zones
- Areas
- Grids

In case “Zones” have been selected for the grouping and some elements do not have a zone defined, all elements without a zone will be treated as if they were in the same zone. The same applies to areas.

#### Calculation times for decomposition

There are two options available:

- **Current study time only:** Calculates the flow decomposition just for one point in time.
- **Calculation times of Quasi-Dynamic Simulation:** Calculates the flow decomposition for all points in time defined in the Quasi-Dynamic Simulation command.

In each case, the used calculation command is linked below the selection. Note that the Flow Decomposition only supports DC load flow calculation.

The Power Exchange Analysis can be performed for the *Current study time only*.

### 41.6.2.2 Algorithm

On the *Algorithm* page, thresholds for the calculation can be defined:

**Min. power sensitivity (PTDF):** Minimum effectiveness of a generator on a line to be used within the PTDF matrix.

**Min. transformer sensitivity (PSDF):** Minimum effectiveness of a phase shift transformer to be used within the PTDF matrix.

**Threshold for power exchange matrix (PEX):** Minimum value for exchange that is still considered.

For the load flow initialisation two options are available:

- Use last calculated load flow results if available
- Always recalculate load flow results

Multi-threading can be used to increase the performance of the flow decomposition.

### 41.6.2.3 Results/Output

#### Recorded results

In case of a *Power Exchange Analysis*, the additional option “Recorded results” is available on the *Results/Output* page. There are three different options for the record of results:

- **Grouping-based power exchange data only** just records the source and sink of each flow regarding the grouping. Therefore, not the source or sink itself is recorded but just e.g. the zone.
- **Node-based power exchange data only** records for each flow the corresponding source and sinks.
- **All power exchange data** (default) stores the grouping-based data as well as the node-based data within the result file.

#### Output

There are three options for the output messages available:

- Off
- Short
- Detailed

### 41.6.3 Flow Decomposition Tabular Report

For the representation of the results a specific report is available. The report can be created using the *Flow Decomposition Tabular Reports* command (✉). In case a Power Exchange Analysis was performed there is the option to choose between different reports. All reports can be exported as html or xlsx. The different reports are described below.

#### 41.6.3.1 Flow Decomposition Report

The Flow Decomposition Report lists the different flow types per component (e.g. line). The report contains the flow categories, as well as the total active power flow and the mismatch on the line. The

mismatch is the difference between the active power flow and the sum of all flow parts. The cause for a mismatch are non-linear tap depended impedances (see Section [41.6.1.2](#)).

#### 41.6.3.2 Power Exchange Analysis Report

The Power Exchange Analysis Report contains for the flow on each analysed component the participating nodes. In the header of the report, the analysed component can be selected and a minimum flow value can be set. It is also possible to show the power exchange analysis for all components.

- **Groupings:** The Power Exchange Analysis Report for Groupings sums up the participating generators and loads according to their grouping. This gives a good overview, especially in large networks, but takes away the possibility of analysing the specific elements that cause a particular flow.
- **Nodes:** In case the Power Exchange Analysis Report for Nodes is created, the node of the generation and the node of the load are displayed in addition to the related grouping. Especially in large networks, this can lead to long tables. In such cases, it is therefore advisable to set an appropriate minimum flow value.

The Power Exchange Analysis Report for Nodes presents all existing flow types in one column that can be filtered using the selector of the Flow Type column (▼).

Note that the report for groupings or nodes can only be displayed, if the corresponding option has been selected on the Results/Output page of the Flow Decomposition command (see Section [41.6.2.3](#)).

#### 41.6.4 Flow Decomposition Plots

To visualise the results of a Flow Decomposition over time a plotting function is available. To create a plot the *Create Flow Decomposition Plot* command (Plot) can be used. This function creates a plot visualising the change of the decomposed flows over time and is especially handy to visualise the decomposed flows of a Quasi-Dynamic Simulation.

More information about plots is available in chapter [Reporting and Visualising Results](#), Section [19.8.9](#).

# Chapter 42

## Distribution Network Tools

### 42.1 Introduction

This chapter presents the *PowerFactory* Distribution Network Tools, which are dedicated functions for the analysis and optimisation of distribution networks. Each section of this chapter introduces the tool and its command dialogs. For the optimisation tools, the objective function and the optimisation procedure are described, too.

The set of functions is organised into two sub-groups, namely Distribution Network Analysis and Distribution Network Optimisation. The toolbox for each subgroup is accessible through the *Change Toolbox* button on the Main Toolbar.

The toolbox “Distribution Network Analysis” is shown in Figure 42.1.1. It contains the following functions:

- *Hosting Capacity Analysis*, described in Section 42.2
- *Backbone Calculation*, described in Section 42.3
- *Voltage Sag Table Assessment*, described in Section 42.4
- *Low Voltage Load Flow Calculation*, described in Section 42.5

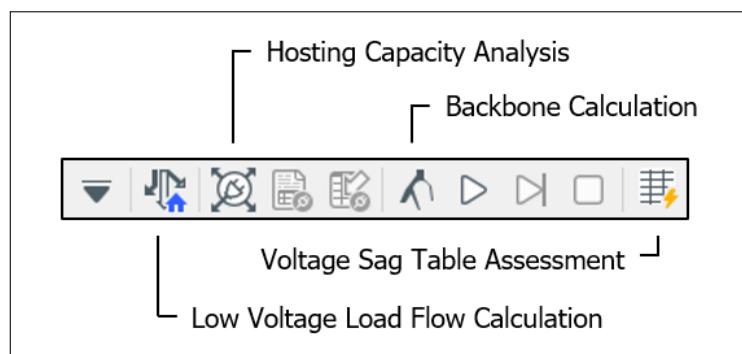


Figure 42.1.1: Distribution Network Analysis Toolbox

The Toolbox “Distribution Network Optimisation” is shown in Figure 42.1.2. It contains the following functions:

- *Tie Open Point Optimisation*, described in Section 42.6
- *Phase Balance Optimisation*, described in Section 42.7

- *Voltage Profile Optimisation*, described in Section 42.8
- *Optimal Equipment Placement*, described in Section 42.9
- *Optimal Capacitor Placement*, described in Section 42.10

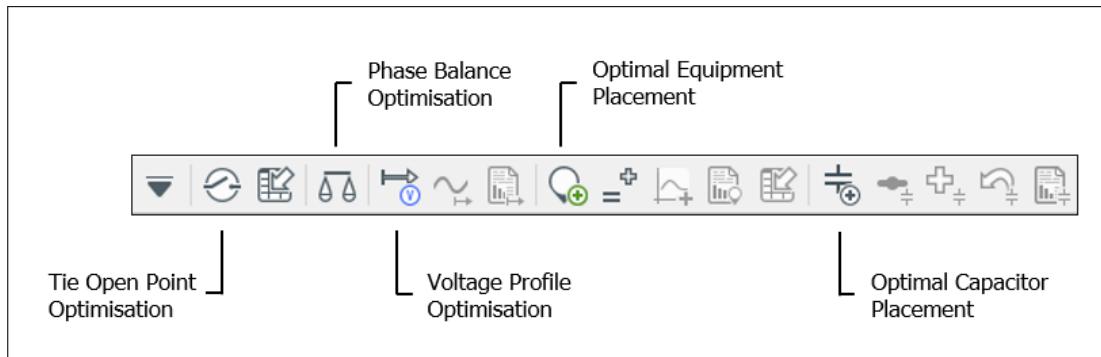


Figure 42.1.2: Distribution Network Optimisation Toolbox

Additionally, the Section 42.11 provides a description of the stochastic optimisation algorithms used.

## 42.2 Hosting Capacity

The Hosting Capacity tool allows the analysis of the impact of adding generation (Distributed Energy Resource (DER) Capacity) or load (Spare Load Capacity) to a power system.

Hosting capacity is generally defined as the amount of new generation or consumption that can be connected to the grid without violation of system constraints (e.g. power quality for connected customers) and without any network expansion. All constraints used to determine the level of possible generation or load accommodation can be quantified using the performance index. This index can correlate with voltage violation, loading violation, protection setup etc. and can be interpreted as the system robustness. Therefore, the hosting capacity can also be defined as the amount of new generation or consumption where the performance index reaches its limit. This is illustrated in the following figure:

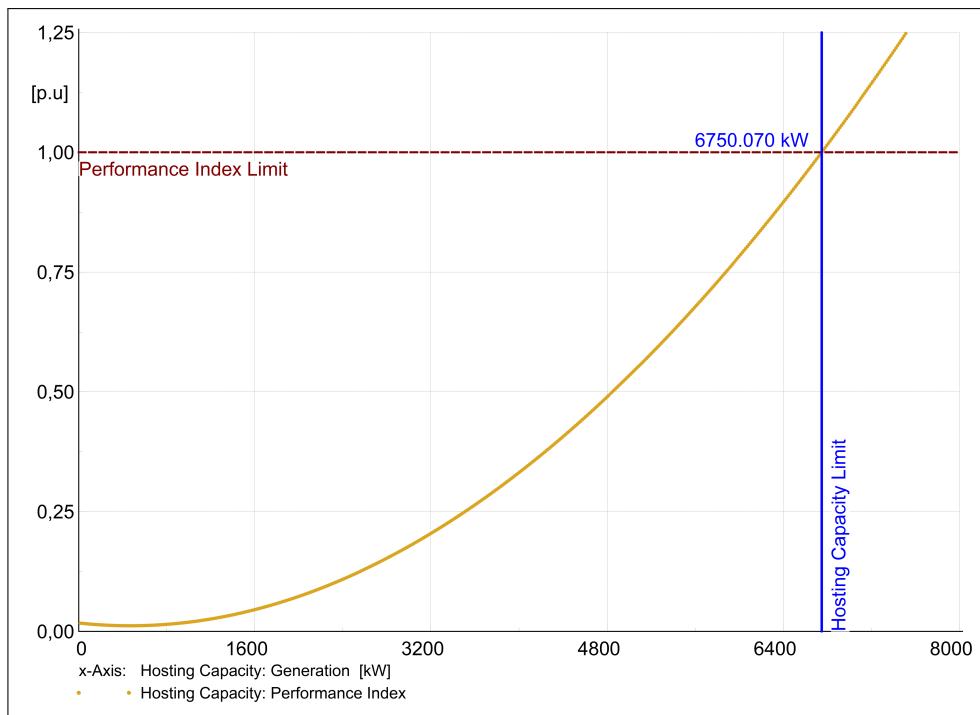


Figure 42.2.1: Performance Index limiting the max. amount of generation/load

### 42.2.1 Technical Background

In this section the process of the Hosting Capacity Analysis, depending on the command settings, is described. It also contains references to the below sections describing the available settings (like calculation objective, constraints, etc.), in order to quickly find the corresponding settings in the command.

As described in Section 42.2.2, first the hosting sites to be analysed have to be selected. From the selection, the corresponding terminals (internal nodes are ignored) are collected, for which the maximum hosting capacity or spare load capacity is calculated. The calculation objective (see Section 42.2.2.1) determines whether generation or load is scaled:

- When option “Distributed Energy Resource (DER) Capacity” is selected each node is processed individually. A virtual generator (ElmGenstat) is internally connected to the first terminal from the hosting sites selection. Beginning from the initial power value (see Sections 42.2.2.3 and 42.2.2.4), the power of this generator is scaled until one of the user-defined constraints is violated. Then, the analysis continues with the next node until all hosting sites are processed.
- If the option “Spare Load Capacity” is selected, per default all loads already connected to the terminals to be analysed are scaled simultaneously. The maximum spare load capacity is obtained when one of the user-defined constraints is violated. Another option (see Section 42.2.2.1, Advanced tab) allows each load to be analysed separately. Here, as well, only terminals to which a load is already connected are processed. On the “Configuration” page (see Section 42.2.2.3) however the user can activate the option “Connect new load”. In this case, the same algorithm as for DER Capacity is used to increase the load at the terminals one after the other.

The algorithm used to obtain the maximum hosting or spare load capacity is based on the binomial search method. It starts from an initial active power value. The power is then increased by the initial step size defined on the *Iteration Control* page (see Section 42.2.2.4).

A load flow calculation is executed and the constraints defined by the user on the *Constraints* page (see Section 42.2.2.2) are checked for limit violations. If no violation has occurred, the algorithm continues with the double of the previous step size and checks again for violations. This process is kept up

until a limit violation occurs. In this case, the half of the previous step size is used. This process is continued until the algorithm converges to the correct solution without violations or the maximum number of iterations is reached.

## 42.2.2 Hosting Capacity Analysis Configuration

The Hosting Capacity analysis can be opened by pressing *Hosting Capacity Analysis* button  in the Distribution Network Analysis Toolbox.

More information about the available settings can be found in the following subsections.

### 42.2.2.1 Basic Options Page

#### Hosting sites

In order to carry out the Hosting Capacity Analysis, so-called hosting sites must be specified. These are the network nodes at which the generation or the demand is increased until a system constraint is reached, allowing to assess the robustness of the network under analysis.

The hosting sites can be specified

- From the network graphic, Network Model Manager as well as Data Manager; by right-clicking on the selection and choosing *Calculation* → *Hosting Capacity Analysis...* from the context menu.
- Via the Hosting Capacity Analysis command, by clicking on the button (▼) next to *Hosting sites* and choosing the elements from the selection browser that appear after pressing the option *Select....*

A multiple selection of elements is possible in both methods, with a Set (*SetSelect*) being created automatically. The set can be edited by clicking on the button (→).

The network elements of the following classes can be assigned to the Hosting sites field:

- Terminal (*ElmTerm*)
- Substation (*ElmSubstat*)
- Secondary Substation (*ElmTrfstat*)
- Feeder (*ElmFeeder*)
- Area (*ElmAra*)
- Zone (*ElmZone*)
- Grid (*ElmNet*)

If a grouping object is selected, the calculation will extract the relevant terminals.

#### Calculation objective

1. Distributed Energy Resources (DER) Capacity: assessment of maximum connectible DER hosting capacity for each terminal within the selection.
2. Spare Load Capacity: assessment of maximum connectible Load Capacity by terminal or overall by increase the power demand of all loads simultaneously.

## Calculation method

The following methods are supported and can be used in the calculation:

- AC Load Flow, balanced, positive sequence
- AC Load Flow, unbalanced, 3-phase (ABC)
- DC Load Flow (linear)

The corresponding load flow command which is used in the calculation is linked ( ) under *Base case load flow settings*. More information on the load flow settings can be found in Chapter [25.3](#).

## Load Flow type

Select the type of Load Flow that shall be used:

- Standard Load Flow (see Chapter [25](#))
- Low Voltage Load Flow (see Section [42.5](#))

## Constraints

In this command section, the user has the possibility to specify which constraints have to be considered during the calculation. Following options are available:

- Thermal limits
- Voltage limits
- Protection limits
- Power quality limits
- Short-circuit contribution limits

The last two constraint options are only available for selection if DER Capacity is selected as calculation objective. Only asserted constraints can be configured in detail on the *Constraints* page as described in Chapter [42.2.2.2](#).

## Advanced

The options on the *Advanced* page are only available when the calculation objective *Spare Load Capacity* is selected.

### Consideration of Loads

For the consideration of loads, the user has two options which are further described in Chapter [42.2.2.3](#):

- *All loads simultaneously*: The possible load growth can be determined by scaling all loads in the network simultaneously. This selection is only possible if the Standard Load Flow has been selected as Load Flow type.
- *Each load separately*: Loads are treated independently. This means that the power demand of each load is increased separately until one of the set limits is reached.

### Load scaling factor

If *All loads simultaneously* is selected as method for the consideration of loads, two options for the usage of a Load scaling factor are available:

- *Equal distribution*: The same scaling factor will be applied to all loads, depending only on the actual step size (step). With this approach, the equation  $P_i = P_{i \text{ ini}} \cdot (1 + step)$  is used for each load  $i$ .

- *Weighted scaling factor:* The scaling factor that is applied to a load depends not only on the actual step but also on its initial power. The larger the initial power of a load  $i$ , the larger the scaling factor that is applied to it. This behaviour can be described with the equation  $P_i = P_{i \text{ ini}} \cdot (1 + \frac{P_{i \text{ ini}}}{P_{\sum}} \cdot \text{step})$ .

### 42.2.2.2 Constraints

The options available in the tabs of the page Constraints are described below.

#### General

- Ignore all constraints for nominal voltage below or above the given value.
- Constraints check
  - *Per feeder:* Only constraints related to network elements within the investigated feeders will be assessed. In addition, also the supplying substation, including the supply transformer and terminals, is assessed.
  - *Whole system:* Constraints are checked within the whole system.
  - *Specific elements:* Constraints are only checked for a pre-defined *Selection* of elements.

#### Thermal Limits

Two option are available for configuring the thermal limit of branches:

- *Global constraint for all components:* If enabled, the user must enter a maximum thermal loading percentage in the *Maximum thermal loading of components* field. Note that this option overrides the individual component thermal limits.
- *Individual constraint per component:* If asserted, the thermal rating of each branch element is considered individually during the analysis. Thermal ratings of a branch components can be set on the field *Max. Loading* on the Load Flow page.

#### Voltage Limits for Terminals

The following options related to the voltage constraints at terminals are available:

- *Consider Voltage Limits:* If this option is enabled, then each terminal in the system is checked against the defined thresholds.

Being the option *Consider Voltage Limits* asserted, two alternatives for configuring the voltage limits become available:

- *Global constraints for all terminals:* If asserted, the limits set on the two fields available in the command (*Upper and Lower limit of allowed voltage*) are used globally to check the constraints. Note that this option overrides the individual terminal voltage limits.
- *Individual constraint per terminal:* If this option is selected, then the voltage of each terminal is checked against its specific limits, which can be set within the data frame *Steady State Voltage Limits* on the Load Flow page of the terminal edit dialog. Note that these local limits are only available if the terminal *Usage* is either Busbar or Junction node.

If the calculation method *AC Load Flow, unbalanced* is selected on the *Basic Options* page, either the line-ground or the line-line voltages can be selected for the *3-phase (ABC) voltage check*:

- *Line-ground, L-G*
- *Line-line, L-L*

## Voltage Limits for Feeders

Next, the options concerning voltage limits within Feeders are described.

- *Voltage drop/rise*: It allows to define constraints related to the voltage variations along feeders. Following sub-options are available:
  - *Global constraints for all feeders*: The global limits defined in the command are considered through the fields that are activated if the option is selected:
    - \* *Maximum Voltage Rise*
    - \* *Maximum Voltage Drop*
  - *Individual constraint per feeder*: If selected, the limits defined in each feeder element are considered. In feeders, these limits can be defined in the *Limits for Voltage Change relative to Feeding Point* field, accessible in the Load Flow page of the element's edit dialog.
- *Voltage regulator tap change*: If checked, allows you to define limits for the number of tap changes with following sub-options:
  - *Max. allowed number of taps (regulator)*
  - *Max. allowed number of taps (capacitors)*
- *Permissible voltage change (from no DER to full DER)*: The option allows you to define a limit for the maximum voltage variation (*Max. voltage change*) between the extreme scenarios. The following can be used to fine-tune the calculation performed for this analysis:
  - *Regulated load flow settings*: A separate load flow command can be used for checking the constraint (*Max. voltage change*). Its settings can be accessed through the button (→). You can use the button *Reset to Base Case Settings* to transfer all configurations from the load flow command set on the Basic Data page.
  - *Generators to be switched off*: This option can be found on the Advanced tab. The assessment is based on the selected generators:
    - \* *All static generators from the feeder*
    - \* *User-defined filter*
- *Voltage unbalance*: This option is only available for selection if unbalanced calculations are performed, i.e. if *AC Load Flow, unbalance, 3-phase (ABC)* is selected as calculation method. If the option is selected, the definition of a constraint for the *Maximum voltage unbalance factor* (parameter `c:maxUnbFacU`) becomes available.

---

**Note:** The settings

- *Voltage rise/drop*
- *Voltage regulator tap change*
- *Permissible voltage change (from no DER to full DER)*

only apply if the advanced option *Ignore terminals which are not part of a feeder* is selected (see Chapter 42.2.2.6) or the selection of *Hosting sites* only contains one or more feeders.

---

## Protection Limits:

The user can choose between the consideration of all relevant protection devices or a user defined set of protection devices. Please note that the Relay or fuse does not have to be modelled as a separate device within the network model, but can also be defined through the circuit breaker setting *Consider as switch with protection device*.

1. *Reversed power flow* option to prohibit the occurrence of reverse power flow during Hosting Capacity analysis.
  - Reversed power flow is analysed for switches where the setting *Consider in the reversed power flow analysis* is set.

2. *Relay/fuse tripping* option to set up relay or fuse tripping to be a hosting capacity limiting constraint.
3. Limit *Fuse current change* by the entered value in relation to base case Fuse current.
4. The *Total fault contribution* limits the maximum allowed fault contribution of one Generator in relation to the base case Short-Circuit results.
  - This constraint is only available for calculation objective *Distributed Energy Resource Capacity*.

#### **Power Quality:**

1. The percentage *Total Harmonic voltage distortion* can be selected, where the limit will be assessed based on the *Harmonic Load Flow* command (Chapter 36.2).
2. *Short-term flicker disturbance factor (continuous operation)* according to Harmonic Load Flow analysis can be defined.

#### **Advanced:**

1. Generators to be switched off
  - Choose between all static generators to be switched off or a user defined set of generators.
  - This set of generators will be used for assessment of *Permissible voltage change* within the *voltage limits* constraints settings.
2. Short circuit calculation
  - The Short Circuit calculation can be executed at all terminals within the feeder or only at the DER location.

#### **42.2.2.3 Configuration**

Depending on the *Calculation objective* selected on the Basic Options page (see Section 42.2.2.1), either the Distributed Energy Resource (DER) or the load to be connected can be configured. The phase technology of the connected element is modified according to the phase technology of the corresponding terminal.

**Connected DER:** Two options for the generator to be connected are available.

- *Use generator with settings below:* A Static Generator (ElmGenstat) is internally set up according to the available settings on this page and used in the analysis.
- *User-defined template:* In order to allow more flexibility a Static Generator template can be configured as required. It can be opened by clicking on the blue arrow →. Please note that options like the phase technology or the dispatch of the generator can be defined in the template but are overwritten by the Hosting Capacity command.

For more information about the Static Generator model, please refer to the corresponding technical reference.

The following settings define the generator which is connected to the terminals during the analysis. Some of them will only be available if *Use generator with settings below* is selected.

- *DER power factor:* The Power Factor used for the generator can be specified.
- *User-defined active power limits:* If activated, the initial and the final generated power can be defined. If deactivated, the *final generated power* is not limited. The initial power is automatically adapted to the voltage level, in order to accelerate the convergence process. The corresponding mapping table is shown in Table 42.2.1.
  - *Final generated power:* This value will limit the maximum active power that is connected to a terminal, if the algorithm does not stop earlier because of a constraint violation.

- *Initial generated power*: This value is useful if the hosted power value is approximately known (this will accelerate the convergence process) or if the initial generated power defined in the internal mapping table shown above is too big (algorithm will find no solution).

---

**Note:** Please ensure that the initial power value is below any constraint violating power. Otherwise, the algorithm will not find a solution.

---

Voltage Level in kV	Initial Power
< 9.5	5 kW
< 39.5	200 kW
< 100	1 MW
≥ 100	10 MW

Table 42.2.1: Mapping Table: Nominal Voltage vs Initial Generated Power

- *Short-circuit contribution*: This field will only be available if *Short-circuit contribution limits* option is enabled on the *Basic Options* page (see Section 42.2.2.1). More information about the effect of the parameters *R/X*" and *Short-circuit model* are available in the technical reference of the Static Generator.
- *Harmonics contribution*: This field will only be available if *Power quality limits* option is enabled on the *Basic Options* page (see Section 42.2.2.1). Depending on the settings on the *Power Quality* tab of the *Constraints* page (see Section 42.2.2.2), the following two settings are available:
  - *Harmonic currents*: Available if *Total harmonic voltage distortion* option is activated. A *Harmonic Source* (*TypHmccur*) can be selected from a global or project library. It is assigned to the generator that is connected to the terminals during the analysis. The definition of the *Harmonic Source* is necessary to evaluate the effect of the increasing generation on the Total Harmonic Distortion (THD) in the network.
  - *Flicker coefficients*: Available if *Short-term flicker disturbance factor (continuous operation)* option is activated. A *Flicker Coefficient* (*TypFlicker*) can be selected from a global or project library. The object is assigned to the generator that is connected to the terminals during the analysis, in order to evaluate the Short-term flicker disturbance factor (continuous operation) with increasing generation.

**Connect new load:** This option is only available if calculation objective *Spare Load Capacity* is chosen and option *Each load separately* is enabled on the *Advanced* tab of the *Basic Options* page.

The Hosting Capacity algorithm will connect new loads depending on the following options:

- *Do not connect*: No new loads are connected, meaning that only existing loads are considered and scaled independently.
- *Only if no previously connected found*: New loads are only connected to terminals where no load element is already connected. Existing and new loads will be considered together and scaled independently.
- *Always connect*: A new load is connected to all available terminals in the network. Only new loads are considered and scaled independently.

---

**Note:** If the *Low Voltage Load Flow* is selected as Load Flow type on the *Basic Options* page, only *Always connect* can be selected.

---

If the connection of loads for the analysis is allowed, the load to be connected can be configured, depending on the *Load Flow type*.

### Load Configuration with Standard Load Flow

- *Initial load active power:* Active power value from which the algorithm starts
- *Power factor:* The Power factor used for the load can be specified

For the calculation of the *Spare Load Capacity*, the *Power Factor* of the loads can be configured in two ways:

- *Keep constant power factor:* The power factor of the loads will be constant, which means that both active and reactive power will be scaled equally.
- *Keep constant reactive power:* The reactive power of the loads will be constant, which means that only the active power will be scaled. The power factor changes accordingly.

### Load Configuration with Low Voltage Load Flow

- *Coincidence definition:* A coincidence definition for the new low-voltage load has to be selected (see also Section 42.5.2). If a fixed load should be assumed, it is suggested to create an additional coincidence definition with  $g_{\infty} = 1$ .

For the calculation of the *Spare Load Capacity*, there are two different ways how an increase in power can be configured in the *Consumption definition*:

- *Customer number increase:* The maximum power from the coincidence definition is used and only the number of customers in the new load is increased. If this configuration is selected, the *Initial customer count* for the start of the iteration can be entered.
- *Individual power increase:* Only one customer is assumed in the new load and the individual consumption of this load is increased, regardless of the maximum power in the coincidence definition.

#### 42.2.2.4 Iteration Control

On this page, settings affecting the iteration process and the convergence of the algorithm can be specified.

**Initial conditions:** In this field, the *initial step size* can be defined. The first step is then calculated by the initial power multiplied by the initial step size, e.g.  $P_{new} = P_{ini} \cdot (1 + step_{ini}/100\%) = 1MW \cdot (1 + 1\%/100\%) = 1.01MW$ .

**Convergence criteria:** *Min. step size* defines the smallest step size of the algorithm before it stops. A value of 1% means a step size of 0.01 MW. The algorithm will also stop if the *Max. number of iterations* is reached.

#### 42.2.2.5 Output

**Results:** Link to the result object (*ElmRes*) for the Hosting Capacity Analysis. A result object can be selected by clicking on and then *Select...*. An already selected one can be edited by pressing the *Edit* button .

---

**Note:** Depending on which calculation objective (*Distributed Energy Resource (DER) Capacity* or *Spare Load Capacity*) is selected, a different result object (*ElmRes*) is automatically created and assigned to the command.

---

Additional options are available for the *Output per iteration* to select the level of detail that is written into the output window.

#### 42.2.2.6 Advanced

##### Ignore terminals which are not part of a feeder:

- **Disabled:** If a set of terminals is selected, the Hosting Capacity command does not automatically search for the corresponding feeders. All nodes are processed individually.
- If activated, only terminals which are part of a feeder are processed. Furthermore, the algorithm analyses the terminals feeder by feeder. It also affects the Spare Load Capacity option *All loads simultaneously* (see Section 42.2.2.1). If this option is activated, all loads within a feeder are scaled simultaneously.

**Include feeder starting terminal:** A feeder which is defined in the Branch direction does not automatically include the feeder starting terminal. If this terminal should be considered in the Hosting Capacity Analysis, this option can be activated.

**Ignore time trigger:** By activating this option, all time triggers will be ignored in the analysis. This can be used to avoid a time dependency of the results.

**Initialisation of load flow:** To determine the hosting capacity of the system under investigation, the algorithm iteratively executes load flow calculations. For the initialisation of each load flow, it is possible to choose between the following options:

- *Based on previous load flow results:* A Load flow calculation uses the results of the previously executed calculation for the initialisation (no flat-start).
- *Full initialisation:* A flat-start is used as initialisation of each load flow.

#### 42.2.2.7 Parallel Computing

**Parallel computation:** By activating this option, a user-definable number of calculation objects (nodes) is distributed to different cores, for which the solution of the Hosting Capacity analysis under consideration of all selected constraints is then calculated and returned to the main process. Due to the parallel computation the performance is multiplied, resulting in a significant reduction in the calculation time.

- *Minimum number of calculation objects:* The number of calculation objects must exceed this number, in order that the analysis is executed in parallel.
- *Package size for parallel process:* Number of calculation objects/nodes that are transferred to each core per cycle.
- *Parallel computation settings:* Refer to Section 6.6.1 for more information.

### 42.2.3 Results of the Hosting Capacity

This section informs about the possibilities to evaluate the Hosting Capacity results. For the analysed terminals *Statistics:Calculation Parameter* are available that can be displayed in the Network Model or Data Manager or in the Result Boxes in the single line or geographical diagram.

**Result Boxes:**

By right-clicking on a result box of a terminal in the network graphic, a predefined format for the Hosting Capacity can be selected (*Format for Nodes → Hosted Power (P, Q)*), in order to display the active and reactive hosted power at the nodes.

Furthermore, different colouring schemes are available, in order to colour the schematic or geographical diagrams according to different parameters. Several tabular reports help the user to easily find for example the maximum hosted power or the limiting component. More information about the diagram colouring and the reports can be found in the following sections.

General information about configuring colours can be found in Section [4.7.8](#).

#### 42.2.3.1 Graphical Representation

**Hosting Capacity Circles:**

In geographic diagrams circles can be displayed around substations, sites, secondary substations and nodes, whose sizes represent the power that can be connected without violating the user-defined constraints. This option can be enabled/disabled by clicking on  and editing the layer *Load/generation distribution*. On the page *Load/Generation* the option to show the *Hosting capacity* circles can be activated or deactivated and the colour for the circles (*Circle colour*) can be defined (see Section [10.8](#) for more information). This option is available for the *Displayed variables* “P” and “Q”.

**Diagram Colouring Scheme:**

The colouring can be enabled by clicking on the *Diagram Colouring...* button  and opening the *Hosting Capacity* page. Activate *3. Others* and select *Results → Hosted Powers*. By clicking on Colour Settings..., the colouring mode can be determined.

Four colouring modes are available. For each one, the colour gradient can be defined in the alongside table called *Hosted Power*.

- *hosted power at nodes*
- *maximum feeder power*
- *minimum feeder power*
- *average feeder power*

---

**Note:** The feeder colouring is only possible if feeders have been analysed (i.e. feeder object(s) selected as *Hosting Site*).

---

**Substation colouring:** This option is only available if colouring mode *hosted power at nodes* is selected. Substations can then be coloured according to the

- *maximum power of connected terminals* or
- *minimum power of connected terminals*.

**Enable branch colouring:** This option is only available if colouring mode *hosted power at nodes* is selected. If enabled, branches can be coloured according to the

- *maximum power of two connected terminals* or
- *minimum power of two connected terminals*.

### 42.2.3.2 Tabular Reports

The results can also be evaluated by tabular report. These can be created by clicking on the *Hosting Capacity Reports* button  in the Distribution Network Analysis toolbar. A dialog will open, in which the tabular report types to be created as well as the constraints to report, can be selected.

**Results:** The result object (*ElmRes*) to be analysed can be selected by clicking on  and then *Select....*. A window showing all result objects contained in the active study case will open and the result object can be selected.

**Report tables:** The following types of tabular reports can be selected:

- *Feeders*: The report shows the maximum, minimum and average hosted active power within the analysed feeders. The tabular report will only display feeders that have been selected as Hosting Sites.
- *Terminals*: For each analysed terminal the maximum active and reactive hosted power that can be connected before a constraint is violated, the corresponding violated constraint (if selected, see *Constraints to report*) and the limiting component are displayed. The limiting component is the element at which a limit has first been violated.
- *Ignored terminals*: Lists all terminals that were selected but have been ignored in the Hosting Capacity analysis.

**Constraints to report:** By selecting the following constraints, columns will be added to the tabular report.

- *Thermal*
- *Voltage*
- *Protection*
- *Power quality* (only available for DER Capacity analysis)

#### Filters:

- *Results per phase*: If selected and an unbalanced analysis has been performed, columns will be added in the tabular report, displaying the connectable power per phase.
- *Show apparent power*: If selected, a column will be added to the tabular report, displaying the connectable apparent power.
- *Colouring threshold*: Threshold for the colouring of the limiting values. A value will be coloured in red, if its relative value based on the limit in percent exceeds the threshold.
  - *Loading Colouring Threshold*: Is used to colour loading values. For example: If the loading threshold is set to 99.8% and the loading limit is 100%, all loading values over this threshold will be coloured.
  - *Voltage Colouring Threshold*: Is used to colour voltage values. For example: If the voltage threshold is set to 1%, and the maximum and minimum voltage limits are 1.04 p.u. and 0.96 p.u. respectively, then all voltage values over 1.039 p.u. and under 0.961 p.u. will be coloured.
  - *THD Colouring Threshold*: Is used to colour THD values. For example: If the THD threshold is set to 10% and the THD limit is 5%, all THD values over 4.5% will be coloured.

**Used format:** Links to the different format templates for the tabular reports.

### 42.2.3.3 Load Hosting Capacity Results

By clicking on the *Load Hosting Capacity Analysis Results* button  in the Distribution Network Analysis toolbar, previous results can be reloaded.

In the dialog window, a Hosting Capacity result object has to be selected, by clicking on  and then *Select....*. From the selection browser the desired result object (*ElmRes*) can be selected. After confirming with **Execute**, the results are reloaded and can be evaluated in the network graphic, the Network Model Manager or Data Manager.

## 42.3 Backbone Calculation

This section describes the Backbone Calculation command (*ComBbone*) dialogs and presents an example calculation. To run a Backbone Calculation, either:

- Select the *Backbone Calculation* icon under *Distribution Network Analysis* as shown in Figure 42.1.1.
- From the *Data Manager*, select and then right-click previously defined feeders and click *Calculation → Backbone Calculation....*
- From the main menu, select *Calculation → Distribution Network Analysis→ Backbone Calculation....*

The Backbone Calculation is used to determine the main paths between adjacent feeders connected via open points, that may serve to restore lost load in case of failures inside a feeder. The command creates objects in the Network Data folder (*ElmBbone*) with the Backbones constituent network elements. This simplifies visualisation of the main path(s) between feeder(s), particularly in large distribution networks where the main paths may not be apparent from the single line diagram.

Backbone objects are created for all feeders or a user-defined set of feeders based on path load, cross-section, network structure, or scoring method criteria. The command can optionally consider existing remote controlled switches at open points, and the availability of connections to alternative transformers or substations when creating Backbones.

From the Backbone dialog, the *Backbone* contents (elements) can be viewed, marked in the graphic, and checked (see example in Section 42.3.4). The **Check Backbone** button is used to verify that the backbone object still defines a valid inter-feeder path matching its calculated parameters.

### 42.3.1 Basic Options Page

#### Generate backbones

Specify all feeders or a user-defined set of feeder/s for the Backbone Calculation.

#### **Calculation based on:**

**Note:** For all calculation methods, *feeder is supposed to be operated radially* must be selected on the Basic Options page of the relevant Feeder/s.

- Path load: Backbones are determined based on the MVA load on the paths between adjacent feeders.
  - (Optional) specify the max. number of backbones per feeder.
  - Optionally select to *Report results* to the output window, including details of backbone open points.

- Pointer to load-flow command (note for balanced calculations only).
- Cross section: Backbones are determined based on the cross-section of lines/cables connecting adjacent feeders.
  - (Optional) specify the max. number of backbones per feeder.
  - Choose to determine backbone using either the mean cross section of lines in the path or the minimum cross section in path.
  - Optionally select to *Report results* to the output window, including details of backbone open points, and minimum and mean cross-section.
- Network structure: Backbones are determined based on the network structure. If none of the options are selected, Backbones are calculated for all feasible inter-feeder paths.
  - (Optional) create backbones only if path leads to different substation.
  - (Optional) create backbones only if path leads to different HV/MV-transformer.
  - (Optional) create backbones only if tie open point is remote-controlled (as specified on the Reliability page of each switch).
  - Optionally select to *Report results* to the output window, including details of backbone open points.
- Scoring method: Backbones are determined using a scoring algorithm based on the restoration ability of the adjacent feeder. Scoring method settings are entered on the *Scoring Settings* page.
  - (Optional) specify the max. number of backbones per feeder.
  - Optionally select to *Report results* to the output window, including details of backbone open points, and loading/voltages of limiting elements.
  - Pointer to load-flow command (note for balanced AC calculation only).

### 42.3.2 Scoring Settings Page

If *scoring method* is selected on the Basic Options page, enter scoring settings on the Scoring Settings page. Backbones are determined based on the restoration ability of every inter-feeder path using Topology, Loading violation, and Voltage violation criteria.

For each criteria satisfied, the path receives the entered number of points. The path with the greatest number of points is the “winning” path.

#### Topology scoring

Define scoring settings for Topology scoring criteria:

- *Path leads to different substation.*
- *Path leads to different HV/MV-transformer.*
- *Tie open point is remote controlled.*
- Greater than a specified number of *remote-controlled switches on path*. A path to another Feeder receives the entered number of points if more (closed) remote-controlled switches than the entered number are on the path of the Backbone contained in the initial feeder.

#### Loading violation scoring

Assign Points for loading violations based on *individual loading constraints* or global loading constraints. If no element is overloaded, the calculation assigns the specified number of points. If *global loading constraints* is selected, then *Max. Loading* should also be defined.

Define scoring settings for Loading violation scoring criteria:

- *Restoring transformer (restoration mode)*. Consider a path from initial “feeder A” to “feeder B”. “Feeder A” is de-energised and the connection to “feeder B” via the tie open point is closed. A load flow is calculated in this so-called restoring mode and the entered number of points is assigned if the supplying HV/MV-transformer is not overloaded.
- *On backbone of restoring feeder (normal mode)*. Consider a path from initial “feeder A” to “feeder B”. A load flow is calculated (in so-called normal mode) and the entered number of points is assigned if no element on the potential backbone path contained in “feeder B”, the restoring feeder is overloaded in the base case.
- *On complete backbone (restoration mode)*. Consider a path from initial “feeder A” to “feeder B”. “Feeder A” is de-energised and the connection to “feeder B” via the tie open point is closed. A load flow is calculated in this so-called restoring mode and the entered number of points is assigned if no element on the potential backbone path is overloaded.
- *In complete feeder (restoration mode)*. Consider a path from initial “feeder A” to “feeder B”. “Feeder A” is de-energised and the connection to “feeder B” via the tie open point is closed. A load flow is calculated in this so-called restoring mode and the entered number of points is assigned if no element in the complete resulting feeder is overloaded (not only on the backbone as for the previous option).

### Voltage violation scoring

Define scoring settings for voltage violation criteria based on *individual voltage drop/rise constraints* or *global voltage drop/rise constraints*. If *global voltage drop/rise constraints* is selected, then *Max. drop* and *Max. rise* should also be defined. If no voltage limits are violated, the calculation assigns the specified number of points.

- *On backbone of restoring feeder (normal mode)*. Consider a path from initial “feeder A” to “feeder B”. A load flow is calculated (in so-called normal mode) and the entered number of points is assigned if no terminal on the potential backbone path contained in “feeder B” violates its voltage drop constraint and voltage rise constraint.
- *On complete backbone (restoration mode)*. Consider a path from initial “feeder A” to “feeder B”. “Feeder A” is de-energised and the connection to “feeder B” via the tie open point is closed. A load flow is calculated in this so-called restoring mode and the entered number of points is assigned if no terminal on the potential backbone path violates its voltage drop and rise constraint.
- *In complete feeder (restoration mode)*. Consider a path from initial “feeder A” to “feeder B”. “Feeder A” is de-energised and the connection to “feeder B” via the tie open point is closed. A load flow is calculated in this so-called restoring mode and the entered number of points is assigned if no terminal in the complete resulting feeder violates its voltage drop and rise constraint (not only on the backbone as for the previous option).

### 42.3.3 Tracing Backbones

When a Backbone is calculated, it always contains a connection to another Feeder via a tie open point. In the worst case of an outage close to the feeding point of the initial feeder, the initial feeder is de-energised by opening its feeding switch and restored by the second Feeder via the tie open point. These restoration steps can be simulated for an existing Backbone using the Backbone trace functionality. The trace buttons are located beside the *ComBbone* command, and can also be accessed via the main menu *Calculation → Distribution Network Analysis → Start trace....*

### 42.3.4 Example Backbone Calculation

Consider a case where there are two parallel feeders with multiple open-points. A Backbone calculation is conducted based on a criteria of minimum cross section in path, and with the *Max. number of backbones per feeder* set to “1”. Backbone objects are created within the Network Data folder.

To highlight Backbones, from the main menu select *View* → *Diagram Colouring* (or select the *Diagram Colouring* icon).

Under 3. *Other* select *Topology, Feeders*. Click on *Colour Settings*, and on the *Feeders* page select *Highlight backbones*.

Figure 42.3.1 shows the result, where the path through “Open Point 2” is highlighted as a result of the cross section of conductors in this path. Refer to Section 42.3.3 for details of how to trace the Backbone restoration steps.

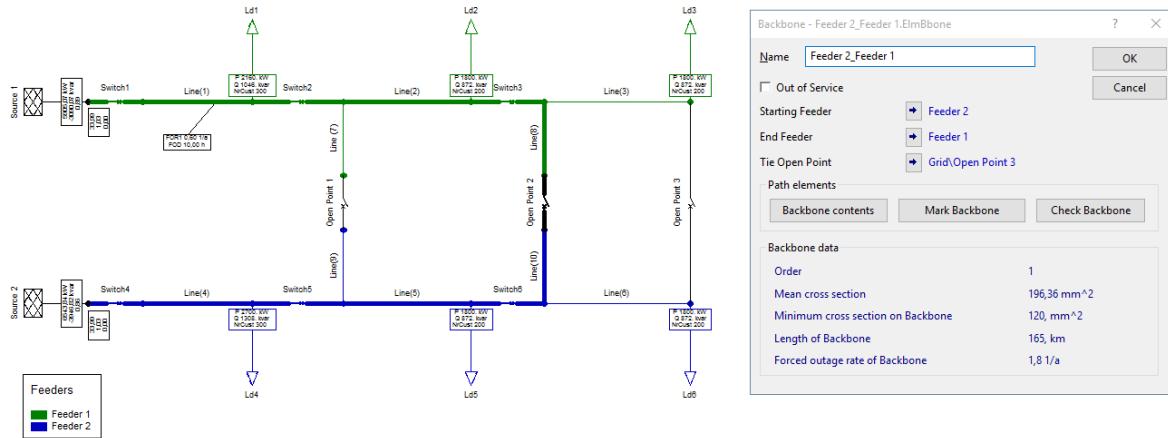


Figure 42.3.1: Example Backbone Calculation

## 42.4 Voltage Sag

The Voltage Sag Table Assessment (*ComVsag*) can be used to assess the expected frequency and severity of voltage sags within a network during an operating period, and determine the expected number of equipment trips due to deep sags. The *PowerFactory* Voltage Sag tool calculates a short-circuit at the selected load points within the system and uses the failure data of the system components to determine the voltage sag probabilities.

Voltage sag analysis is similar to probabilistic reliability analysis, in that it uses fault statistics to describe the frequency of faults, and then use these statistics to weight the results of each event and to calculate the overall effects of failures. However, reliability analysis looks for sustained interruptions as one aspect of quality of supply, whereas voltage sag analysis calculates the voltage drop during the fault until the protection system has disconnected the defective component.

This section describes the calculation options, how to perform a Voltage Sag Table Assessment, and how to view the results.

### 42.4.1 Calculation Options

#### 42.4.1.1 Basic Options Page

##### Load selection

Reference to the set of load points. A load point can be defined by a busbar, terminal, or load.

## Short-circuit command

Displays the short-circuit command that is used. The options for the short-circuit type will be changed during the voltage sag calculation, depending on the Advanced Options specified in the *ComVsag* dialog. However, other settings can be inspected or changed by clicking on the Edit button (→).

## Results

Reference to the results file that is used for storage of results.

## Exposed area limit

This defines the minimum remaining voltage for the voltage sag calculation to continue calculating short-circuits at busbars which are further away from the selected load points. If short-circuits at all busbars (at a certain distance away from all load points) result in voltages at the load points being higher than this limit, then no further short-circuit will be analysed.

### 42.4.1.2 Advanced Options Page

The *Advanced Options* page shows the various short-circuit types that can be analysed by the voltage sag assessment command. All components for which a failure model has been defined use the same short-circuit frequency. The relative frequency for each type of short-circuit is entered uniformly for all components.

## 42.4.2 How to Perform a Voltage Sag Table Assessment

A voltage sag table assessment is performed in two phases:

1. A results file with remaining voltages and short-circuit impedances is created by executing the *ComVsag* command. This can be done by selecting one or more nodes, right-clicking and executing the *Calculation → Voltage sag table...* option, or by initiating the command directly from the *Distribution Network Analysis* toolbar by clicking on the *Voltage Sag Table Assessment* icon (⚡).
2. A voltage sag plot is created by selecting one or more of the nodes for which the *ComVsag* command was executed, right-clicking and selecting the option *Plots → Voltage sag plot*

Alternatively,

- The *Load selection* in the *ComVsag* dialog can be completed manually with a set of objects. A load point is defined by a terminal, a busbar, or by a single-connection element (a load, motor, generator, etc.). These kinds of elements can be multi-selected from the single-line diagram or Data Manager. Once selected, right-click on them and select *Selections → Set - General → New...* from the context menu. This set can then be selected as the *Load selection*.
- A voltage sag plot can be created from the *Insert Plot* dialog (📈) manually, and the load points can then be selected from the list of analysed load points.

If several objects are selected which are all connected to the same busbar, then that busbar will be added only once to the set of load points.

The *Load selection* parameter in the voltage sag assessment command should be set to use the *SetSelect* which has the *Used for: Voltage sag table* flag set. However, any other selection can be assigned to the *Load selection*.

The voltage sag analysis simulates various faults at the selected busbars. The calculation starts with the selected load points, and proceeds to neighbouring busbars until the remaining voltage at all load

points does not drop below the defined *Exposed area limit*. The remaining voltages and the short-circuit impedances for all load points are written to the results file specified by the *Results* parameter.

After all relevant busbars have been analysed, the sag table assessment continues by analysing short-circuits at the midpoint of all lines and cables that are connected between the relevant busbars. Again, the remaining voltages and short-circuit impedances for all load points are written to the results file.

After the complete exposed area has been analysed in this way, the results file contains the values for Z\_F1, Z\_F2, Z\_F0, Z\_S1, Z\_S2, Z\_S0 and ura, uia, urb, uib, urc, uic for the two ends of all relevant lines and cables and at their midpoints.

To reduce computation time, the written impedances are interpolated between the ends of a line and the middle with a second-order polynomial. Then, the remaining voltages and various source impedances are estimated. These estimated impedances are also interpolated between the ends and the midpoint. The interpolated impedances are then used to estimate the remaining voltages between the ends and the midpoints of the lines or cables. This quadratic interpolation gives a good approximation for longer lines, as well as parallel lines.

#### 42.4.3 Voltage Sag Table Assessment Results

The voltage sag tables are not calculated until a voltage sag plot is constructed. Upon reading the remaining voltages, short-circuit frequencies and short-circuit impedances from the results file, a voltage sag table is constructed for each selected load point.

Because there is no single definition of a voltage sag, the plot offers a selection of sag definitions:

- Minimum of Line-Neutral Voltages.
- Minimum of Line-Line Voltages.
- Minimum of Line-Line and Line-Neutral Voltage.
- Positive Sequence Voltage.

Secondly, the x-variable against which the sag frequency will be shown has to be selected. Possible x-variables are:

- Remaining Voltage.
- Nom. Voltage at Shc-Busbar.
- Fault Clearing Time.
- Short-Circuit Type.

Additionally, the x-variable can be sub-divided according to a split-variable (parameter name: *Split Bars in*). Possible split variables are:

- no split.
- any of the possible x-variables.

The same parameter cannot be selected for the x-variable and the split-variable.

An example of the resulting voltage sag plot, is shown in Figure [42.4.1](#).

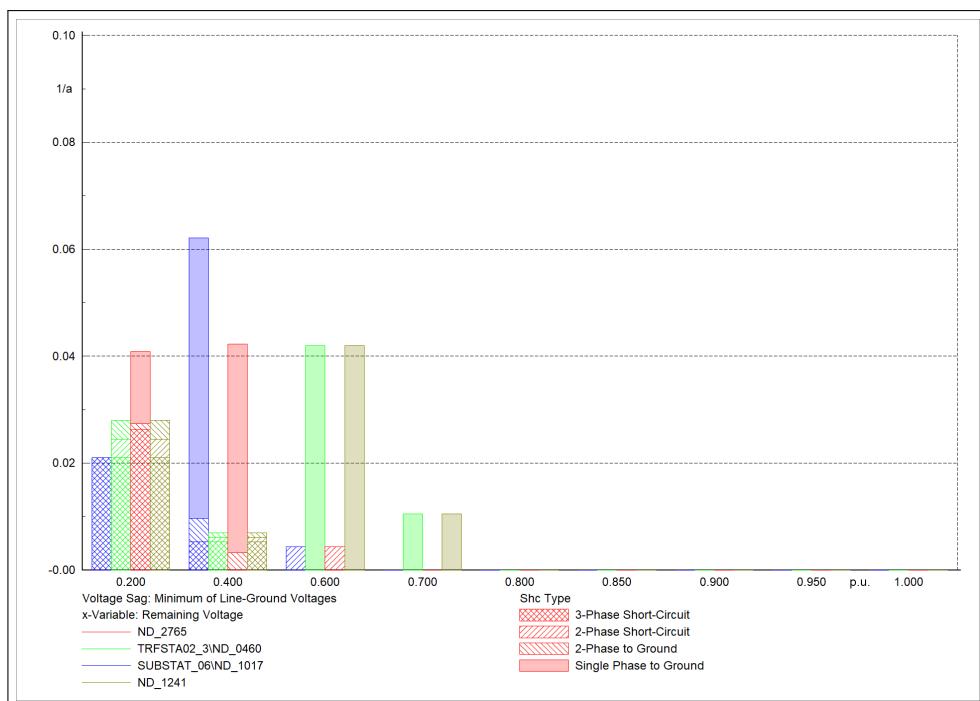


Figure 42.4.1: Example Voltage Sag Plot

The voltage sag plot always shows the annual frequency of occurrence on the y-axis.

The example plot shows a bar for each load point for each x-variable, which is the Remaining Voltage. All three loads can be seen to suffer either deep sags (remaining voltage less than 0.4 p.u.), or shallow sags, although the values at 0.8 p.u. are also significant. Each bar is subdivided to the nominal voltage at SHC-Busbar. The shallow sags are caused by the low voltage network, as well as the deep sags. The high voltage network seems to cause moderate voltage sags. This is caused by the fact that the low voltage networks in this example are radially operated and the higher voltage networks are meshed. More detailed information about a specific value in the voltage sag plot can be viewed in the balloon help that appears when placing the mouse over a bar or part of a bar (without clicking).

The voltage sag plot dialog has a **Report** button which outputs the voltage sag plot data to the output window. A table for each selected load point will be written in accordance to the selected Voltage Sag definition, x-Variable and Split Bars in selection.

## 42.5 Low Voltage Load Flow Calculation

In low-voltage grids it is often not possible to make exact statements about the specific power of certain consumers and generation units. Furthermore, electrical loads such as electric vehicles and heat pumps can lead to a greater utilisation of the existing power system and need to be considered. It is therefore necessary to execute load flow calculations to evaluate the network utilisation with the use of as little data as possible, only by categorising the types of load and generation.

The Low Voltage Load Flow Calculation (*ComLvldf*) enables the calculation of realistic maximum loadings of branch elements like lines and transformers as well as minimum or maximum voltages at all nodes in a network considering these requirements. Basically, the electrical loads and generation units only need to be categorised in order to find bottlenecks in the grid without overestimating the criticality at the same time. The Low Voltage Load Flow Calculation can be used in both radial and meshed grids and is based on the use of Coincidence Definitions (*IntCoincdef*) that take a stochastic approach for the power consumption of multiple consumers into account.

The structure of this section is as follows:

- Section 42.5.1 gives a short overview of the technical background for the Low Voltage Load Flow.
- Section 42.5.2 explains how *Coincidence Definitions* can be defined.
- Section 42.5.3 describes how low-voltage loads have to be set up.
- Section 42.5.4 looks at the use of generation units in the Low Voltage Load Flow.
- Section 42.5.5 goes into detail about the configuration of the Low Voltage Load Flow command.
- Section 42.5.6 explains the results of the Low Voltage Load Flow Calculation.

## 42.5.1 Technical Background

In the following section, the technical background is described using the example of electrical loads. However, the same behaviour can also be applied to generation units.

Coincidence curves  $g(n)$  take the stochastic consumption behaviour of low voltage loads into account. They depend on the number of customers  $n$  of a specific load type that have an impact on the investigated branch or node. The more consumers have to be considered, the less the share of each individual consumer statistically becomes, since it is not to be expected that all consumers exhaust their maximum permissible power at the same time. An example of a Coincidence Curve can be seen in Figure 42.5.1.

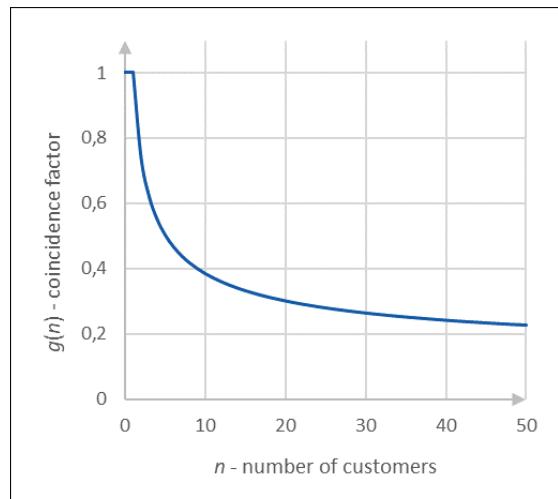


Figure 42.5.1: Exemplary Coincidence Curve for households

A calculation example is presented in the small example in Figure 42.5.2 with four 24 kVA loads. The coincidence factors are 0.74 for two and 0.55 for four customers with the curve in Figure 42.5.1, which leads to 35.4 kVA (instead of  $2 \cdot 24 \text{ kVA} = 48 \text{ kVA}$ ) and 53.0 kVA (instead of  $4 \cdot 24 \text{ kVA} = 96 \text{ kVA}$ ) in the respective branches using Equation 42.1. It should be noted that Kirchhoff's current law (the sum of all currents is zero at each node) is not valid in these calculations, as the coincidence factor at each branch connected to the same node can and often will be different.

$$S_{\max \text{ total}} = n \cdot g(n) \cdot S_{\max} \quad (42.1)$$

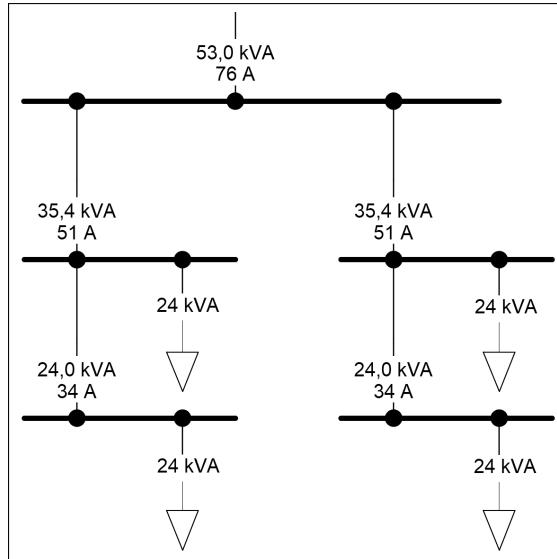


Figure 42.5.2: Example of a simple calculation using coincidence curves

In case of individual power settings  $S_{\max,i}$  in the loads (see Section 42.5.3.1), Equation 42.1 cannot be applied. There are scenarios with power settings that differ greatly from one another and could result in a total power that is smaller than one of the individual loads ( $S_{\max \text{ total}} < S_{\max,i}$ ). To avoid this, Equation 42.2 is used to calculate the total power. For this, the power of all relevant loads (or customers, if several customers are modelled with a single load element) is sorted in descending order and a  $(n+1)$ th power of zero is appended, so that the last power difference can be calculated. As to be expected, Equation 42.2 has the same result as Equation 42.1 if the power of all loads is the same.

$$S_{\max \text{ total}} = \sum_{i=1}^n [i \cdot g(i) \cdot (S_{\max,i} - S_{\max,i+1})] \quad (42.2)$$

To decide which loads have an impact on a specific branch flow (in meshed networks) or node voltages (in meshed and radial networks), a sensitivity analysis is used, see Chapter 50 for more information. In those cases the calculation uses sensitivities instead of power values. Sensitivity thresholds for the consideration can be applied, as explained in Section 42.5.5.2.

## 42.5.2 Coincidence Definition

The Coincidence Definition object (*IntCoincdef*) is used to define the coincidence curve  $g(n)$  of load or generation categories.

### Default power values

For a quick set-up of existing or new loads or generation units, it is possible to define default values for power and power factor. These values are applied in the Low Voltage Load Flow if the setting *Individual power consumption* in the element is disabled. The *Input Mode* can be set either to “Apparent power (S)” or “Active power (P)”. If the maximum active power is entered, the maximum apparent power is calculated with Equation 42.3.

$$S_{\max} = \frac{P_{\max}}{\cos \phi} \quad (42.3)$$

## Coincidence

As *Input Mode* it is possible to use either a “Formula” or a “Table” to define the coincidence curve.

### Formula

If “Formula” is selected as *Input Mode*, Equation 42.4 is applied. Three parameters are used to define the coincidence curve with this setting:

- *Limit (ginf)*: Determines which coincidence factor results for  $n \rightarrow \infty$
- *End of constant part (A)*: Gives the possibility to define a constant part with  $g = 1$ . Application examples could be heat pumps or electric vehicles, where the first “ $A$ ” units should not reduce the consumption of each other.
- *Exponent (B)*: Determines the steepness of the reduction of  $g$ . The higher “ $B$ ”, the faster the decrease.

$$g(n) = g_{\infty} + \frac{1 - g_{\infty}}{(n + 1 - A)^B} \quad (42.4)$$

If a Gaussian distribution is assumed (described with the default values  $A = 1$  and  $B = 0.5$ ), the coincidence curve is given with Equation 42.5

$$g(n) = g_{\infty} + \frac{1 - g_{\infty}}{\sqrt{n}} \quad (42.5)$$

### Table

If “Table” is selected as *Input Mode*, an individual coincidence curve can be created. With a right-click in the table, rows can be inserted, appended or deleted. In each row, a value for the “Number of units”  $n$  and “Coincidence factor”  $g(n)$  has to be entered. It is not necessary to define values for each possible number of units as an automatic linear interpolation takes place between given values. The coincidence factor of the last row is automatically used and displayed as limit value  $g_{\infty}$  (*ginf*).

### 42.5.3 Setting up Low Voltage Loads

Low Voltage Loads (*ElmLodlv* and *ElmLodlp*) can have a linked Low Voltage Load Type (*TypLodlv*) in *Basic Data → Type*. To use the Low Voltage Load Flow Calculation, this *TypLodlv* has to be extended with a Coincidence Definition (*IntCoincdef*) in *Low Voltage Load Flow → Coincidence definition*. It is also possible to select an individual *IntCoincdef* directly in Low Voltage Loads without setting up a *TypLodlv*. This is explained in Section 42.5.3.2.

#### 42.5.3.1 Settings in the Low Voltage Load - General

- *Number of Customers (NrCust)*: Determines how many customers are represented with this load object. If no individual power consumption is used, the maximum theoretical power of the load object is calculated with  $NrCust \cdot S_{max}$ , where  $S_{max}$  is taken from the linked coincidence definition.
- *Adjusted by Load Scaling*: Defines whether the load consumption is influenced by scaling factors as used for the feeder loading scaling for example.

### Individual power consumption

If the *Individual power consumption* is deactivated (as it is by default), power and power factor for the load object are taken from the coincidence definition as explained in Section 42.5.2. However, if more detailed knowledge about the specific power or the annual energy consumption is available, it is possible to activate the *Individual power consumption*. Additional settings are shown in this case:

#### *Input Mode*

The input mode defines which values are given by the user. The available settings are the same as on the *Load Flow* page of the low-voltage load and therefore use the same load variables:

- $S, \cos(\phi)$ : Apparent power and power factor
- $P, \cos(\phi)$ : Active power and power factor
- $U, I, \cos(\phi)$ : Voltage, current and power factor
- $E, \cos(\phi)$ : Yearly energy consumption, power factor, and consumption profile

#### *Use consumption as*

The inserted values for power or energy can represent different kinds of measurements or approaches, therefore it has to be defined how these values should be interpreted in the calculation:

- *Theoretical sum*: Values represent the maximum theoretical power **of all** customers in this load.
- *Maximum per customer*: Values represent the maximum theoretical power **per customer** in this load.
- *Maximum with coincidence*: Values represent the resulting total power taking into account the coincidence of all customers that are modelled with this load object. This selection therefore only differs from *Theoretical sum* if the *Number of Customers* is larger than 1.
- *Average (value for g<sub>inf</sub>)*: Values represent the total power of all customers in this load object with the assumption  $n \rightarrow \infty$  and therefore  $g_{\infty}$ .

An example for these settings is presented in the next section along with the resulting values.

If *Theoretical sum* or *Maximum per customer* is selected, an *Additional scaling factor* can be specified that scales the given power values. This makes sense, for example, when using the yearly energy in combination with standard load profiles, as these already take coincidence into account. By using a factor, the peak power can then be scaled up accordingly (the Velander equation is a popular example to determine such a factor).

### 42.5.3.2 Settings in the Low Voltage Load - Coincidence Definition

#### Coincidence definition

- *Used coincidence definition*: By default, the coincidence curve that is selected in the linked *TypLodlv* is applied (selection: “From type”). Alternatively, an “Individual” coincidence curve can be selected for the specific load object. This enables to link a coincidence curve without using a *TypLodlv*. It is also possible to set the coincidence definition to “None”.
- *Consider coincidence*: By default, all Low Voltage Loads are considered with their defined coincidence. This selection can be deactivated, to ignore a specific load element in all other related coincidence definitions and keep its power consumption to its maximum theoretical value.

### Resulting values

If a Coincidence Definition is linked, the resulting values for active and apparent power are shown:

- *theoretical*: Maximum theoretical power consumption, when  $g = 1$  is applied.
- *with coincidence*: Power consumption, when  $g$  is calculated with the selected coincidence definition and the *number of customers* from the *General* tab. This part is only visible if *Consider coincidence* is selected.

If *Individual power consumption* is selected for a load element, the selection *Use consumption as* from Section 42.5.3.1 has an influence on the resulting values. The influence of this setting in a single load element with the following exemplary assumptions is shown in Table 42.5.1:

- $n = 2$
- $g(2) = 0.7$
- $g(\infty) = 0.1$

Use consumption as	$S_{\text{input}}$ in kVA	$S_{\text{theoretical}}$ in kVA	$S_{\text{with coincidence}}$ in kVA
Theoretical sum	10	10	7
Maximum per customer	10	20	14
Maximum with coincidence	10	14.29	10
Average (value for ginf)	10	100	70

Table 42.5.1: Influence of the *Use consumption as* setting on the resulting values

---

**Note:** If the number of customers in the load element is 1, the resulting values “theoretical” and “with coincidence” are always identical, since  $g(1) = 1$ .

---

### 42.5.4 Setting up Generation Units

Generation units *ElmGenstat* and *ElmPvsy*s can have a linked coincidence definition (*IntCoincdef*) on their *Low Voltage Load Flow* page. The following settings are available.

- *Number of parallel units (ngnum)*: Determines how many generations units are represented with this element. If no individual power consumption is used, the maximum theoretical power of the whole generation object is calculated with  $ngnum \cdot S_{\text{max}}$ , where  $S_{\text{max}}$  is taken from the linked coincidence definition.
- *Used coincidence definition*: By default, an “Individual” coincidence curve can be selected for the specific generation unit. It is also possible to set the coincidence definition to “None”.
- *Consider coincidence*: By default, all generation units are considered with their defined coincidence. This selection can be deactivated, to ignore a specific generation unit in all other elements with the same (or a coupled) coincidence definition and keep its power consumption to its maximum theoretical value.

### Resulting values

If a Coincidence Definition is linked, the resulting values for active and apparent power are shown as explained for loads in Section 42.5.3.2. For generation units, the entered power values are always assumed to be the *Maximum per generation unit*.

## Individual power generation

If the *Individual power consumption* is deactivated (as it is by default), power and power factor for the generation unit are taken from the coincidence definition as explained in Section [42.5.2](#). However, if more detailed knowledge about the specific power is available, it is possible to activate the *Individual power generation*. Additional settings are shown in this case which are the same as on the *Basic Data* page of the generation unit and therefore use the same variables:

- *Rated apparent power*
- *Rated power factor*

## 42.5.5 Low Voltage Load Flow Configuration

The Low Voltage Load Flow Calculation (*ComLvldf*) offers the possibility to adjust some global settings that are explained in the following.

### 42.5.5.1 Basic Options Page

#### Load Flow configuration

The following calculation methods are supported and can be used in the calculation:

- AC Load Flow, balanced, positive sequence
- AC Load Flow, unbalanced, 3-phase (ABC)

The corresponding load flow command which is used in the calculation is linked (→) in *Load Flow*. More information on the general load flow settings can be found in Chapter [25.3](#).

#### Coincidence calculation

The selection of the coincidence calculation determines the type of elements for which the coincidence factors are calculated and applied. It also determines whether the highest or lowest voltages that can occur at the nodes are determined.

- *For loads (consumption case)*: The coincidence calculation is done for all loads in the network. Generators can be considered with a constant contribution if they are not scaled to zero. The calculated voltages are the minimal voltages that can occur.
- *For generators (production case)*: The coincidence calculation is done for all generators in the network. Loads can be considered with a constant contribution if they are not scaled to zero. The calculated voltages are the maximum voltages that can occur.

#### Global scaling factors

It is possible to define global scaling factors for loads, generators and motors as explained in Section [25.3.6](#).

- *Custom*: Enables the setting of individual scaling factors for loads, motors and generators.
- *Automatic*: For the consumption case, loads and motors are scaled with 100 % while generators are scaled with 0 %. For the production case, generators are scaled with 100 % while loads and motors are scaled with 0 %.

## Output

- *Short*: Only basic information about the status of the calculation is written to the output window.
  - *Detailed (full load flow output)*: Additional information about reference elements, iteration loops and conditions of relevant controllers is written to the output window.
- 

**Note:** QDSL models are ignored in the Low Voltage Load Flow Calculation.

---

### 42.5.5.2 Advanced Options Page

#### Additional scaling

In order to be able to quickly run through different scenarios, it is possible to adjust all loads and generation units in the network that have the same coincidence definition or no coincidence definition with these additional scaling settings.

- *Scale loads without coincidence curve to zero*: If selected, the power of all low-voltage loads (*ElmLodlv* or *ElmLodlvp*) without a linked coincidence definition is set to zero.
- *Scale generators without coincidence curve to zero*: If selected, the power of all generation units (only *ElmGenstat* and *ElmPvsys*) without a linked coincidence definition is set to zero.
- *Scaling by curve*: With the button (▼), a “Scaling Factors for LV Load Flow” object (*SetLvscale*) can be selected. The corresponding *SetLvscale*, which is used in the calculation, is then linked (→) and can be edited. See Section 42.5.5.3 for details about *SetLvscale*. This object can be used to scale specific coincidence curves.
- *Assess number of active units stochastically*: If selected, a “Considered quantile” can be defined to account for a probabilistic influence. This is explained in Section 42.5.5.3.

#### Coupling of different curves

With the button (▼), a “Coupling Definition” (*IntCrosscoinc*) can be selected. The corresponding *Crosscoinc* object is then linked (→) and can be edited. See Section 42.5.5.4 for details about *IntCrosscoinc*.

#### Elements with coincidence definition

While the global scaling factors on the *Basic Options* page are used to scale loads and generation units with and without coincidence definitions equally, these settings will only affect elements with a linked coincidence definition.

- *Allow zero consumption/generation as worst case*: Sometimes, the superposition of loads and generation units is not the most critical calculation case. If this selection is activated, it is additionally checked whether a load (in the consumption case) or generation (in the production case) of zero results in the worst voltages or loadings.
- *Power of Generators/Loads*: If the global scaling is not set to 0 %, this additional option of only adjusting the influence of elements with coincidence definitions is enabled. In the consumption case, the behaviour of generators and in the production case the behaviour of loads is treated with this option. With the selection *Theoretical maximum*, the power for a coincidence factor of 1 is assumed while the selection *Scale to zero* sets the power of all relevant elements with coincidence definition to zero.

### Threshold for consideration of effects

Sensitivity matrices are used to consider the relevant influence of all elements with coincidence curves to branches and nodes in the calculation. In large grids, especially if a high number of elements with coincidence curves are modelled, these matrices become large. Therefore, it is necessary to define thresholds to avoid a calculation with dense sensitivity matrices and improve the performance. The influence of an element with a coincidence curve on a specific branch or node with values below these thresholds in the sensitivity matrix is neglected. The default settings should be used if there are no performance issues.

- *Influence on power flow*: threshold for the minimal considered influence of an added apparent power at a node on the power flow through a branch element.
- *Influence on voltage*: threshold for the minimal considered influence of an added apparent power at a node on the voltage at another node.

#### 42.5.5.3 Scaling by coincidence curve

The object *SetLvscale*, that can be linked in the Advanced Options page of *ComLvldf*, is used to scale elements according to their coincidence curve. These curves have to be inserted in the table *Curves*. This can either be done manually if only some curves are needed or automatically with the button **Add used curves**.

There are two different possibilities to scale elements by their coincidence curve with *SetLvscale*. Both methods can be used simultaneously and are explained in the following.

##### Direct scaling of the power

By changing the “Scaling Factor”, the power of all elements using the corresponding coincidence curve is directly adjusted. The number of customers or units and thus the effective coincidence factor is not changed by this scaling.

##### Scaling with percentage of active units

Changing the percentage of “Active Units” influences the consumed (or generated) power by changing the number of considered customers (or units) that have an impact at a specific node in the grid. This makes it possible, for example, to vary the penetration of the grid with electric vehicles and thus to carry out prognosis studies.

In case of a radial feeder and identical power settings in all elements, the influence of the “Active Units” setting (AU) can exemplarily be calculated with the following equations (the smallest possible value for  $n_{AU}$  is always 1):

$$n_{AU} = \frac{AU}{100 \%} \cdot n \quad (42.6)$$

$$S_{\max \text{ AU}} = n_{AU} \cdot g(n_{AU}) \cdot \sum_n S_{\max,i} \quad (42.7)$$

As explained in Section 42.5.1 this general approach has some limitations in case of individual power settings. Therefore, generally the Equation 42.8 is used to calculate the total power. If all  $S_{\max,i}$  are identical, the result is the same as calculated with Equation 42.7.

$$S_{\max \text{ total}} = \sum_{i=1}^n \left[ i \cdot \frac{AU}{100 \%} \cdot g \left( i \cdot \frac{AU}{100 \%} \right) \cdot (S_{\max,i} - S_{\max,i+1}) \right] \quad (42.8)$$

**Note:** It is possible to save the “Scaling Factor” and the “Active Units” settings in Operation Scenarios. However, the corresponding fields in the object *SetLvscale* are not displayed in blue, as these settings are stored in the objects *ElmLvscale* that are located in the *SetLvscale*. *SetLvscale* is saved in the active Variation (if a Variation is present). This means that newly created *SetLvscale* will be stored within a Variation, but are then also activated, deactivated and deleted with the Variation.

---

#### *Stochastic approach with considered quantiles*

If a *SetLvscale* has been assigned and the option *Assess number of active units stochastically* is activated, a *Considered quantile* can be set to enhance the *Active Units* scaling with the use of a quantile function. The chosen quantile is used to determine the number of *Active Units* that is assumed at a specific node in the grid.

The mechanism is explained in the following with an example. 10 customers with the same coincidence definition are assumed and the *Active Units* scaling is set to 20 %. Without a *Considered quantile* this would lead to a calculation with 2 active units assumed ( $20 \% \cdot 10 = 2$ ). The quantile function is set to 95 %, which means that all cases within this probability shall be considered. In the following equations, the probability of different numbers of active units are calculated with a binomial distribution:

$$p_{0 \text{ active units}} = \binom{10}{0} \cdot 0.2^0 \cdot 0.8^{10} = 0.107 \rightarrow 10.7 \% < 95 \% \quad (42.9)$$

$$p_{\leq 1 \text{ active units}} = \binom{10}{1} \cdot 0.2^1 \cdot 0.8^9 + p_{0 \text{ active units}} = 0.376 \rightarrow 37.6 \% < 95 \% \quad (42.10)$$

$$p_{\leq 4 \text{ active units}} = \binom{10}{4} \cdot 0.2^4 \cdot 0.8^6 + p_{\leq 3 \text{ active units}} = 0.967 \rightarrow 96.7 \% > 95 \% \quad (42.11)$$

It can be seen that the probability of 4 or less loads has a higher probability than the chosen quantile of 95 %. Therefore, the Low Voltage Load Flow Calculation will assume 4 active units as this is the extreme case with the chosen quantile. In comparison with the case without a *Considered quantile* (2 active units), the results will reflect a higher utilisation.

#### 42.5.5.4 Coupling of different curves

By default, all coincidence curves are independent of each other, which means that the number of customers or units for all given curves is counted separately. However, a coupling between different categories and their respective coincidence curves can be explicitly defined with the coupling object *SetCrosscoinc* that can be selected in the calculation command (see Section 42.5.5.2). This coupling definition can be used to define which categories are mutually considered in the coincidence and which are not.

##### Curves

The coincidence curves (see Section 42.5.2) that shall be considered for the coupling can be defined in this table. With the button **Add all available curves** all coincidence curves can be inserted automatically. The way the button works depends on where the *SetCrosscoinc* is stored:

- *IntCrosscoinc stored in the project library*: All coincidence curves that are stored in the project library or that are referenced in the project are automatically added.

- *IntCrosscoinc stored in a custom global library*: All coincidence curves that are stored in the custom global library are automatically added.
- *IntCrosscoinc not stored in a library*: All coincidence curves that are stored in the same folder as *IntCrosscoinc* are automatically added.

### Mutual consideration

In this matrix the coupling between the selected coincidence curves can be defined. Only values in the range 0 to 1 are allowed, where zero stands for no consideration and one for full consideration. Intermediate values are also possible, a value of 0.5, for example, means that only half the number is considered.

As an example, different load types could be set up for 11 and 22 kW charging stations, whose coincidence should be coupled. On the other hand, the coincidence of households, heat pumps and electric vehicles should possibly be completely independent of each other. An exemplary coupling definition required for this behaviour can be seen in Table 42.5.2.

	<b>Household</b>	<b>Heat Pump</b>	<b>EV 11 kW</b>	<b>EV 22 kW</b>
<b>Household</b>	1	0	0	0
<b>Heat Pump</b>	0	1	0	0
<b>EV 11 kW</b>	0	0	1	1
<b>EV 22 kW</b>	0	0	1	1

Table 42.5.2: Exemplary mutual consideration matrix in the coupling definition

### Full self-consideration

By default, elements with the same coincidence curves consider each other with a factor of 1 (Full self-consideration as can be seen in Table 42.5.2). In some cases (e.g. when single-phase loads are modelled as three-phase loads for simplification reasons) it may be useful to reduce this factor. To enable self-consideration factors below 1 in the main diagonal, the selection *Full self-consideration* has to be deactivated.

### Symmetric consideration

The mutual consideration is symmetric by default. By deactivating the setting *Symmetric consideration*, it becomes possible to define asymmetric mutual consideration matrices. This is useful, for example, if charging stations are already integrated in some household loads, but have not been explicitly modelled. Then it would make sense to take them into account for the coincidence of explicitly modelled charging stations. On the other hand, however, the explicitly modelled charging stations should not be considered for the coincidence calculation of households.

For the asymmetric case, the example in Table 42.5.3 can be read as follows: the coincidence curve "Household" will consider the number of customers with the coincidence curve "EV" with zero, while the coincidence curve "EV" will consider the number of customers with the coincidence curve "Household" with the factor 0.5.

	<b>Household</b>	<b>EV</b>
<b>Household</b>	1	0
<b>EV</b>	0.5	1

Table 42.5.3: Exemplary asymmetric mutual consideration matrix in the coupling definition

## 42.5.6 Results of the Low Voltage Load Flow Calculation

### 42.5.6.1 Result Variables

Most results of the Low Voltage Load Flow Calculation are stored in the same variables as for the normal Load Flow Analysis, therefore the same user-defined routines and automations can be applied. While the maximum values are always entered for currents and powers, minimum values are given for voltages in the consumption case and maximum values in the production case.

### 42.5.6.2 Feeder Load Scaling

As described for the normal Load Flow Analysis in Section 25.4.3, the feeder load scaling can also be performed in a Low Voltage Load Flow Calculation if *ElmFeeder* objects have been defined. For this to be considered in the calculation, two conditions must be met:

- The setting “Feeder Load Scaling” in the linked Load Flow command has to be activated.
- In each feeder with a defined setpoint, the setting “Adjusted by Load Scaling” must be activated in at least one load element.

For the Low Voltage Load Flow Calculation one of the following setpoints can be selected for each feeder object:

- No scaling
- Apparent Power
- Current
- Active Power
- Manually (To select an individual *Scaling Factor*)

---

**Note:** The selection of a *Phasewise input* for the Feeder Load Scaling is possible. However, the scaling will be done for the sum of the three input values.

---

### 42.5.6.3 Update Database

As described for the normal Load Flow Analysis in Section 25.5.7, it is possible to overwrite input parameters with calculated values by using the *Update Database* (*ComDbupd*  ) command. The handling of the tool is the same, but the selection of variables that can be updated is limited to the following two values:

- Transformer taps
- Scaling factors of loads

## 42.6 Tie Open Point Optimisation

The function of the Tie Open Point Optimisation (TOPO) (*ComTieopt*) is to optimise a radial system of connected feeders by determining the best location for network open points. An open point can be moved by the TOPO tool by opening and closing switches on the networks to be optimised.

This section is separated into three sub-sections. Firstly, the steps to access the TOPO tool are described. Next, the background and function of the TOPO tool is presented and finally the procedure for running a Tie Open Point Optimisation is described. The TOPO command can be accessed from the Distribution Network Optimisation Tools, as shown in Figure 42.1.2

### 42.6.1 Technical Background

The function of the Tie Open Point Optimisation (TOPO) tool is best explained using an example. Consider the network illustrated in Figure 42.6.1

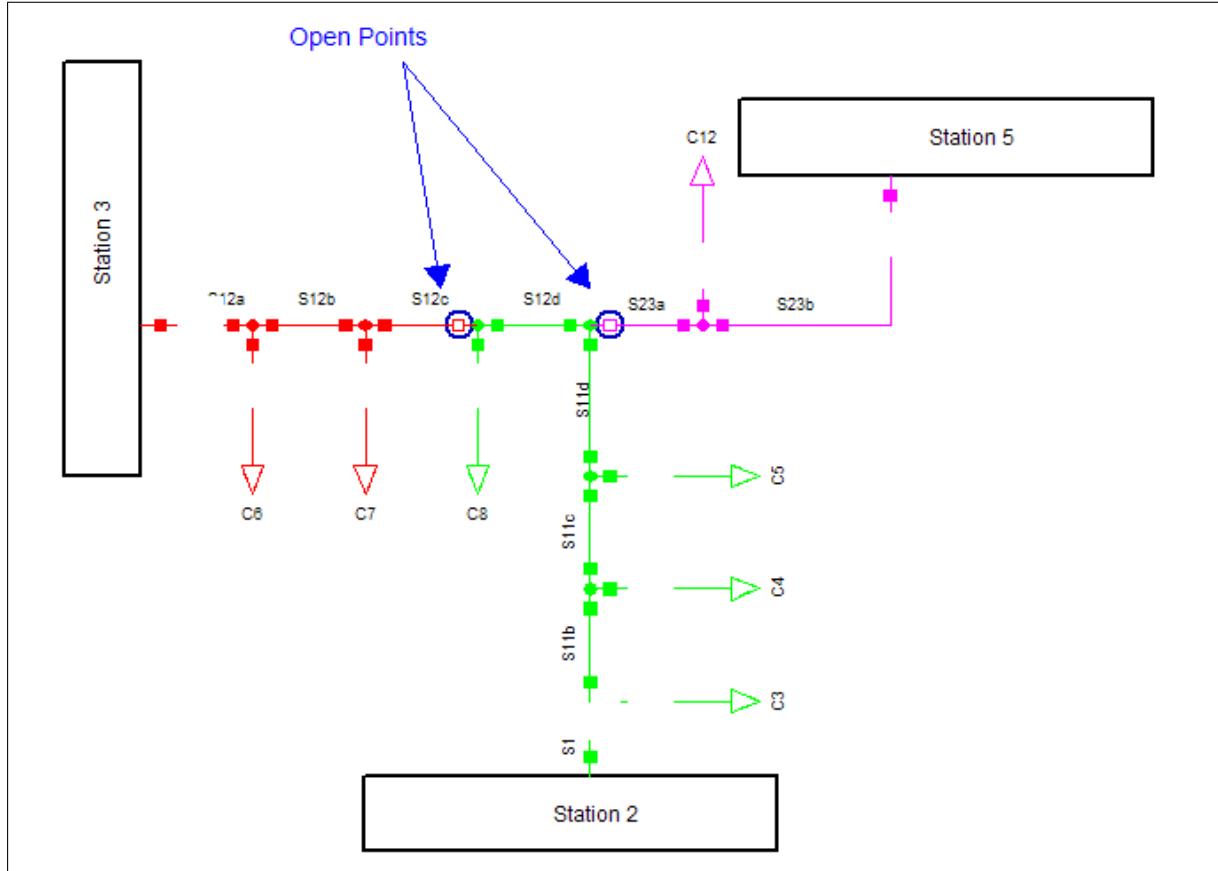


Figure 42.6.1: Example network for Tie Open Point Optimisation

The network consists of three feeders, one from each of the three “stations”. Each feeder begins at a “station” and ends at one of the two illustrated open points. The two open points in this network are not necessarily the optimum open points. For example, it might be more economic (i.e. less network losses and / or less impact of outages) to shift these open points by closing the open switches and opening two switches in different positions on the feeders. The purpose of the TOPO tool is determine these optimum open points automatically. Additionally, the TOPO tool can automatically consider network voltage and thermal constraints - for instance it might be economic to shift an open point in terms of reducing systems losses, however doing so might cause a cable to overload.

### 42.6.2 How to run a Tie Open Point Optimisation

This section describes the procedure for running a Tie Open Point Optimisation (TOPO) calculation. The steps are summarised below, and discussed in more detail in the following sections:

- How to Create Feeders.
- How to configure the Tie Open Point Optimisation Command.
- How to configure constraints for the Tie Open Point Optimisation.
- How to configure Switching Options.
- How to configure Reliability Options.
- How to configure the Advanced Options.
- How to configure Scenario/Time-Sweep Options.
- How to configure Output Options.

### How to Create Feeders

The TOPO tool requires that feeders are defined for the section of the network that you wish to optimise. Additionally, the TOPO tool only works on radial feeders - mesh systems cannot be optimised automatically. Furthermore, it is recommended that the target feeders for optimisation do not have any overloaded components or voltage violations in the base case.

To define a feeder, right-click the cubicle at the beginning of the feeder and select the *Network Groupings* → *Feeder* → *New...*. Alternatively, for fast creation of multiple feeders right-click the bus the feeder/s are connected to and select the option *Network Groupings* → *Feeder* → *New...*. More information on feeders and feeder creation can be found in Chapter 15: Grouping Objects, Section 15.6.

### How to configure the Tie Open Point Optimisation Command

After a set of feeders has been defined, open the TOPO tool and configure the basic options:

1. Click the *Change Toolbox* icon (▼) and select *Distribution Network Optimisation*.
2. Open the dialog for the Tie Open Point Optimisation tool (⊕).
3. Use the selection control for Feeding Points to select previously defined feeder/s, or a feeder “Set”. If the *Select* option is chosen and multiple feeders are selected, a “Set” of feeders will automatically be created within the active study case. By default the set will be named ‘Tie Open Point Optim. - Feeder Set’.

---

**Note:** It is generally recommended to define all feeders in the network as Feeders, and to conduct a TOPO calculation for ‘All Feeders’.

4. Select the desired *Objective Function* to minimise losses and / or reliability indices. If *Optimisation of Reliability Indices* or *Cost Optimisation (Losses + Reliability)* is selected, complete the required fields on the Reliability page, see (How to configure Reliability Options).
5. “Method” can be selected, where the optimisation explores the meshes iteratively, uses a stochastic optimisation according to Section 42.11 (Simulated Annealing or Genetic Algorithm).
6. “Balanced, positive sequence” or “Unbalanced” network representation can be selected. The Load-flow command referenced below these radio buttons is automatically adjusted to the correct calculation method based on this selection.
7. Optional: You can inspect and alter the settings of the load-flow command that is used for determining the losses and identifying the constraints of the system by clicking the blue selection arrow next to load-flow command.

## How to configure constraints for the Tie Open Point Optimisation

It is optional whether you choose to consider thermal and voltage constraints for the Tie Open Point Optimisation. If you wish to consider constraints follow these steps:

1. Open the Tie Open Point Optimisation dialog and select the Constraints page.
2. Optional: Choose to enable or disable the option *Consider Thermal Constraints*. If enabled, the TOPO tool will automatically consider thermal constraints in the network. Therefore, if an optimal point were to cause an thermal overload on any system component, then this would not be considered as a valid open point for reconfiguration of the system. There are two more options for thermal constraints:
  - Global constraint for all components. This is the default option. If enabled you must enter a maximum thermal loading percentage in the *Max. thermal loading of components* field. Note this option overrides the individual component thermal limits.
  - Individual constraint per component. Select this option to automatically consider each component's unique thermal rating. Note, the thermal rating for each component is determined by the field *Max Loading* within the Tie Open Point Optimisation page of each component.
3. Optional: Choose to enable or disable the option *Consider Voltage Constraints*. If this option is enabled then each terminal in the system is checked against the Lower and Upper limit of allowed voltage. If a particular open point causes a voltage violation, then such an open point cannot be considered as "optimal". There are two options for configuring the upper and lower voltage limits:
  - Global constraints for all terminals (absolute value). If you choose this option then you must enter an upper and lower voltage limit in the two corresponding fields within this dialog box.
  - Individual constraint per terminal. If you choose this option, then each terminal has a unique voltage limit which is assigned on the Tie Open Point Optimisation page of each terminal (note that this excludes Substation internal nodes).
4. Optional: Choose to enable or disable the option *Consider Voltage Drop / Rise*. If this option is enabled then each feeder in the system is checked against the Maximum Voltage Drop / Rise. If a particular open point causes a voltage violation, then such an open point cannot be considered as "optimal". There are two options for configuring the maximum voltage drop / rise limits:
  - Global constraints for all feeders (percent). If you choose this option then you must enter the Maximum Voltage Drop and Maximum Voltage Rise in the two corresponding fields within this dialog box.
  - Individual constraint per feeder. If you choose this option, then each feeder has a unique voltage drop / rise limit which is assigned on the Tie Open Point Optimisation page of each feeder.
5. If the option *Consider Boundary Constraints outside feeders* is set, the boundary constraints, applied on the boundaries "Reliability" settings are considered.
6. Choose the *ignore all constraints for...* option. You can use these options to optionally ignore constraints where the nominal voltage is above or below user-defined thresholds entered here. This can be useful for example to exclude all LV systems (say less than 1 kV) from the constraints identification process as it may be acceptable to have these systems outside the "normal" range.

## How to configure the Switching Options

The options in the Advanced page can generally be left on default values. The options are described as follows:

- Switches to be optimised. These options configure the switches / elements considered by the optimisation procedure.
  - All switches. All switches will participate in the optimisation.
  - Selected switches. Only the selected switch types will participate in the optimisation. For example, if "Circuit-Breaker" and "Load-Breaker-Switch" are ticked, then both circuit breakers and load breakers will be considered by the optimisation. The switch type is defined on the

switch element “Basic Data” page. Similar to Switch type, only the selected control types will participate in the optimisation. The control type is defined in switch element “Reliability” page in the “Sectionalising” field. Switches are considered in the optimisation only when its switch type AND the control type satisfies the selected settings.

- Assume each edge element is switchable. If selected, lines that do not have a switch can also be switchable (either out of service or in service).

### How to configure Reliability Options

If *Optimisation of Reliability Indices* is selected, the user may select between optimisation of SAIFI or EPNS indices on the Reliability page. Where:

- SAIFI (System Average Interruption Frequency Index) in units of [1/C/a], indicates how often the average customer experiences a sustained interruption in one year. Note that the number of customers at each load should be defined on the Reliability page.
- EPNS (Expected Power Not Supplied) is in units of [MW]. Multiplying EPNS by the study duration gives the expected energy not supplied.

Contingency definitions can be optionally considered for Busbar / terminals, Lines / Cables, and Transformers.

If *Cost Optimisation (Losses + Reliability)* is selected, *Costs for Losses* and *Interruption costs per customer* should be defined, as these are used in the Objective Function calculation to determine the network configuration that optimises both Losses and Reliability.

### How to configure Explore Meshes Options

- Maximum number of outer loops. This option controls the maximum number of outer loops which is the total number of times the optimisation procedure will be repeated when searching for an optimal solution.
- Maximum change in system losses. This option determines the threshold above which a change in open point is considered. If the reduction in losses is below this threshold, the iteration will stop.
- Constraint Priority options can be selected for the relevant constraints. For example, consider the following scenario:
  - The TOPO calculation is to consider Global Thermal constraints, with the *Max. thermal loading of components* set to 100 %, and Global Voltage Constraints with a Lower limit of 0.90 p.u.
  - The constraint priorities *for loading constraint* is set to 1, and *for voltage lower limit* is set to 3.
  - In the current configuration, a line is loaded to 102 % of rating.
  - Shifting the open point causes the voltage at a terminal on an adjacent feeder to decrease 5 % below 0.90 p.u. (i.e. 0.855 p.u.).
  - As a result of the priorities, the thermal loading deviation will be “penalised” to a greater extent than the voltage deviation, and the open point will change, despite the resultant voltage deviation.

### How to configure Simulated Annealing and Genetic Algorithm Options

Please refer to Section 42.11 for details about this method and its input parameter.

### How to configure Scenario/Time-Sweep Options

The Tie Open Point Optimisation can optionally be executed for different Operation Scenarios or certain Study Times.

---

**Note:** It is important to note that when different Operation Scenarios or points in time are selected, that the **switching state** in the network has to be the same for all investigated scenarios or study times. If the switching states differ, the results are not reliable. The operating points of consumers and generation units can of course vary.

---

The following options are available:

- **Not considered:** This option only considers the network in its current state (actual study time, currently active scenario) for the optimisation.
- **Scenarios:** Different Operation Scenarios can be selected which are considered in the optimisation:
  - *Consideration of the “No scenario case”:* The network in its current state (active study case) is considered as well. Be aware, that when the considered study case has an active operation scenario, the scenario will be deactivated. A weight factor can be entered for this case to define its significance in the optimisation.
  - *Definition of scenarios:* All scenarios that should be considered in the optimisation can be defined in this table. With a right-click into the open area of the table, new rows can be appended to assign *Operation Scenarios (IntScenario)*. Scenarios can optionally be ignored and given a certain weight factor to define their significance in the optimisation.

**Note:** The entered *weight factors* are used to give certain scenarios a higher significance over others. The weight factors are included in the objective function of the optimisation to determine the optimal tie open point.

---

- **Study Times:** The calculation time period for the optimisation can be defined according to the following options:
  - *Complete day:* a specific user-defined day can be selected as simulation time period. The day is chosen in the corresponding field.
  - *Complete month:* a specific user-defined month can be selected as simulation time period. The month is chosen in the corresponding field.
  - *Complete year:* a specific user-defined year can be selected as simulation time period. The year is chosen in the corresponding field.
  - *User defined calculation times:* specific user-defined calculation times can be added for the simulation (using the time format *dd:mm:yyyy hh:mm:ss*). The simulation will only be executed at the defined calculation times. The user also has the option to ignore certain points in time by activating the associated option.

**Note:** The optimisation will only be executed for the defined calculation times, while all other points in time are not considered. Therefore, the user has to choose the calculation times carefully to get reliable results.

---

- *User defined time range:* a customisable time period can be chosen for the simulation by defining the *Begin* and *End* time points (using the time format *dd:mm:yyyy hh:mm:ss*).
- A fixed *Step size* is used for the simulation. The step size is defined by the *Step* (integer value) and the *Unit*. The *Unit* can be selected from the corresponding drop-down list (*Minutes, Hours, Days, Months or Years*).

## How to configure Output Options

There are several different output and result options available:

- *Report objective value after each iteration:* If selected, the objective value of the selected objective function is printed into the output window after each iteration.
- The result objects *Before Optimisation* and *After Optimisation* are linked and can be selected here.

- Optional: Enable the *Report* flag. This control allows you to turn on the automatic printing of an ASCII report to the output window.
- Change the *Saving of solution* option. The three options are as follows:
  - *Change existing network*: This is the default option. The TOPO tool modifies the base network model. Note that if a variation but no operation scenario is active, the changes will be implemented in the variation. If an operation scenario is active, the results will be saved in it.
  - *Record to operation scenario*: If you choose this option a selection control appears and you can choose an existing operation scenario to save the results of the Optimisation procedure to. Alternatively, you can leave the selection empty and *PowerFactory* automatically activates a new Operation Scenario called “Tie Open Point Optimisation Results”. Any changes made to the network as a result of the optimisation procedure are stored within this operation scenario. You can revert to the original network by disabling the scenario.
  - *Record to results only*: This option does not save the optimisation results. The results are only available in the corresponding result files.
- Optional: Enable the *Record multiple solutions* option to get not only the one optimal tie open point, but several suboptimal solutions according on the entered *Number of solutions*.

---

**Note:** The option *Record multiple solutions* is only available when

- the calculation method *Generic algorithm* is used,
- no additional operation scenarios or study times are considered (page *Scenarios/Time-Sweep*)
- and the saving of solution option *Record to operation scenario* is used.

For each additional solution that could be found by the genetic algorithm (pay attention to warning messages in the output window), a new Operation Scenario will be created.

---

### 42.6.3 Results of the Tie Open Point Optimisation

The newly found tie open point is saved in a new or existing operation scenario.

---

**Note:** The optimisation determines always **one** solution for optimal tie open points except when multiple solutions are recorded. This is also the case when several operation scenarios or study times are investigated.

---

All results are recorded in the dedicated results file before and after the optimisation.

#### 42.6.3.1 Graphical Representation

The newly found tie open points will be automatically applied to the network model depending on the results handling setting. In addition, the determined switching operations are highlighted in the schematic diagram.

#### 42.6.3.2 Reports

After executing a *Tie Open Point Optimisation* calculation, reports can be generated using the *Tie Open Point Optimisation Report* icon  in the *Distribution Network Optimisation* toolbox.

This brings up the Report Generation command (*ComReport*), where the available reports are listed and can be selected. Reports can be generated as separate documents or combined into one. By

default, reports are generated as PDFs and presented in *PowerFactory*'s inbuilt PDF viewer, but it is also possible to export reports from *PowerFactory* in various different formats.

For more information about reports and the Report Generation command, see Chapter 19: Reporting and Visualising Results, Section 19.5. It should be noted that reports viewed internally in *PowerFactory* are stored in the graphics board of the study case, even after they have been closed, until they are actively deleted by the user.

The following calculation-specific reports are available:

- **Tie Open Point Optimisation**

This report shows the results of the analysis. The switching actions in the network are shown, together with the feeder results before and after the optimisation

- **Tie Open Point Optimisation Settings**

This report shows relevant parameters from the *Tie Open Point Optimisation* command.

---

**Note:** When multiple operation scenarios or study times are investigated, the displayed results can be interpreted as follows:

- Result variables that represent extreme values, like for example the minimum voltage, represent the minimum voltage that occurred in the feeder in any of the investigated operation scenarios/study times.
  - Other results variables, like for example the losses, represent the **average** losses that occurred in the feeder considering all of the investigated operation scenarios/study times.
- 

#### 42.6.3.3 Load Tie Open Point Optimisation Results

The *Load Tie Open Point Optimisation Results* functionality () allows the user to load the results from the optimisation into the network. The loaded result file (*ElmRes*) can be selected under the option *Results after optimisation*. To apply the new tie open points, the user has the following options:

- *Change existing network:* This option loads the tie open points from the selected result file into the current network. If an operation scenario is active, the results will be recorded by it.
  - *Create new scenario:* This option creates a new operation scenario with the name "Loaded TOPO Results" which includes the new tie open points from the selected result file.
  - *Apply changes to multiple scenarios:* This option loads the tie open points from the selected result file to all scenarios that are selected in the corresponding table.
- 

**Note:** The original switching state of each scenario, the changes should be applied to, has to be equal to the scenario(s) that were used in the optimisation. The selected result file (after optimisation) is only saving switching states that were changed during the optimisation.

---

## 42.7 Phase Balance Optimisation

Distribution networks are generally designed to support asymmetric loads and feed in. This fact leads to asymmetric load flows, which can be calculated with *PowerFactory* using the unbalanced Load Flow Calculation. The asymmetric load flows result in higher loadings and losses in single phases and transformer windings and are therefore not welcome. In networks with a high number of asymmetric loads and/or elements with less than 3 phases, the *Phase Balance Optimisation*  offers a possibility to distribute the connected phases of asymmetric elements in a way to minimise the power unbalance

in the network. The feature works on radially operated feeders using one of two possible algorithms to satisfy the chosen objective function.

The following sections describe the objective functions of the Phase Balance Optimisation, the implemented algorithms and the possibilities regarding the solution and its output.

### 42.7.1 Objective functions

The optimisation algorithm may be executed for two different objective functions. The minimised quantity is in both cases the power unbalance  $s$ . It is defined for branch elements as follows:

Let  $N$  be the number of phases, and let

$$\hat{S} = \frac{1}{N} \sum_{i=1}^N S_i$$

be the average complex power (at one end) of a branch element, where  $S_i, i = 1, \dots, N$  are the complex powers on phases  $1, \dots, N$ . Let

$$\bar{S} = \frac{1}{N} \sum_{i=1}^N |S_i|$$

be the average of the absolute values of the powers on the different phases. Then the power unbalance factor  $s_b$  for the branch element  $b$  is defined as

$$s_b := (1/\bar{S}) \max_{i=1, \dots, N} \{|S_i - \hat{S}|\}.$$

The user can choose between the following two objective functions for the optimisation:

- **Minimise average power unbalance:** This function takes into account the power unbalance of all  $M$  branch elements, which are part of the analysed feeder. The average power unbalance is defined as:

$$\bar{s} = \frac{1}{M} \sum_{b=1}^M s_b.$$

- **Minimise power unbalance at feeding point:** This function permutes the connection of the feeders elements to get a minimum power unbalance at the feeding point (branch element, where feeder 'starts'), regardless the unbalance of the rest of the feeder elements.

### 42.7.2 Methods

To achieve the minimisation of power unbalance, three different algorithms are available to choose from:

- Large loads and generators first
- Simulated annealing
- Genetic Algorithm

All three methods have their advantages. The Large loads and generators first method is easier to understand and to configure, and leads in most cases to very good solutions. The Simulated annealing and genetic algorithm methods are more theoretical regarding the configuration, but due to the random approach can find solutions for networks in which the power balance is difficult to achieve.

#### Large loads and generators first:

This algorithm iterates over all loads and generators in order of their apparent power, starting with the largest load or generator. For each load or generator, it will permute the connections of the load or generator or their supplying branch elements. After calculating the objective function for all possible connections, it will choose the best connection for this load or generator.

**Settings:**

The behaviour of the algorithm can be controlled with the following two settings:

The setting 'Disconnect loads and generators at beginning' will disconnect all loads and generators before the algorithm starts to iterate, and in each iteration, the actual load or generator will be connected to the grid in the best way for this iteration step. If this setting is not chosen, the actual load or generator will be just reconnected.

At each iteration, the algorithm will evaluate several modifications. The threshold in the frame 'Acceptance of change' determines the minimal improvement required for such a modification. If a modification does not lead to an improvement larger than this threshold, it is not applied to the solution.

**Simulated annealing**

This algorithm as well as the input parameters are described in detail in Section [42.11](#).

### 42.7.3 Elements considered

The set of elements, whose connections are permuted during the optimisation can be parameterised by the following settings:

- **Allow phase permutation:** defines, which types of elements are considered by the algorithm. At least one box has to be checked.
- **Elements with fixed phases:** a selection of various elements may be chosen, which are excluded from the optimisation.

### 42.7.4 Representation of solution

The optimised connections of the affected elements can be applied to the network by choosing one of the two possible options:

The preferable option is to use 'Create new Variation', where all connection changes will be stored into a new variation. They can be undone by deactivating the newly created variation. This option is also advantageous if the impact of the optimisation has to be analysed or if different settings have to be compared between each other.

As alternative, the changes may be set directly in the network without creating a new variation. Select this option only if surely intended!

After the execution of the optimisation the result boxes show the unbalance factors of power, current (for branch elements) and voltage (nodes). By calculating another load flow, the result boxes are reset to the normally shown variables.

### 42.7.5 Output

The Phase Balance Optimisation tool displays by default some information in the output window. The internal and effective objective function value before and after the optimisation are printed (the internal objective value may differ from the effective objective value due to an approximation made to achieve the high performance). The number of modified elements, differentiated for the element types, are listed, too. Additional information may be displayed by checking the following settings in the Output page of the ComBalance-dialog:

- **Output changed elements:** lists after execution of the optimisation all elements, which were changed including the affected phases.

- **Report objective value after each iteration:** displays the internal objective value for every iteration in the output window.

## 42.8 Voltage Profile Optimisation

The Voltage Profile Optimisation (VPO) command (*ComVoltplan*) is used to optimise distribution transformer taps over the expected range of network load and generation conditions. It can be selected from the Distribution Network Optimisation Tools, as shown in Figure 42.1.2.

The VPO calculation considers two scenarios:

- A maximum demand/minimum generation scenario, or “Consumption Case”.
- A minimum demand/maximum generation scenario, or “Production Case”.

For the representation of the transformer and the supplied LV network the user can choose between two possible modelling solutions:

- **MV Load:** This requires that loads be represented as medium voltage (MV) loads (*ElmLodmv*). MV load elements include transformer type data and LV network parameters, as illustrated in Figure 42.8.1a.
- **Transformer in a secondary substation:** This requires that a MV/LV-transformer is located within a secondary substation (*ElmTrfstat*). All supplied loads and generators that are modelled to represent the LV network will be considered. The more detailed modelling gives the user more accurate results for the voltage drop/rise over the transformer. An example is shown in Figure 42.8.1b.

To show terminal colouring based on maximum/minimum LV grid voltages, select *View* → *Diagram Colouring* from the main menu (or select the *Diagram Colouring* icon). Under 3. Other, select *Results* → *Voltages / Loading*. Click on *Colour Settings*, go to the second page of the *Voltages / Loading* page, and select *Consider LV grid voltages for colouring*. In the example below, the minimum voltage is below the lower limit and the maximum voltage is above the upper limit (the limits set in the colouring options) and the terminal therefore shows two colours.

The load and generation scaling factors used in the tap optimisation calculation override the values specified on the “Load Generation Scaling” page of the *Load Flow Calculation*, but the *Load Flow Calculation* settings remain unchanged.

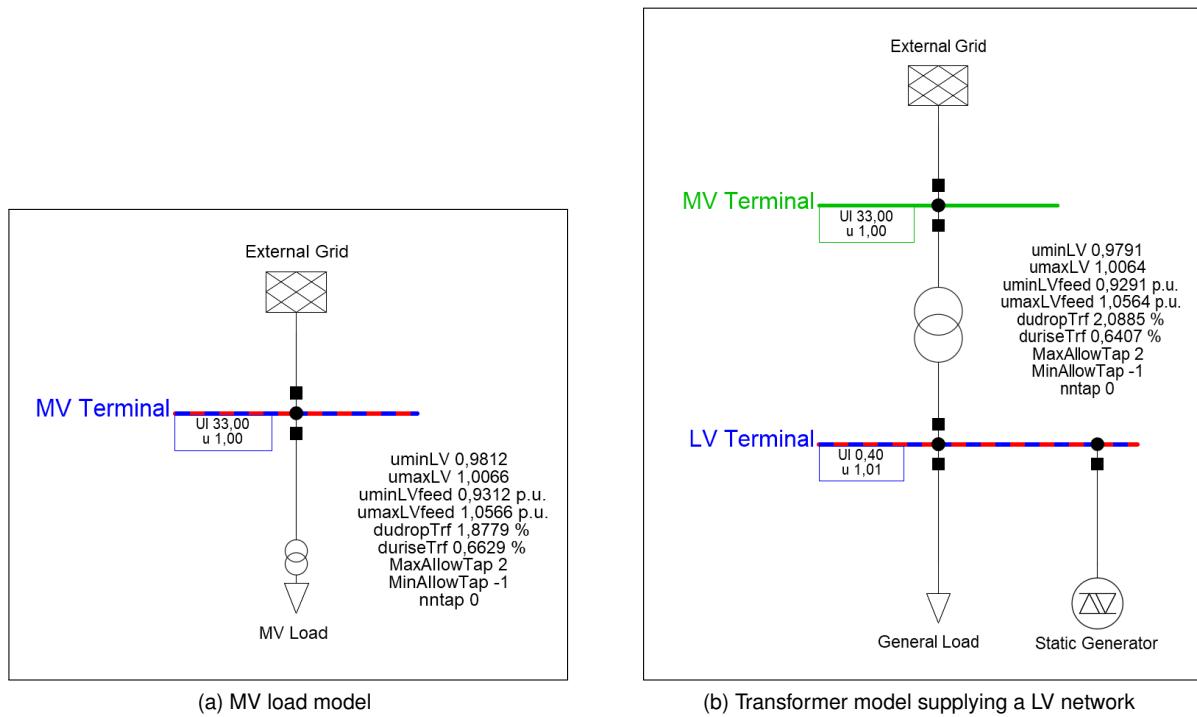


Figure 42.8.1: Use of different transformer representations for the Voltage Profile Optimisation

**Note:** In the MV load model, the transformer tap changer is represented on the LV side of the transformer.

### 42.8.1 Optimisation Procedure

The optimisation procedure can be summarised as follows:

1. If *Distribution Transformer Tap Limits* are specified by the user, the tap range of transformers will be limited within *Min. allowed tap position* and *Max. allowed tap position*. This is illustrated in Figure 42.8.2, where a transformer with seven tap positions is limited to taps “-1” to “2”, which limits the transformer voltage rise to 7% and voltage drop to -5%. The height of each bar is determined by the voltage rise and voltage drop across the transformer in the production and consumption cases, respectively.

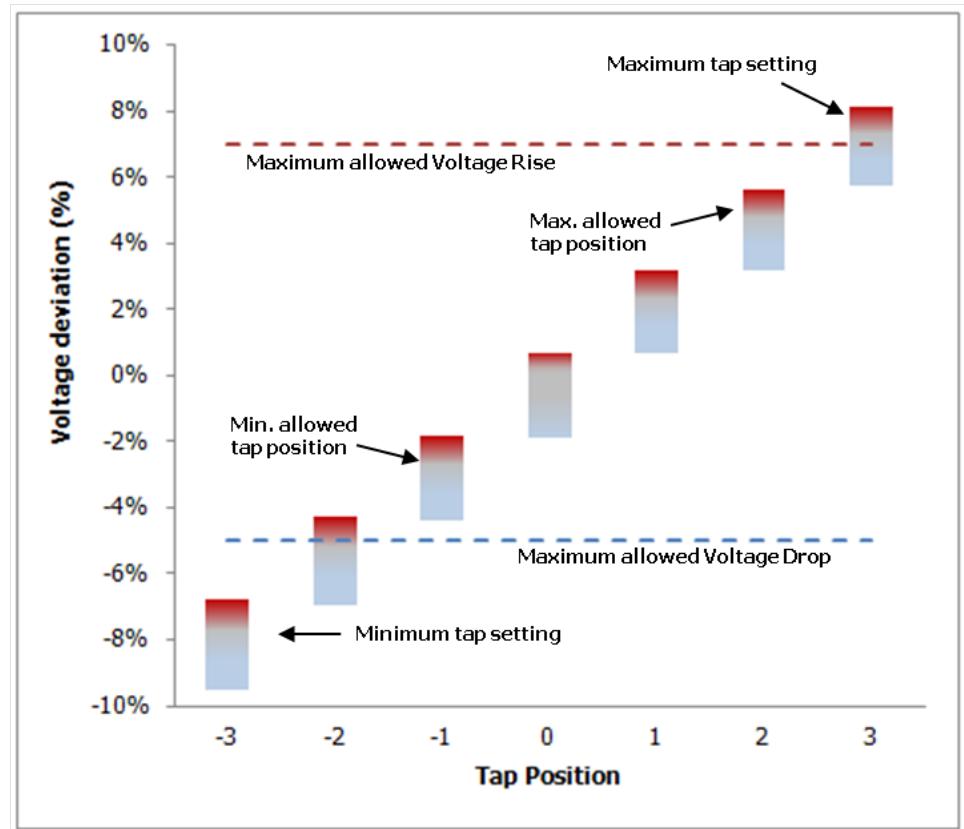


Figure 42.8.2: Distribution transformer tap limits

2. The *Upper tap limit* and *Lower tap limit* are calculated based on settings that will keep the range of expected LV grid voltages within the *Upper voltage limit* and *Lower voltage limit*. This is illustrated in Figure 42.8.3, where the limits are set to between 0.92 p.u. and 1.10 p.u. In cases where only *Production case* or *Consumption case* is set, only the corresponding voltages within the LV grid will be considered.
3. Both tap positions “0” and “1” would be acceptable, and maintain transformer voltage drop and LV grid voltages within acceptable limits. The optimisation routine selects the optimal tap position based on the objective function defined in the command. Figure 42.8.3 shows an example for the objective function *Maximisation of generation*. The lower tap limit (position “0” in Figure 42.8.3) is selected in order to minimise the voltage rise.

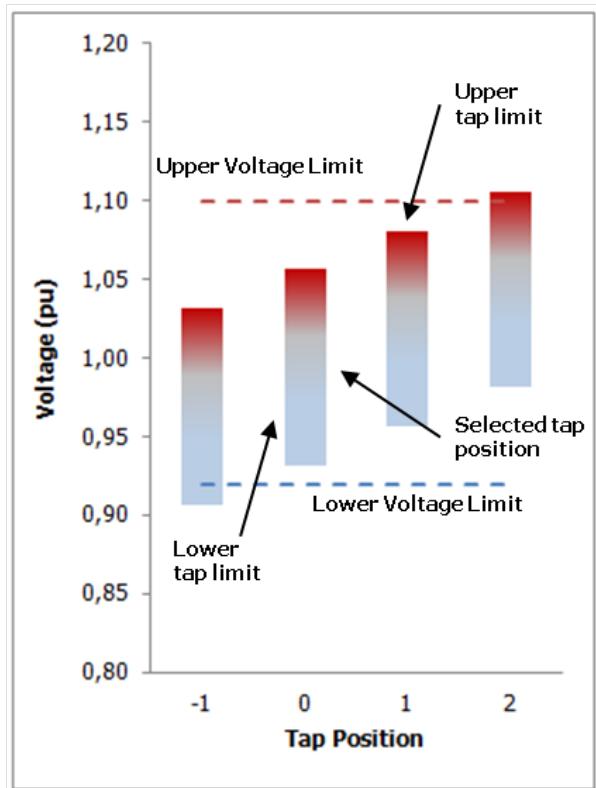


Figure 42.8.3: Voltage limits for LV grids with the objective function *Maximisation of generation* selected

The possible scenarios for optimisation are summarised as follows:

1. There is a single tap position that will satisfy both LV grid lower and upper voltage limits -> this tap is selected.
2. There are multiple tap positions that will satisfy both LV grid lower and upper voltage limits -> the selected objective function defines which tap position is selected.
3. There is no tap position at all, that will satisfy either the lower nor upper voltage limits of the LV grid -> *PowerFactory* will try to secure the LV grid violations of either the minimum (for *Maximisation of consumption*) or maximum (for *Maximisation of generation*) voltage limit.
4. There are tap positions that will satisfy the LV grid upper voltage limit, but all of them violate the lower voltage limit -> the highest tap position that will not violate the upper voltage limit is selected.
5. There are tap positions that will satisfy the LV grid lower voltage limit, but all of them violate the upper voltage limit -> the lowest tap position that will not violate the lower voltage limit is selected.

Note that *Distribution Transformer Tap Limits*, if specified on the *Advanced Options* page, take precedence over the *Upper voltage limit* and *Lower voltage limit* specified on the *Basic Options* page. This means that if distribution tap limits are considered, a tap range is first determined which respects these drop/rise limits over the MV/LV transformer (HV to LV side of the MV/LV transformer). Afterwards, an optimal tap position which obeys the voltage limits (in the LV feeder) for the selected *Calculated cases* is sought within this range.

## 42.8.2 Basic Options Page

- Calculation mode
  - **Optimisation:** the tap of the distribution transformers will be set to the optimal position within the given limits.

- **Verification:** the tap position of the distribution transformers will remain as-is. The algorithm checks whether this setting exceeds the given limits. This could be used to verify whether the given tap positions in a network are still valid after changes within the LV grid.
- Calculation cases
  - **Consumption- and production case simultaneously:** the consumption- and production case will be calculated and shown in the results. In addition, a tap position which conforms in both cases is selected.
  - **Consumption case only:** the consumption case will be calculated with the *Voltage limits for LV grids*. Hence, the tap position will be optimised for the highest possible voltage within the limit. A typical application would be a LV grid with high power consumption and no generation units.
  - **Production case only:** the production case will be calculated with the *Voltage limits for LV grids*. Hence, the tap position will be optimised for the lowest possible voltage within the limit. A typical application would be a LV grid with a high proportion of photovoltaic.
- Objective function (only available for *Calculation method: Optimisation*)
  - **Maximisation of generation:** if multiple tap positions of the distribution transformer meet the given limits for the consumption- and/or production case, the result with the lowest voltage level will be used.
  - **Maximisation of consumption:** if multiple tap positions of the distribution transformer meet the given limits for the consumption- and/or production case, the result with the highest voltage level will be used.
- Voltage limits for LV grids
  - **Upper voltage limit:** upper limit that the LV grid must not exceed (e.g. 1.1 p.u.).
  - **Lower voltage limit:** lower limit that the LV grid must not fall below (e.g. 0.9 p.u.).
- Consumption case (not available for *Production case only*)
  - **Load scaling factor:** percentage load scaling for the calculation of the consumption case (e.g. 100 %).
  - **Generation scaling factor:** percentage generation scaling for the calculation of the consumption case (e.g. 0 %).
- Production case (not available for *Consumption case only*)
  - **Load scaling factor:** percentage load scaling for the calculation of the production case (e.g. 25 %).
  - **Generation scaling factor:** percentage generation scaling for the calculation of the production case (e.g. 100 %).
- **Load Flow calculation:** a reference to the Load Flow command used by the optimisation algorithm. A copy is made of the command meaning that any changes made do not affect the settings of the original Load Flow command.

### 42.8.3 Output Page

In cases where *Calculation method* “Consumption- and production case simultaneously” is used, the following option is available:

- Shown results
  - **Consumption case:** the results for the consumption case are shown.
  - **Production case:** the results for the production case are shown.

#### 42.8.4 Advanced Options Page

##### Distribution Transformer Tap Limits

Transformer *Maximum Allowed Voltage Rise* and *Maximum Allowed Voltage Drop* can be optionally specified. These limits restrict the feasible range of taps in the optimisation procedure.

#### 42.8.5 Results of Voltage Profile Optimisation

The result of the Voltage Profile Optimisation can be shown as a tabular or ASCII report, or as a voltage profile plot.

##### Tabular and ASCII report

The tabular or ASCII reports, which show the recommended tap settings, including details of MV loads with critical voltage drop or rise can be accessed after the Voltage Profile Optimisation has been calculated. This is done by clicking on “Reports Voltage Profile Optimisation” (📋). An example of the Optimal Transformer Tap Positions section of the report is shown below in Figure 42.8.4 (results consistent with Figure 42.8.1 and the discussion in Section 42.8.1). In the case where only the production or consumption case are calculated, only the corresponding results will be available in the last columns (voltages).

Optimal transformer tap positions												
Studied elements	Transformer type data			Allowed		Opt.		Voltages				
	Ratio	Limits	Neut.	Add.v.	tap	/tap	range	position	Consumption case	HV	LV	Production case
	[1:x]	[Min,Max]			[%]		[Min,Max]		[p.u.]	[p.u.]	[p.u.]	[p.u.]
Terminal_External Grid												
MV Load	1.000	[-3, 3]	0	2.5	[-1, 2]		0	1.000 0.981	0.931	1.000 1.007	1.057	

Figure 42.8.4: Voltage profile results

The recommended tap settings are also available on the Flexible Data page of MV loads under the *Voltage Profile Optimisation* calculation parameter “c:nntap”. To update the network model with the recommended tap settings, the user may either manually adjust MV load tap positions, or click the *Update Database* icon on the main toolbar (⌚), and update the case with the calculated distribution transformer taps.

##### Voltage profile diagram

To display a plot of the resultant profile for one feeder for the consumption case, production case or both, select the *Voltage Profile Plot* icon (〽️). Figure 42.8.5 shows an example plot for the consumption case only:

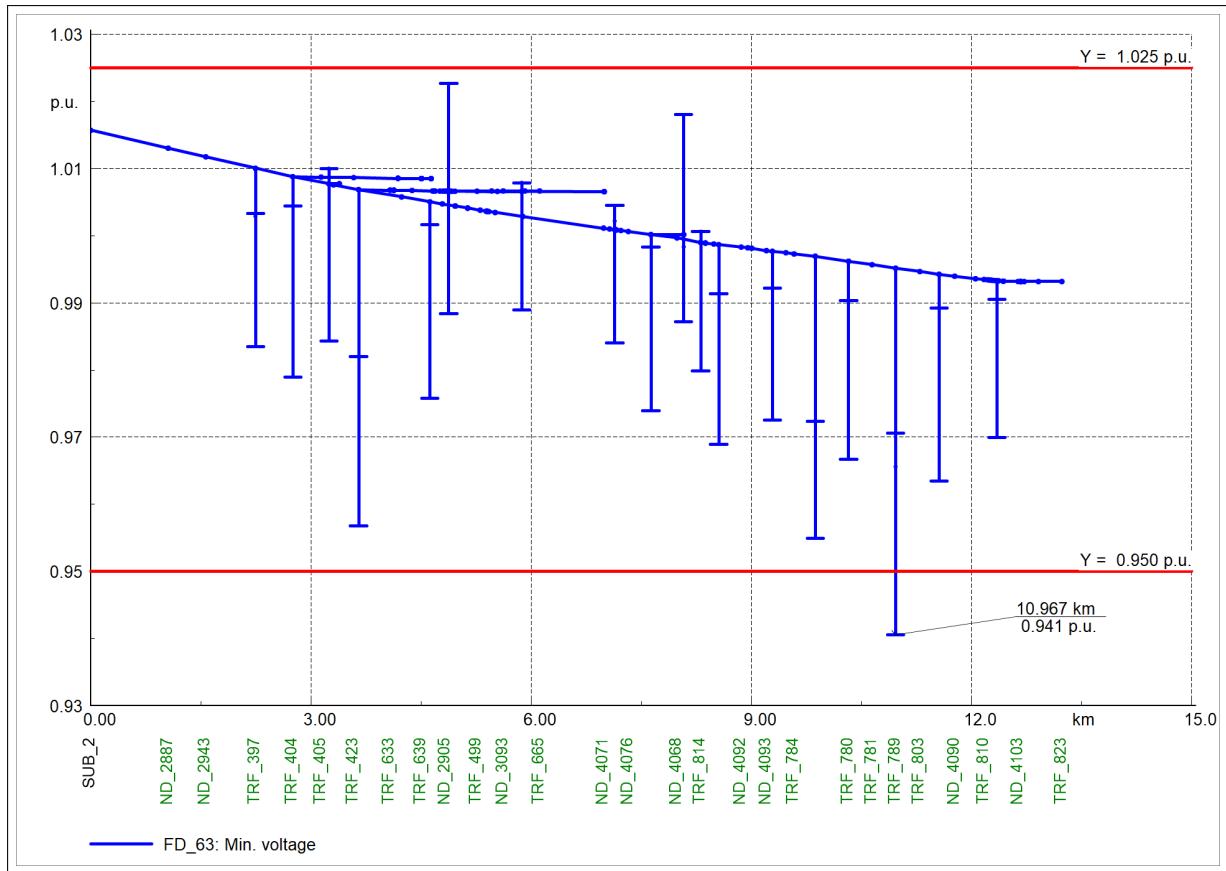


Figure 42.8.5: Voltage profile plot for the consumption case

## 42.9 Optimal Equipment Placement

The *Optimal Equipment Placement* function (⊕) allows the user to place storage units and voltage regulators at optimal locations in the network. In addition, existing storage units and voltage regulators can be optimised.

In distribution networks, the operator is often faced with temporarily high loading of equipment or voltages that lie outside the required voltage band. Due to increasing decentralised generation units and more variable loads, fluctuations in networks increase. Besides traditional network expansion, there is need for more smart equipment such as batteries and voltage regulators. The *Optimal Equipment Placement* function can therefore help to determine the right installation location of storage models and voltage regulators.

There are several optimisation options available:

- Placement of new storage models with a pre-defined capacity
- Placement of new storage models with optimised capacity
- Optimise existing storage models
- Placement of new pre-defined voltage regulators
- Optimise tap positions of existing voltage regulators and transformers

*Optimal Equipment Placement* can be executed using a balanced AC load flow for a user-defined time range and step size. The tool offers a wide range of configuration possibilities for the equipment to be used, as well as constraints to match specific requirements.

The objective function of the *Optimal Equipment Placement* is to minimise the total installation and operation costs of the equipment under consideration, taking pre-defined constraints into account.

### 42.9.1 Optimal Equipment Placement Configuration

The *Optimal Equipment Placement* command  can be found in the *Distribution Network Optimisation* toolbar. More information about the available settings can be found in the following subsections.

#### 42.9.1.1 Basic Options

##### Equipment type:

There are two equipment types available that can be used in the *Optimal Equipment Placement*:

- Voltage regulators
- Storage models

The options seen on the different pages of the command can change depending on the equipment type used.

##### Feeder:

The optimisation will always be executed for one defined feeder. The necessary selection can be done by clicking on the  button and then press *Select...*. From the selection browser the desired feeder (*ElmFeeder*) for the investigation can be selected. Detailed information on feeders and how to define them, can be found in Section 15.6.

##### Load Flow:

For each time step that is defined on the *Time Sweep* page in the command (see Section 42.9.1.5), a balanced AC Load Flow calculation is executed. The command used can be edited by pressing the Edit button . Detailed information on the options of the *Load Flow Calculation* command can be found in Section 25.3.

##### Saving of solution:

There are two options for saving the solution of the optimisation (newly placed/optimised equipment):

- *Change existing network*: If this option is selected, all changes will be directly recorded in the current network configuration. If a variation with a recording expansion stage is present, the changes will be recorded by this variation.

---

**Note:** It is **important** to be aware of the current network configuration and where the changes will be recorded.

---

- *Create new variation*: This option will create a new variation with a dedicated *Variation name*. The activation time of the expansion stage will be set to the first time step of the simulation.

#### 42.9.1.2 Equipment

The input parameters and tables on the *Equipment* page depend on the *Equipment type* that is selected for the optimisation. Here the user is able to define the available equipment for the *Optimal Equipment Placement*.

**Voltage regulators:**

The definition of new voltage regulators is done via the table for **Available Equipment**. The following configurations can be defined:

- **Type (TypVoltreg):** The type of the *Step-Voltage Regulator* contains all electrical parameters. Available types can be selected from the *Equipment Type Library* or can be newly created (). More information on the different parameters of the type can be found in the corresponding technical reference document.
- **Voltage Regulator element (ElmVoltreg):** For each row of the table a new *Step-voltage Regulator* is created, which will use the defined type (*TypVoltreg*) in the same row. The following options can be specified for the *Optimal Equipment Placement*:
  - The parameter *Position of tap 1* cannot be changed for new equipment. For existing *Step-Voltage Regulators* this option can be either activated to optimise the tap position or deactivated to leave the tap changer at its current position during the optimisation.
  - The *Penalty costs per Tap deviation* is used by the solver to find the cheapest solution. This is only a virtual parameter which does not contain actual operation costs.

**Note:** It is recommended to enter a value  $> 0$  so that the solver can find the cheapest available solution. Without penalty costs for a tap change, the solver can theoretically change the tap position within the limits as often as possible, which is usually not wanted.

- 
- The *Max. loading* of the voltage regulator can be considered in the optimisation. The user has the option to turn this constraint off, or to define the maximum loading as hard or soft constraint.
  - *Installation costs* for one element.
  - *Maintenance costs* for one element per year. The resulting maintenance costs depend on the *Planning Period* of the optimisation.
  - **Min. number:** Defines the minimum number of voltage regulators for each row that have to be placed.
  - **Max. number:** Defines the maximum number voltage regulators for each row that are allowed to be placed.

For the definition of new voltage regulators, the button **Add voltage regulator** can be used. Alternatively, the user can insert or append new rows by right-clicking into the table and choose the corresponding option.

With the buttons **Show chosen types** and **Show voltage regulators**, the user can see a list of objects that are defined in the table. The displayed table selection gives the user a good overview of the parameters of the available equipment.

By activating the option **Max. total number of placed voltage regulators** the user can define the maximum number of voltage regulators within the feeder being investigated. Note that existing voltage regulators are included in the total number.

The **Planning period** for the *Optimal Equipment Placement* can be defined in years. With this parameter, the user is able to determine the total investment costs of the equipment for the defined planning period.

- *Installation Costs* are defined in the *Voltage Regulator (ElmVoltreg)* and are only considered once to install the new equipment.
- *Maintenance Costs* are defined per year in the *Voltage Regulator (ElmVoltreg)* and are scaled according to the planning period.

The total costs over the whole *Planning period* can be found in the result file *Summary* (see Section 42.9.1.7).

In addition to the installation of new equipment, the *Optimal Equipment Placement* is also able to optimise the operation of **Existing Equipment**. For this, the option **Optimise tap positions of existing voltage regulators and transformers** is activated. For each existing transformer and voltage regulator element within the feeder being investigated, the following options on the *Optimal Equipment Placement* page can be configured:

- The parameter *Position of tap 1* can be either activated to optimise the tap position of the element or deactivated to leave the tap changer at its current position.
- The *Penalty costs per Tap deviation* is used by the solver to find the cheapest solution during the optimisation. This is only a virtual parameter which does not contain actual operation costs.

---

**Note:** It is recommended to enter a value  $> 0$  so that the solver can find the cheapest available solution. Without penalty costs for a tap change, the solver can theoretically change the tap position within the limits as often as possible, which is usually not wanted.

---

- The *Max. loading* can also be considered. The user has the option to turn this constraint off, or to define the maximum loading as hard or soft constraint.

On the **Advanced tab** of the *Equipment* page, the strategy for the optimisation of the tap position can be specified. For the **Minimisation of voltage regulator and transformer tap change** the user has two options:

- *Based on current/neutral position*: Here the preferred tap position is the neutral tap position (for new elements) or the current tap position (for existing elements) of the equipment, which can be specified on the load flow page in each element.
- *Based on optimised value of previous time point*: Here the objective is to minimise the overall tap changes, so the preferred tap position is the one of the previous time point.

### Storage models:

The definition of new storage models is done via the table for **Available Equipment**. The *Storage model (ElmStorage)* will be linked to the *Static Generator (ElmGenstat)* for each row to restrict the generated and consumed energy by the charging/discharging power of the generator. The following configurations can be defined:

- **Storage model (ElmStorage):** For each row of the table a new storage model is created. It contains all energy related parameters and constraints of the storage:
  - *Storage type*: There are three storage types available. *Battery*, *Hydropower* and *General Storage*.
  - *Capacity*: Represents the size of the storage model. For the types *Battery* and *General Storage* the capacity can only be entered in MWh. For *Hydropower* the capacity can be entered in MWh or as water volumes which are then converted with the head of water.
  - *Operational energy limits*: Minimum and maximum operational limits can be defined in p.u. values. The limits can be considered as soft constraints.
  - *Energy constraint at study period end*: Option to define the energy of the storage at the end of the time period being investigated. The user can either define no constraint, a *min. energy* that the storage should have or decide that the energy at study period end is *equal to the energy at study period start*. It is possible to define the energy constraint at study period end as soft constraint.
  - *Energy at study period start*: It can be defined by a *fixed value* or it can be optimised by the *Optimal Equipment Placement*.
  - The *Self-discharge* of the storage model can be defined in p.u. per hour.
- **Static Generator (ElmGenstat):** For each row of the table a new generator is created. The generator is linked to the storage model of the same row and contains the power related parameters and constraints as well as the costs:

- *Efficiency*: For the generation and consumption mode an *Efficiency curve (IntEffcurve)* can be selected. The user can define the rated efficiency in p.u. at different active power dispatches. More details on the different options for the *piecewise linearisation of the efficiency curve* can be found in Section 40.6.1.
  - The parameter *Optimise active power* cannot be changed for new equipment. For existing storage models this option can be activated to optimise the power dispatch.
  - *Active Power Operational Limits*: The minimum power limit is used for the consumption mode and defines the maximum charging power, while the maximum power limit is used for the generation mode and defines the maximum discharging power.
  - The *Generation costs* contain *Penalty costs* that represent the costs per MWh in generation mode as well as *Fixed costs* per hour.
  - The *Consumption costs* represent the costs per MWh in consumption mode.
  - The *General costs* contain the *Installation costs* of the entire storage model as well as the *Maintenance costs* per year.
- **Min. number**: Defines the minimum number of storage models for each row that have to be placed.
  - **Max. number**: Defines the maximum number storage models for each row that are allowed to be placed.
  - **Optimise capacity**: Option to optimise the capacity of the selected storage model (row). If this option is selected, then the min. and max. capacity columns in the table will define the limits for the optimal size. The capacity costs will be considered in the optimisation process. The objective of the *Optimal Equipment Placement* is to find a large enough storage model to reduce the constraint violations, that is as cheap as possible.
  - **Capacity costs**: Costs per MWh for the storage model that is to be optimised.
  - **Min. capacity** of the storage model that is to be optimised.
  - **Max. capacity** of the storage model that is to be optimised.

For the definition of new storage models, the button **Add storage** can be used. Alternatively, the user can insert or append new rows by right-clicking into the table and choose the corresponding option.

With the buttons **Show chosen storage models** and **Show chosen generators**, the user can see a list of objects that are defined in the table. The displayed table selection gives the user a good overview of the parameters of the available equipment.

By activating the option **Max. total number of placed storage models** the user can define the maximum number of storage models within the feeder being investigated. Note that existing storage models are included in the total number.

The **Planning period** for the *Optimal Equipment Placement* can be defined in years. With this parameter, the user is able to determine the total investment costs of the equipment for the defined planning period.

- *Installation Costs* are defined in the *Generator (ElmGenstat)* and are only considered once to install the new equipment.
- *Maintenance Costs* are defined per year in the *Generator (ElmGenstat)* and are scaled according to the planning period.
- *Operation Costs* are defined in the *Generator (ElmGenstat)* and are calculated for the time period being investigated. These costs contain the generation as well as the consumption costs and are scaled according to the planning period.

The total costs over the whole *Planning period* can be found in the result file *Summary* (see Section 42.9.1.7).

In addition to the installation of new equipment, the *Optimal Equipment Placement* tool is also able to optimise the operation of **Existing equipment**. For this, the option **Optimise existing storage models** is activated. To optimise the power dispatch of existing storage models within the feeder being investigated, the following options in the *Generator (ElmGenstat)* on the *Optimal Equipment Placement* page can be configured:

- *Optimise active power*: If this option is activated, the active power dispatch of the storage model will be optimised. If the option is deactivated, the power dispatch will not be changed.
- *Active Power Operational Limits*: The minimum power limit is used for the consumption mode and defines the maximum charging power, while the maximum power limit is used for the generation mode and defines the maximum discharging power.
- *Separate consumption mode*: If this option is activated, separate consumption power limits can be defined. The max. power corresponds then to the maximum charging power of the storage model.

---

**Note:** When the *Separate consumption mode* is active, the minimum operational limits will only be considered for values > 0, because smaller values are already defined by the max. limit of the other mode.

---

- *Generation and Consumption costs* per MWh and hour.

#### 42.9.1.3 Locations

For the optimisation, a feeder has to be defined on the *Basic Options* page. Only locations within this feeder are considered for the placement of new equipment. The following options are available to define the candidate locations:

- *All terminals*
- *Only busbars*
- A user-defined *Terminal selection*. By clicking on the (▼) button and choose *Select...*, the *Candidate terminals* from the selection browser can be selected.

Equipment that is to be optimised also needs to be located within the specified feeder.

#### 42.9.1.4 Constraints

Within the feeder being investigated, the following **General Constraints** are considered:

- *Consider thermal constraints (loading)* of all components within the feeder:
  - If the option *Global constraint for all components* is selected, then a *Maximum thermal loading* has to be defined.
  - If the option *Individual constraint per component* is selected, then the maximum loading constraint can be specified within each component on the *Optimal Equipment Placement* page.
  - The *Constraint type* can either be regarded as hard or soft constraint.

**Note:** For the equipment being investigated, individual loading constraints can also be considered for candidate voltage regulators.

---

- *Consider voltage limits* of all terminals within the feeder:
  - If the option *Global constraint for all terminals* is selected, then a *Lower* and *Upper voltage limit* has to be defined.
  - If the option *Individual constraint per terminals* is selected, then the voltage limits can be specified within each terminal on the *Optimal Equipment Placement* page.

- The *Constraint type* can either be regarded as hard or soft constraint.
- *Soft constraints*
  - The constraints for the *Optimal Equipment Placement* can be either hard or soft constraints. When soft constraints are defined, violations are allowed, but cause additional costs. These costs are defined by the *Penalty factor for soft constraints*.

**Note:** The costs for soft constraints are only virtual to find the most cost-effective solution. They are considered in the optimisation but not added to the total costs in the reported results.

---

Furthermore, **Advanced Constraints** can be considered:

- *Ignore all constraints for nominal voltage...:*
  - If this option is activated, the user can define voltage thresholds, above/below which all constraints are ignored.
  - The parameter < (below) defines the lower voltage threshold and > (above) defines the upper voltage threshold.
- *Constraints outside feeder:*
  - In addition to considering constraints within the feeder being investigated, voltage and loading limits outside the feeder can also be observed. This allows the user to take into account violations outside the feeder which have been caused by the new or optimised equipment.
  - If the option *In supplying substation* is selected, an internal algorithm will find the substation that is supplying the feeder. The constraints will then be considered for all elements within the supplying substation.
  - If the option *For selection* is selected, the user is able to define elements outside the feeder where constraints should be observed. By clicking on the (▼) button and choose *Select...*, elements from the selection browser can be selected.

**Note:** When constraints outside feeder are considered, the optimisation does not only take into account violations caused by the new equipment, but also tries to solve existing constraint violations. Since new equipment may not have a big influence there, these can cause the problem to become infeasible. The constraints outside the feeder should therefore be chosen carefully.

---

**Note:** For the equipment being investigated, element-specific constraints are considered as well (see Section [42.9.1.2](#)).

---

#### 42.9.1.5 Time Sweep

The optimisation and placing of new equipment is always done for a user-defined time period. The following options can be selected:

- *Calculation time period:*
  - *Complete day:* a specific user-defined day can be selected as simulation time period. The day is chosen in the corresponding field.
  - *Complete month:* a specific user-defined month can be selected as simulation time period. The month is chosen in the corresponding field.
  - *Complete year:* a specific user-defined year can be selected as simulation time period. The year is chosen in the corresponding field.
  - *User defined calculation times:* specific user-defined calculation times can be added for the simulation (using the time format *dd:mm:yyyy hh:mm:ss*). The simulation will only be executed at the defined calculation times. The user also has the option to ignore certain points in time by activating the associated option.

**Note:** The optimisation will only be executed for the defined calculation times, while all other points in time are not considered. Therefore, the user has to choose the calculation times carefully to get reliable results.

- *User defined time range*: a customisable time period can be chosen for the simulation by defining the *Begin* and *End* time points (using the time format *dd:mm:yyyy hh:mm:ss*).
- A fixed **Step size** is used for the simulation. The step size is defined by the *Step* (integer value) and the *Unit*. The *Unit* can be selected from the corresponding drop-down list (*Minutes, Hours, Days, Months or Years*).

#### 42.9.1.6 Algorithm

##### Solver:

*PowerFactory* offers different possibilities for solving the mixed-integer linear program (MILP) problem of the *Optimal Equipment Placement* function. The optimisation supports both internal solvers (*lp\_solver*, *cbc solver*) as well as commercial external solvers like IBM CPLEX and GUROBI. More detailed information on the different solvers can be found in Section 40.3.7.2.

- The “Cbc”-solver is selected by default and is recommended if no commercial solvers are available.

One of the *Basic parameters* that can be defined is the *Time limit*. If activated, the user can set a time limit for the solving time of the solver to avoid very long run times.

---

**Note:** If a very low time limit is set, the solver might not find the optimal solution or may find no solution at all.

---

The *Solver parameters configuration* for each MILP-solver is usually set to pre-configured default values. If the user wants to change the input parameters of the selected solver, the option *Advanced* can be selected to modify the values.

##### Effectiveness:

To solve violated constraints within the feeder, the sensitivities/effectiveness of the equipment (generator, transformer, voltage-regulator) on the constraints has to be calculated. The option *Thresholds for power flow/voltage constraints* allows the user to specify the *Min. generator/transformer effectiveness* to be considered. If the effectiveness of an equipment on a constraint is below the defined limit, then it is not considered in solving the constraint.

To *Ignore violated constraints without effective controls*, the corresponding option can be activated. If this option is not active, a warning before the optimisation will inform the user that violated constraints within the feeder can't be resolved.

#### 42.9.1.7 Results/Output

The *Optimal Equipment Placement* calculation writes three result files during execution. The main result file is linked under the parameter **Results**. The linked result file contains three sub result files:

- “Optimal Element Placement (summary):” This result file records the overall results of the calculation including installation, maintenance and operation costs of the placed equipment as well as equipment-specific result variables.
- “Optimal Element Placement (before optimisation):” This result file saves the load flow results of each time step before the optimisation and is similar to the results of a Quasi-Dynamic Simulation.

- “Optimal Element Placement (after optimisation):” This result file saves the load flow results of each time step after the placement and optimisation of equipment.
- 

**Note:** Additional variables that should be recorded by the *Optimal Equipment Placement* can be defined by the user via the *Edit* button (=+) from the toolbar. A detailed description on defining new variable can be found in Section 28.3.1. By default, the result files before/after the optimisation record all variables that are constrained or controlled by the optimisation. The variables recorded in the summary result file are pre-defined and cannot be changed.

---

The **Output** options correspond to the degree of information that is shown on the *Output Window* for the time period calculations before and after the optimisation. The user can choose between no output, a *Short summary* or *Detailed load flow information*.

---

**Note:** When no output is selected, a summary report of the optimised equipment will still be written to the *Output Window*.

---

Additionally, an *Output message in case of constraint violations after the optimisation* is written to the *Output Window*. The **Max. acceptable error** for constraints defines a threshold against which the optimal solution should be verified. When the solution is verified and violations are detected, there will be sets of violation elements reported in the output window.

## 42.9.2 Results of the Optimal Equipment Placement

The newly placed or optimised equipment is either saved within a new variation or integrated into the existing network (see Section 42.9.1.1).

The optimal power dispatch of storage units and the optimal tap position of voltage regulators that are obtained by the *Optimal Equipment Placement* are stored within an additional result file “*Optimal controls*” which is located in the active study case in a folder named “*Result Characteristics Opt. Placement*”. The characteristics from this result file are created in the project *Operational Library* → *Characteristics*→ “*Placed equipment*” and are linked to the corresponding equipment.

### 42.9.2.1 Output Window

For a first evaluation of the *Optimal Equipment Placement* results, the report in the *Output Window* can be inspected. As well as the installed equipment and its configuration, the installation, maintenance and operation costs are listed.

### 42.9.2.2 Graphical Representation

Newly placed equipment will be automatically built into the network model. In addition, the new equipment is highlighted in any schematic diagram.

### 42.9.2.3 Flexible Data Page

After the execution of the *Optimal Equipment Placement*, there are additional statistical variables available for certain elements, in the variable selection on the flexible data page. For those, the addition “statistics” can be found in the column *Type* in the table of the available variables.

- **Statistics: Currents, Voltages and Powers** These variables contain statistical values for control variables such as average, minimum, maximum and variance over the complete simulation time. The statistical values are available for all variables that are recorded in the result files (see Section 42.9.1.7).
- **Statistics: Calculation Parameter** These variables contain statistical values such as redispatch costs and amounts or loadings. They are available for the optimised equipment.

Other variables, that have not been recorded, show the results of the last time step of the simulation. Note: The summary result file also contains result parameters from the entire simulation period.

#### 42.9.2.4 Plots

The *Create curve plot* button ( ) can be found in the “Distribution Network Optimisation” toolbar. Similar to the Quasi-Dynamic Simulation curve plots over time, duration curves and energy plots are available for the *Optimal Equipment Placement*.

#### 42.9.2.5 Tabular Reports

The *Optimal Equipment Placement Report* ( ) is also found in the “Distribution Network Optimisation” toolbar. Built-in reports are available for further analysing the *Optimal Equipment Placement* results:

- *Optimisation results*
  - **Optimal solution:** This report lists the control elements (storage models, transformers/voltage regulators) and their power dispatch/tap position before and after the optimisation. The costs of the equipment are also listed.
  - **Storage report:** This report is specifically made to summarise the storage models of the optimisation. Therefore different values such as storage energy at the start/end of the optimisation as well as the generated/consumed energy and costs are listed.
  - The *Study time* for the *Optimisation results* reports can be set to a single time point or to summarise all investigated time points.
- *Time sweep load flow results*
  - **Loading ranges:** This report lists the loading of all elements for the selected time range. A threshold can also be defined, so as to only display elements which exceed this loading limit.
  - **Voltage ranges:** This report lists the min. and max. voltage of all terminals for the selected time range. Thresholds can also be defined, so as to only display elements which exceed these lower or upper voltage limits.
  - **Non-convergent cases:** This report lists the points in time where the load flow is not converging before or after the optimisation.
  - The results can either be reported *Before* or *After* the optimisation.
  - The *Time Range* can be set to the *Complete* or a *User defined time range*.

#### 42.9.2.6 Load Optimal Equipment Placement Results

The *Load Time Sweep Load Flow Results* functionality ( ) allows the user to reload the results from the selected result file (ElmRes) to the network elements. The user can choose between the results *Before* and *After* the optimisation. The results are loaded for a selected point in time of the available *Time Range*. This enables the user to investigate critical hours in the network model.

### 42.9.3 Troubleshooting

Some model setups can lead to an infeasible problem for the solver, which makes it impossible for the *Optimal Equipment Placement* function to find a solution. In the following sections, some factors that can cause the infeasibility are described:

#### 42.9.3.1 Solver Selection

The solver selection usually only has an influence on the performance. When a **Time Limit** is specified, it might be the case that the included solvers are not able to find a solution within the defined limited time. Therefore, it might be beneficial to increase the time limit. For large and complex problems in particular, the commercial solvers should be able to find optimal solutions much faster.

#### 42.9.3.2 Soft Constraints

The optimisation can become infeasible if a set (hard) constraint cannot be kept or resolved with the given controls. With more constraints and limited control options this problem increases. To avoid this, soft constraints can be used instead of hard constraints. Soft constraints are allowed to be violated and therefore the solver can find a solution more easily. To influence this effect, the price for violating soft constraints can be set in the *Optimal Equipment Placement* command on the page *Constraints* (see Section 42.9.1.4).

---

**Note:** Thermal and voltage constraints within the feeder are defined in the command. Additional constraints of the equipment used also have to be considered.

---

#### 42.9.3.3 Network Analysis

In general, network analysis before executing the *Optimal Equipment Placement* is always beneficial. By executing a *Quasi-Dynamic Simulation* before the optimisation, the feeder can be investigated for the specified time period. This can be especially beneficial if hard constraints are to be used. The following list may help the user to investigate whether the equipment has the required capacity:

- *Storage models:*
  - *Capacity:* Can the used storage model store enough energy for the time period under investigation? It might help to optimise the capacity of new equipment.
  - *Energy Limits:* The operational limits of the storage model as well as a required energy at the start and end of the simulation can limit the capacity used.
  - *Charging/Discharging Power:* In the generator linked to the storage model, the max. charging and discharging power are defined. It might be beneficial to check whether the operational limits allow the storage model to charge/discharge with the required power.
- *Voltage regulators/Transformers:*
  - *Available tap positions:* Are there enough tap positions available to solve the voltage violations?
  - *Min. and max. voltage in feeder:* In some feeders it might happen that the voltage of the first node of the feeder is under the min. voltage limit or over the max. voltage limit. In that or similar cases, no voltage regulator within the feeder can solve the problem.
- *Locations:*
  - Are suitable locations for the placement of equipment available to solve the current constraint violations?

## 42.10 Optimal Capacitor Placement

Optimal Capacitor Placement (OCP) is an automatic algorithm that minimises the cost of losses and voltage constraints (optional) in a distribution network by proposing the installation of new capacitors at terminals along the selected feeder/s. The optimal size and type of capacitor is selected from a list of available capacitors entered by the user. The algorithm also considers the annual cost of such capacitors and only proposes new capacitors for installation when the reduction of energy loss and voltage constraint costs exceeds the annual cost of the capacitor (investment, maintenance, insurance etc).

The OCP functions can be found in the Distribution Network Optimisation toolbar and are as follows:

- The main Optimal Capacitor Placement command is started with the *Calculate Optimal Capacitor Placement* icon (). The command and the various user-defined options are described in detail in Sections 47.3.1 to 47.3.3.
- After a successful optimisation, the list of nodes (terminals) where capacitors are proposed for installation can be accessed by selecting the *Show nodes with New Capacitors* icon ().
- Following a successful OCP, the list of proposed capacitors can be accessed with the *Show New Capacitors* icon ().
- The *Remove previous solution* icon () deletes the results (removes all placed capacitors) from a previous OCP routine.
- To list all results from the OCP in a ASCII text report written to the output window use the *Optimal Capacitor Placement Reports* icon (). The report also displays the original system losses and voltage constraint costs and such costs after the installation of the proposed capacitors.

### 42.10.1 OCP Objective Function

The OCP optimisation algorithm minimises the total annual network cost. This is the sum of the cost of grid losses, the cost of installed capacitors, and optionally the fictitious penalty cost of voltage violations:

$$TotalCosts = CLosses + \sum_{i=1}^m (CCap_i) + \sum_{i=1}^n (CVoltViol_i) \quad (42.12)$$

Where:

- $CLosses$  is the annual cost of grid losses (i.e. including the grid losses, not only the feeder/s for which the optimal capacitor placement is performed). Essentially, this is the  $I^2R$  loss of all elements in the network.
- $CCap_i$  is the annual cost of a capacitor (investment, maintenance, insurance), as entered by the user in the list of possible capacitors.  $m$  is the total number of installed capacitors.
- $CVoltViol_i$  corresponds to a fictitious cost used to penalise a bus (terminal) voltage violation.  $n$  is the total number of feeder terminals with voltage violations.

Note that if the OCP is not able to reduce the Total Costs by installation of a capacitor/s, the following message will be reported:

*Costs can not be reduced with the given "Available Capacitors"*

### Evaluating the Voltage Violation Cost

As there is no 'real' cost for a voltage violation, if the user wants to consider voltage violations as part of the OCP algorithm, they must assign a 'fictitious' cost for such violations. The voltage violation cost is calculated based on the user specified voltage limits and penalty factors. The voltage limits are defined in the *Basic Options* page of the OCP command dialog ('vmin' and 'vmax' parameters, see Section 47.3.1: Basic Options Page). The penalty factors are defined in the *Advanced Options* page of the same command ('weight' and 'weight2' fields, see Section 47.3.3: Advanced Options Page). The penalty values are applied for voltages inside the admissible voltage band (parameter 'weight': Penalty Factor 1) and for voltages outside the admissible band (parameter 'weight2': Penalty Factor 2).

There are two possible situations for a terminal voltage and the calculation for the fictitious voltage violation cost is slightly different for each situation. The two situations are explained as follows:

1. In situation one, the voltage  $U$  of a terminal is within the allowed voltage band (between  $vmax$  and  $vmin$ ) but deviates from the nominal voltage of 1 p.u. The penalty cost is calculated as:

$$CVoltViol = w1 \cdot \Delta U \quad (42.13)$$

where:

$\Delta U$  is the absolute deviation from the nominal voltage in p.u. ( $\Delta U = |U - U_n|$ ).

$w1$  is the penalty factor (parameter 'weight') inside the admissible voltage band in \$/% from the *Advanced Options* page.

2. For situation two, the voltage  $U$  is outside the allowed voltage band (greater than  $vmax$  or less than  $vmin$ ) and the penalty cost is calculated as:

$U > U_n + \Delta U_{max}$ , if voltage is higher than max. limit:

$$CVoltViol = w2 \cdot (\Delta U - \Delta U_{max}) + w1 \cdot \Delta U$$

or

$U < U_n - \Delta U_{min}$ , if voltage is lower than min. limit:

$$CVoltViol = w2 \cdot (\Delta U - \Delta U_{min}) + w1 \cdot \Delta U$$

where

- $\Delta U$  is the absolute deviation from the nominal voltage  $U_n$  in p.u.
- $U_n + \Delta U_{max}$  is the higher voltage limit in p.u.
- $U_n - \Delta U_{min}$  is the lower voltage limit in p.u.
- $w1$  is the penalty factor (parameter 'weight') for voltage inside the admissible voltage band in \$/% from the *Advanced Options* page.
- $w2$  is the penalty factor (parameter 'weight2') for voltage outside the admissible voltage band in \$/% from the *Advanced Options* page.

The algorithm can be summarised in as follows:

- If the voltages are **inside the admissible band** the penalty cost applied is equal to  $w1 \cdot \Delta U$
- If the voltages are **outside the admissible band** the penalty cost applied is equal to the penalty inside the band ( $w1 \cdot \Delta U$ ) plus the factor  $w2 \cdot (\Delta U - \Delta U_{lim})$ , with  $\Delta U_{lim}$  being either the maximum or the minimum limit value of the admissible band.

Figure 42.10.1 illustrates the concept of the voltage band violation cost.

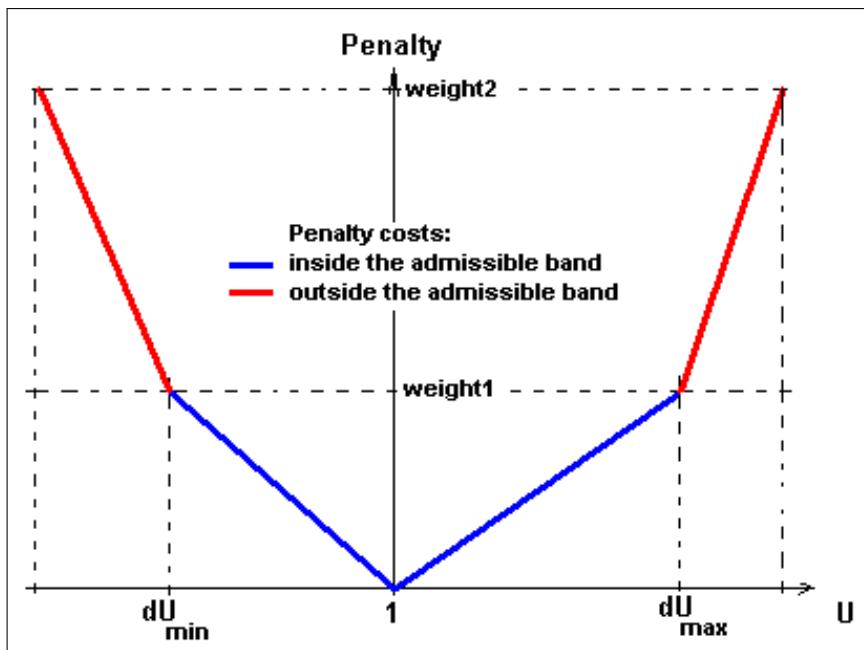


Figure 42.10.1: Fictitious cost assigned by voltage band violations

## 42.10.2 OCP Optimisation Procedure

To find the optimal configuration of capacitors, *PowerFactory* applies the following steps:

- First a sensitivity analysis determines the 'best' candidate terminal; This involves evaluating the impact on the total cost (Losses + Voltage Violations) by connecting the largest available capacitor from the user-defined list of capacitors to each target feeder terminal. At this stage the cost of the largest capacitor is *excluded*.
- Terminals are ranked in descending order of total cost reduction. The terminal that provides the largest cost reduction becomes the 'best' candidate terminal for a 'new' capacitor.
- The optimisation routine then evaluates the cost reduction at the candidate terminal using each available capacitor from the user-defined list *including* the cost of each capacitor. The 'best' capacitor is the one that reduces the cost the most when also considering the annual cost of that capacitor.
- Repeat step one but any terminals that have previously been selected as candidates for capacitor installation are not included in the ranking of candidate terminals. The algorithm stops when all terminals have had capacitors installed, or the installation of capacitors cannot reduce costs any further.

**Note:** If Load Characteristics are considered, then the above algorithm will be completed for every independent load state. See Section 42.10.5 for how the load states are determined.

## 42.10.3 Basic Options Page

### Feeder

Here the target feeder for the optimum capacitor placement is selected. The feeder is a special *PowerFactory* element that must be created by the user before it can be selected in this dialog (for information about feeders refer to Chapter 15: Grouping Objects 15.6 (Feeders)).

## Method

- Optimisation; This option calculates the optimal placement for capacitors using the methodology described in Section 42.10.2. The output of the analysis is printed to the output window and any new capacitors are connected to the target terminal/s if the 'Solution Action' - 'Install capacitors' is selected.
- Sensitivity Analysis; Performs the sensitivity analysis that ranks the candidate terminals according to their impact on the total loss cost excluding the capacitor cost. The output is presented in the output window. This option provides a quick indication of the most effective place for a single capacitor. No capacitors are installed if this option is selected.

## Network Representation

Here either a 'Balanced, positive sequence' or a 'Unbalanced' network representation can be selected. The Load-flow command referenced below these radio buttons is automatically adjusted to the correct calculation method based on this selection.

## Constraints

Here the voltage constraint limits (upper and lower) can be entered, along with a limitation for the 'Total Reactive Power of all Capacitors' that can be added by the Optimal Capacitor Placement tool. The total reactive power of all capacitors includes all existing capacitors along the feeder plus any more capacitors proposed by the optimisation tool.

---

**Note:** The voltage constraints are meaningless if penalty factors for deviations outside of the nominal range are not entered as discussed in detail in Section 42.10.1: OCP Objective Function.

---

## Energy Costs

The energy cost (\$/kWh) can be entered manually or taken from an External Grid. Note, if more than one External Grid exists in the network, the algorithm takes the first External Grid by database ID. The calculation of the cost of the network losses is as follows:

$$TC = MC \times 8760 \times L \quad (42.14)$$

where:

$TC$  is the total cost per annum in \$;

$MC$  is the energy cost of losses in \$/kWh; and

$L$  is the total losses in kW.

Note that if characteristics are applied to the loads and the analysis uses the option 'Consider Load Characteristics' (see Section 42.10.5), then the losses calculation becomes a summation over each time state considered.

---

**Note:** The default currency unit is USD. However, this can be changed on the *Project Settings* dialog (see Section 9.1.3).

---

### Solution Action

- Report only (do not modify network); The result of the optimisation is a report to the output window only, no modifications are made to the network model.
- Install capacitors (modify network). If this option is chosen, the capacitors that the optimisation proposes for the network will be automatically installed. However, note that the single line diagram is **not** automatically updated, only the network model database. To draw the installed capacitors in the SLD the option must be selected in the Advanced Options page (see Section 47.3.3). The placed capacitors can be also visualised on the Voltage Profile Plot of the Feeder, see (Viewing results on the Voltage Profile Plot) in Section 42.10.7.

### 42.10.4 Available Capacitors Page

On this page, the user defines the available capacitors for the OCP command. One capacitor is entered per row. To add a new capacitor, right-click within any cell and select the option 'Insert Rows', 'Append Rows' or 'Append n Rows'. The following fields are mandatory for each row:

- Ignored; If this option is checked, then the capacitor specified in this row will be ignored by the OCP command.
- Q per Step Mvar; Here the rated reactive power of the capacitor in Mvar per step is specified.
- Switchable; If this option is enabled then the algorithm can use a capacitor with multiple steps.
- Max. Step; If the 'Switchable' option is enabled, then this option specifies the maximum number of steps available to the optimisation algorithm. The maximum available reactive power is therefore Max. Step \* Q per Step Mvar.
- Technology; Specifies whether the capacitor is Three-phase or Single-phase.
- Cost; **Important**. This is the total cost of the capacitor bank per annum. This is a critical parameter for the OCP command as the capacitor will only be installed if the losses offset by its installation are greater than the annual cost of the capacitor.

---

**Note:** It is theoretically possible to force the installation of a particular capacitor at an optimal location on a feeder by defining a very low cost for the capacitor, and limiting the number of capacitors to say, one.

---

#### Available Capacitors

- Allow use of each capacitor multiple times; This is the default option and it means that every capacitor in the list can be used at more than one feeder terminal (multiple times).
- Use each capacitor only once; If this option is enabled then each capacitor can only be placed at one terminal along the target feeder.

#### Treatment of 3-phase capacitors

This option allows the specification of the 'technology' type for 3-phase capacitors. This option is only available when the 'Network Representation' is set to 'Unbalanced' in the Basic Options page.

### 42.10.5 Load Characteristics Page

If load characteristics are to be considered by the optimisation algorithm, then the option 'Consider Load Characteristics' should be enabled on this page.

## Load States

Two options are available:

1. 'Use existing Load States'; If this option is selected then the system load state that is active in the system (the load state observed as a result of a single load-flow at the current point in time) will be used as the load state for the optimisation algorithm. For example, if there is a 1 MW load with a active characteristic that gives the current load value of 0.6 MW, then the load used for the optimisation will be 0.6 MW, not 1 MW.
2. 'Create Load States'; If this option is selected then *PowerFactory* automatically discretises all load characteristics into a number of 'states' using a sophisticated algorithm. The algorithm iterates through every hour of the selected time period to determine the number of unique operating load states that exist. Every operating state is assigned a probability based on the number of times that it occurs and this probability is used to determine the cost of losses for each state.

## 42.10.6 Advanced Options Page

### Candidate Buses

- All terminals in feeder; If this option is selected, every terminal in the feeder is considered as a possible candidate for a 'new' capacitor.
- Percentage of terminals in feeder; Selecting this option and entering 'x' percent for the parameter means the optimisation algorithm will only consider 'x' percent of the feeder terminals as targets (candidates) for 'new' capacitors. The ranking of terminals is according to the Sensitivity Analysis as described in Section [42.10.2](#).

### Max. Number of Iterations

This parameter determines the maximum number of iterations of the optimisation algorithm before it automatically stops. As a maximum of one capacitor is placed per iteration, this can effectively limit the total number of capacitors that can be placed by the optimisation routine.

### Max. Execution Time

This parameter specifies the maximum time the optimisation routine can run before it is automatically interrupted.

### Penalty Factors for Voltage Deviation

- Factor for Deviation from 1 p.u (weight); This parameter is used to determine the total 'fictitious cost' for terminals deviating from 1 p.u. The cost is applied to each phase of the terminal. For example, if a three phase terminal voltage is measured at 0.95 p.u for each phase and the 'fictitious cost rate' is \$10,000/% then the total cost of this deviation is \$150,000 ( $5\% * \$10,000/\%$  \* 3).

---

**Note:** If no penalty costs are to be applied within the admissible band, this factor should be set to zero. If this value is greater than zero, the program will add costs to all terminals with voltage different than 1.0 p.u.

- 
- Additional Factor outside range [vmin, vmax] (weight2); This parameter can be used to apply an additional weighting factor to the first deviation factor when the terminal voltage falls outside the voltage limits defined on the 'Basic Options' page. The factor is cumulative, so using the previous example and a additional factor of 20,000/% with a vmin of 0.975, the fictitious cost becomes \$300,000 ( $5\% * \$10,000/\% + 2.5\% * \$20,000/\%$ ) \* 3.

---

**Note:** The values for the two voltage penalties 'weight' and 'weight2' should be carefully chosen because the target optimisation function is a sum of three objective functions (losses, capacitor cost and voltage deviation cost). If the voltage weights are too high, the algorithm might not consider the other two objectives. Likewise, if they are very low, the algorithm may not consider voltage violations at all.

---

### Print report after optimisation

The automatic printing of the optimisation results can be disabled by unchecking this option.

### Draw the installed capacitors

This option draw the installed capacitors in the Single Line Diagram when checked.

## 42.10.7 Results

Three OCP tool-bar buttons give access to the optimisation results.

### Show Nodes with New Capacitors

When pressing the *Show Nodes with New Capacitors* icon (), after a successful optimisation is complete, a list appears of all terminals where capacitors are proposed for installation.

### Show New Capacitors

Pressing the *Show New Capacitors* icon () shows a list of proposed new capacitors.

### Optimal Capacitor Placement Reports

The *Optimal Capacitor Placement Reports* icon () is used to generate a report with the results of the sensitivity analysis and the final optimisation procedure.

### Viewing results on the Voltage Profile Plot

Following a successful optimisation, the 'new' capacitors can be visualised on the voltage profile plot of the feeder. To enable this, navigate to the voltage profile plot display after the optimisation and click the rebuild  button. An example of such a plot showing the placed capacitors is shown in Figure 42.10.2.

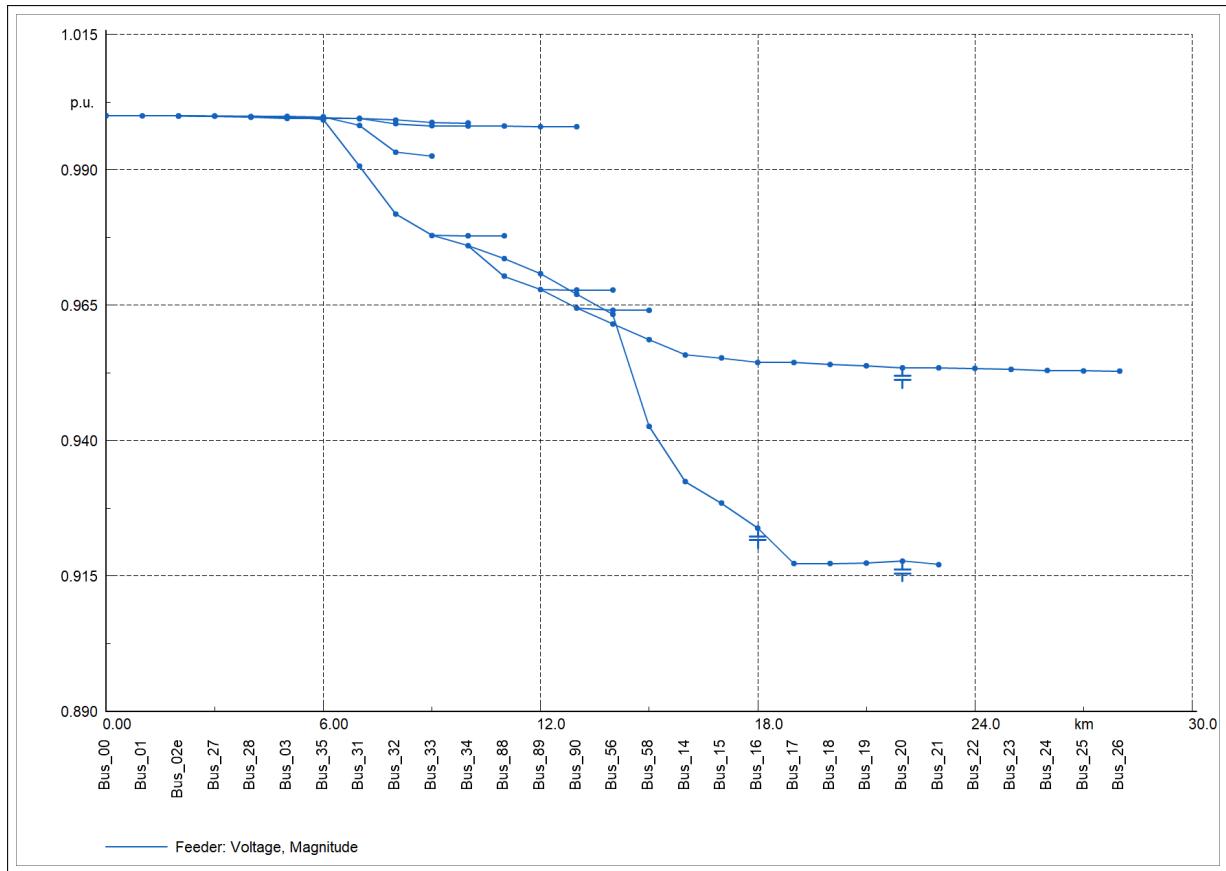


Figure 42.10.2: Voltage profile plot showing the new capacitors after an Optimal Capacitor Optimisation.

### Removing Capacitors Placed by the Optimal Capacitor Placement Routine

The capacitors placed by the OCP command can be removed at any time after the analysis has been completed by using the Remove previous solution icon (undo icon). This button is like an 'Undo' for the 'Optimal Capacitor Placement'.

## 42.11 Optimisation Algorithms

Genetic Algorithm and Simulated Annealing are well suited to solve the following optimisation problems.

- Objective function is not differentiable
- Only “discrete” states are allowed
- Possibility of having lots of local minima
- large amount of solutions within the state space

### 42.11.1 Genetic Algorithm

To illustrate how the Genetic Algorithm improves the optimisation process, the procedure for solving the Phase Balance Optimisation using this algorithm is shown below.

The starting point of this algorithm is the state space. This set contains all possible phase connection permutations within the network as a sequence, the so-called genotypes. These permutations are

coded numerically within the genotypes.

Figure 42.11.1 illustrates one genotype used for Phase Balance Optimisation. In this example, the numbers are the coded possibilities for phase permutations at the different cubicles within the network, whereas each vertical line represents one cubicle.

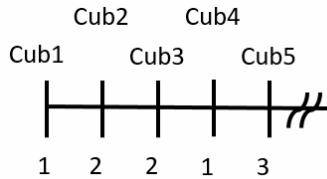


Figure 42.11.1: Sample genotype for Phase Balance Optimisation simulation

Out of the set of possible genotypes, a user-defined number of states is drawn and defined to be the population.

Starting from this population, the single genotypes are mutated and crossed over.

Mutation means the replacement of single code numbers, in this example phase permutations. A possible mutation is shown in Figure 42.11.2.

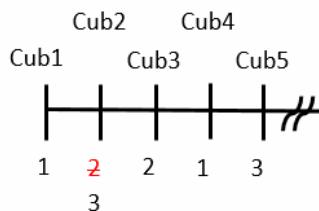


Figure 42.11.2: Sample genotype mutation for Phase Balance Optimisation simulation

Crossover stands for the exchange of sequences of code numbers (genes) between two genotypes. These two steps are executed in every iteration.

In each iteration, the objective function is calculated for each genotype within the population. In this example, the objective function would be the power unbalance. The genotype representing the minimum of the objective function is stored globally and gives the resulting optimum at the end of the optimisation.

#### **Settings:**

the algorithm stops if the 'Maximum number of iterations' is reached or the objective value is less than the defined value. If the latter is set to zero, the algorithm always stops after the maximum number of iterations.

The population settings define how many genotypes will be considered.

The mutation rate defines the portion out of the population, for which a mutation will be executed.

The number of mutation points defines the number of mutations, executed within one genotype.

The number of crossover points defines the lengths of the sequence to be replaced during cross over process.

### 42.11.2 Simulated Annealing

Simulated annealing is a stochastic optimisation method, which reconnects the grid randomly, and during a cool down of the 'system' will reach a good solution. During the execution of the algorithm, a so called temperature  $T_n$  is tracked, which reduces the longer the algorithm lasts.

At each iteration  $n$  of the algorithm, first a new proposal for possible solutions is generated. These solutions are then applied to the grid and the objective value  $v_p$  for this proposal is calculated.

If the objective  $v_p$  is better than the last objective value  $v_l$  (meaning  $v_p < v_l$ ), the algorithm accepts the proposal and will continue with the next iteration step.

If the objective  $v_p$  is worse than the last objective value  $v_l$ , with some probability  $p_n \in (0, 1)$  the algorithm still accepts the proposal. Note, that in this case  $v_p > v_l$ . The probability  $p_n$  decays as the temperature  $T_n$  decays:

$$p_n \propto \exp\{-(v_p - v_l)I_n\},$$

where  $I_n = 1/T_n$  is the inverse temperature.

Some stopping criteria can be given, to determine when the algorithm should stop iterating.

**Settings:** the algorithm stops if the 'Maximum number of iterations' is reached or the objective value is less than the defined value. If the latter is set to zero, the algorithm always stops after the maximum number of iterations.

The two settings in the frame for the inverse temperature  $I_n$  define how fast the variation between two iterations decreases.

# Chapter 43

# Outage Management

## 43.1 Introduction

With version 2017 of *PowerFactory*, a new methodology for modelling planned outages of elements of the network was introduced. The underlying principle is that in the active scenario, the network is intact. When the Planned Outage objects (*IntPlannedout*) are applied, the network changes are just held in memory rather than being applied in the scenario. Thus, resetting of outages becomes straightforward. All calculations will take account of applied changes such as switch positions and earths.

In addition, it is possible to add a range of associated actions to an outage, such as additional switch actions, transformer tapping and power transfer, which will subsequently be enacted whenever the outage is applied. A “record” mode is provided, which lets the user define events such as switch operations simply by carrying out the open and close actions via a diagram or filter.

An Outage Management toolbox is provided, to facilitate the handling of the Planned Outage objects.

## 43.2 Creating Planned Outages

By default, any new outages created will be objects of class *IntPlannedout*. Users wishing to create *IntOutage* objects will need to enable a project setting: on the Project Settings, Miscellaneous page select *Create IntOutage (obsolete)*.

### 43.2.1 Creating Planned Outages from Graphic or Network Model Manager

To create a Planned Outage object, the elements to be outaged are first selected via a graphic or from a Network Model Manager. Then right-click, *Operational Library* → *Planned Outage* → *New...* is used to create the object. The Start and End dates will by default be the start and end of the year to which the study case refers, and these can be edited. The selected element(s) form the contents of the outage. Two buttons are available to the user at this stage to visualise the outage on a graphic before it is applied: The *Outaged Comp.* button can be used to show the contents of the Planned Outage and the *Affected Comp.* button can be used to show all the affected components; this will include for example loads which are isolated by an outage.

### 43.2.2 Creating Planned Outages in Data Manager

Outages can also be created from the Outage folder in a Data Manager. The *New Object* icon is used and the Element Planned Outage (*IntPlannedout*) selected from the list. Elements can then be selected to populate the Outaged components list.

### 43.2.3 Recurrent Outages

It is possible to define a recurrence pattern for a planned outage. For example, a circuit may be switched out for work at weekends but put back in service for weekdays, with this pattern being followed throughout the outage duration.

#### 43.2.3.1 How to define a recurrent outage

The *Recurrent* box should be checked, and then the **Edit...** button is used to set up the recurrence pattern. The settings are used as follows:

The *Start* field and the two fields in the *Duration* panel are used to define the start time and the length (in days and hours) of the outage period that will be repeated. Note that the start time does not have to coincide with the main Start Time of the Planned Outage, as defined on the Basic Data page.

Then, in the *Recurrence* panel, the user defines how often this outage period is to be repeated.

- **Daily:** the outage period will be repeated every day or every n days, according to the number specified.
- **Weekly:** the user can configure the recurrence to be every week or every n weeks, and also chooses the days of the week when the outage should be applied.
- **Monthly:** the user can configure the recurrence to be every month or every n months; the days where the outage is applied are either individually specified or a rule such as “the second Monday of the month” is applied.
- **Yearly:** the user can configure the recurrence to be every year or every n years; the days where the outage is applied are either individually specified or a rule such as “the first Monday of April” is applied.

Figure 43.2.1 shows an example where a line is to be taken out of service for eight hours each weekend day, but left in service during the week.

#### 43.2.3.2 How recurrent outages are applied

When an outage which has recurrence defined is applied, both the main outage period and the actual period of recurring outage as defined in the *Recurrence Pattern* dialog are considered in relation to the study case time and date. The equipment will be considered as being out of service during the periods defined in the *Recurrence Pattern* dialog. Outside these periods, the equipment will be in service.

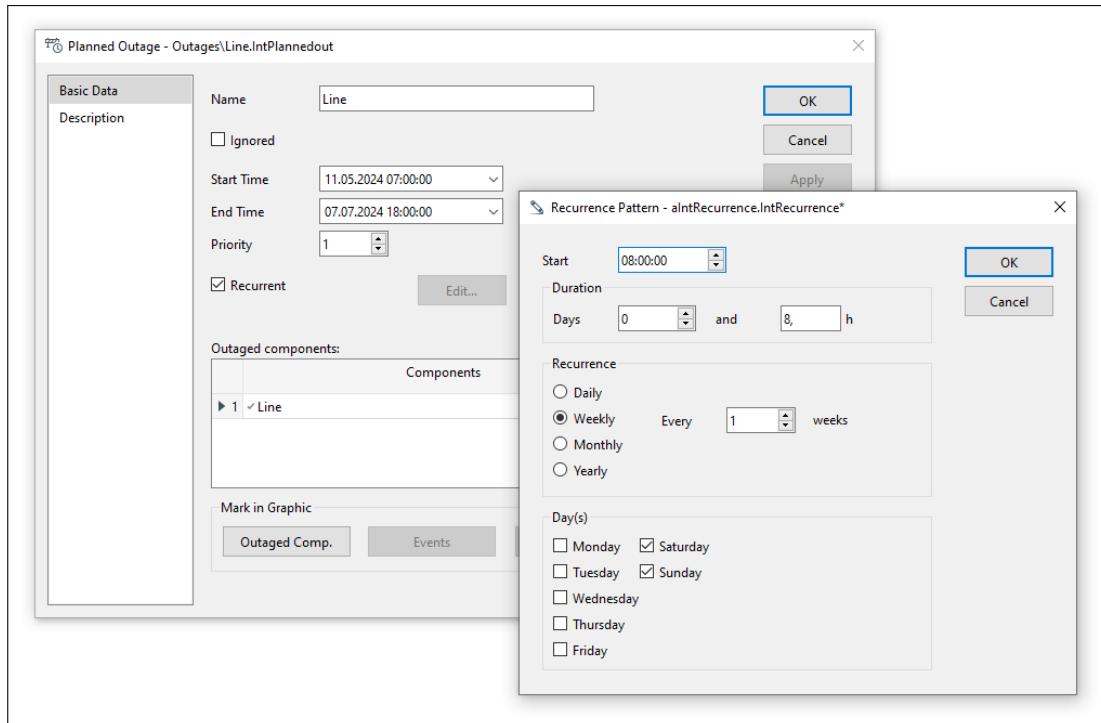


Figure 43.2.1: Planned Outage with recurrence

#### 43.2.4 Adding Additional Events to an Outage

A Planned Outage object can be modified to include events in addition to the outage of elements. Typically, switch operations might be executed to alter the configuration of a substation, but transformer tapping or Power Transfers can also be made. These are the events supported:

- *EvtSwitch* to open and close switches, or switch off and switch on elements
- *EvtTap* to change a tap setting
- *EvtTransfer* to transfer real and reactive power between load objects or between static generators.
- *EvtParam* to change other parameters such as generation or load set points

New events can be added to a particular Planned Outage object by editing the object and pressing the Start Rec. button. This will automatically apply the outage to start with, then the dialog should be closed and the user can return to the graphic or a filter of elements to start executing events which are to be recorded in the Planned Outage object.

It will be noted at this point that the Record Events button in the Outage Management toolbox (see Section 43.3.4) will now appear depressed.

When all required events have been recorded, the Stop Rec. button in the Planned Outage object should then be pressed.

When an outage is selected for recording, this is reported in the output window. The name of the outage will also appear in the status bar (at the bottom of the *PowerFactory* window) and stay there until recording is turned off again.

Alternatively, additional events can be added to an outage by editing the outage object, using the Events button to view the events, and using the *New Object* icon to add more events.

When additional events have been added to an outage, they are stored in a *IntEvtrel* folder within the Outage Object, called "Remedial Actions". They can also be deleted from here if required.

### 43.3 Handling Planned Outages using the Outage Management Toolbox

The Outage Management toolbox offers the following options to facilitate working with Planned Outages.

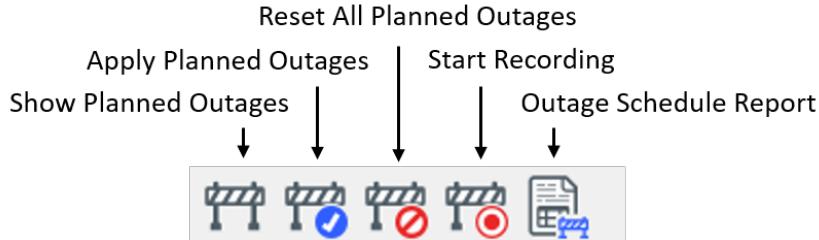


Figure 43.3.1: Outage Management functions

#### 43.3.1 Show Planned Outages

Pressing this button will bring up a filter of all the outage objects in the Outages folder of the Operational Library.

#### 43.3.2 Apply Planned Outages

Pressing this button will bring up a filter of all the outage objects in the Outages folder of the Operational Library which are applicable for the current study case time. All or some of these may be selected, then when OK is pressed these outages will be applied.

If multiple outages are selected, it is possible that there may be conflicts where more than one outage (and its associated events) refers to the same network element. By default, outages are applied sequentially according to the start date and time, but if outages have the same start date and time, the *Priority* flags on the outage objects are used to determine the sequence of application: the lower the number, the higher the priority, meaning that the outage with *Priority*=1 will be applied first, then the outage with *Priority*=2 and so on.

However, this rule can be reversed using a project setting on the Miscellaneous page of the project settings dialog, called *Application sequence of Planned Outages* (see Section 9.1.3.5). With the default setting of “Time-based application (time before priority)”, the application sequence rule is as described above. With the other setting, “Priority-based application (priority before time)”, the logic is that outages will be applied according to the *Priority* flags (*Priority*=1 first, and so on), and if some outages have the same priority, then these will be applied in an order based on the start date and time, the earliest ones being applied first.

In either case, if there are outages with both the same start date and time *and* the same priority, these will be applied in alphabetical order of name.

If some switch states are not applicable, a warning appears in the output window with the corresponding list.

#### 43.3.3 Reset All Planned Outages

This button will reset all the Planned Outages which have been applied.

#### **43.3.4 Start Recording**

If this button is pressed, a filter of possible outages in which to record events is presented. An outage is selected, and upon pressing OK the record mode is started, and additional events can be added to the chosen Planned Outage by making changes via a graphic or data filter. For example, a switch may be closed, a transformer tapped, a running arrangement selected or a load set point changed.

#### **43.3.5 Outage Schedule Report**

This button is used to generate a list of all Planned Outages for a defined time period. When the button is pressed, a dialog box is presented which can be used to set the start and end date of the report. “Detect” buttons give the option to have the dates set automatically according to the start and end dates of the earliest and latest Planned Outage objects. When the report is executed the list of Planned Outages is presented in tabular format, including a bar-chart to give an overview of the outage plan.

# Chapter 44

# Economic Analysis Tools

## 44.1 Introduction

This chapter presents the *PowerFactory* Economic Analysis Tools, which are dedicated functions for the economic assessment and profitability evaluation of network development plans.

The *Economic Analysis Tools* module is accessible through the *Change Toolbox* ▾ button on the Main Toolbar and contains the following tools:

- *Techno-Economical Calculation*, described in Section 44.2.
- *Techno-Economical Study Case Comparison*, described in Section 44.3.
- *Power Park Energy Analysis*, described in Section 44.4.

## 44.2 Techno-Economical Calculation

This section describes the *PowerFactory Techno-Economical Calculation* function. An overview, a technical background, a detailed explanation of the calculation command options as well as an explanatory example are provided.

A Techno-Economical Calculation can be used to evaluate and compare network expansion strategies from an economic and financial point of view. It allows the user to calculate the net present value of a particular network expansion strategy, which is defined through Network Variations and their respective Expansion Stages over time, considering several factors, among others:

- The monetary value of the equipment necessary for the renovation, optimisation or expansion of the investigated network, its depreciation over time, its residual value and its expected useful life,
- Costs arising from technical losses in the grid as well as those arising from service interruptions,
- User-defined costs and additional annual costs (independent of the network expansion).

### 44.2.1 Technical Background

#### 44.2.1.1 General Information

When considering the development of an electrical network, several solutions are often presented that meet the technical requirements of the problem. A techno-economical analysis adds an economic-

financial dimension to the network design problem and allows thus to include an additional decision criterion when considering different strategies for the optimisation and expansion of the electrical power supply system. In addition, a techno-economical analysis can also help to find the optimal allocation of resources to maximise profitability.

In *PowerFactory*, the economic-financial value of planned measures to optimise or extend the network is assessed using the Net Present Value (NPV). Most techno-economical evaluation professionals use the NPV as a standard metric for the financial evaluation of projects. The NPV is calculated as the difference between the present value of cash inflows and the present value of cash outflows over a period of time. This can be expressed in general as:

$$NPV = \sum_{t=0}^N \frac{NCF_t}{(1 + DR)^t} \quad (44.1)$$

where:

- $t$  is the time period.
- $N$  is the total number of periods.
- $NCF$  is the net cash flow at period  $t$ .
- $DR$  is the discount rate.

In general, an investment with a positive NPV is profitable, while an investment with a negative NPV results in a net loss. This concept is the basis for the NPV rule, which states that only investments with a positive NPV should be made. When comparing alternative strategies, the NPV is a useful indicator.

The total NPV is calculated in *PowerFactory* as: NPV of investment - depreciated value + NPV of interruption costs + NPV of cost of losses + NPV of additional costs. The higher the various NPVs, the higher the total NPV and therefore in *PowerFactory* the strategy with the lowest NPV is the most attractive.

#### 44.2.1.2 Economical Data of a Network Strategy

In a techno-economical analysis, the objective is usually to evaluate the economic-financial value of a given network expansion strategy. A strategy can be defined as the set of measures taken to improve the network, together with their execution time.

In *PowerFactory*, a strategy can be captured through Network Variations (*IntScheme*) and their corresponding Expansion Stages (*IntSstage*). In the Expansion Stages, the changes in the network and the expected execution time (*Activation Time*) are saved.

In addition to the network modifications and their schedule, the Expansion Stages allow the definition of essential parameters for the techno-economical analysis of a strategy. These set of parameters can be define in the *Economical Data* tab of the Expansion Stage. Each of them is described below:

- **Costs for expansion**
  - **Investment costs**  
Investment costs are costs incurred by an investor, for example for the acquisition of assets, land, buildings, plant, machinery etc. needed to successfully complete a project.
  - **Additional costs**  
Additional costs are costs which are not explicitly captured or defined by *PowerFactory* TEC. This can include for example maintenance and operational costs.
- **Commercial equipment value**
  - **Original Value**  
The original value is the cost paid to initially acquire an asset. This cost normally includes

the cost to buy an asset, transport the asset to where it is intended to be used, install it, and test it.

– **Scrap value**

Indicates the value of the individual components of an asset when the asset itself is considered no longer usable, because it has reached its expected useful life. The scrap value of an asset is also known as residual value, salvage value, or break-up value.

– **Expected life span**

Assets are manufactured based on a given standard and the manufacturer also provides proper guidelines for wiring, servicing and operation. The expected life span is a statistical value of the average time an asset is expected to carry out specified services as designed by the manufacturer. For example, transformers are generally designed with an expected life span of 30 years, although in many countries transformers have been in service for more than 50 years.

---

**Note:** The *Expected life span* is only used in the Techno-Economical Calculation. This means that a Variation (*IntScheme*) is not taken out of service once its expectancy life is reached.

---

## 44.2.2 Techno-Economical Calculation Command

The following subsections describe the options available on each page of the *Techno-Economical Calculation Command* (*ComTecoco* ).

### 44.2.2.1 Basic Options

#### Calculation Points

Here it is possible to define the period to be considered as well as the points in time at which future cash flows will be calculated.

• **Calculate**

The options in this data framework allow to define how the points in time for which cash flows are calculated are determined. The following options can be chosen:

– **once per year**

If selected, calculations are performed at the beginning of each of the years within the calculation period, i.e. 01.01.“year” 00:00:00:00.

– **for every expansion stage**

If selected, calculations are executed at the *Activation Time* of each Expansion Stage considered in the analysis.

– **for user-defined dates**

If selected, calculations are executed at each user-defined date. To add calculation points to the table, right-click on it, select *Insert Row(s)* and specify the required dates. To automatically populate the table of calculation points with *once per year* dates and *for every expansion stage* dates, press the **Get All Calculation Points** button. The dates can then be edited as required.

---

**Note:** Irrespective of the option selected, calculations are also carried out at the beginning and at the end of the *Calculation Period*. That means that a start calculation on 01.01.*Start* 00:00:00 and an end calculation on 31.12.*End* 23:59:59. are always performed.

---

- **Calculation Period**

The parameters to be entered here define, on the one hand, the point in time at which future cash flows are discounted and, on the other hand, the time horizon for which the analysis is carried out and thus the network expansions to be considered. The following parameters are entered:

- **Start**

Defines the beginning of the period considered for the analysis.

- **End**

Defines the end of the period considered for the analysis.

---

**Note:** Only Expansion Stages whose *Activation Time* lies within the period defined by *Start* and *End* are considered in the calculation of cash flows.

---

### Strategy

A network development strategy can be captured in *PowerFactory* through Network Variations and their corresponding Expansion Stages. In this section of the command it is possible to access the Variations defined in the Network Model and select their status so they can be considered in the Techno-Economical Calculation.

The **Show Activated Variations** button opens a dialog box that lists all Variations and it is possible to activate or deactivate them according to the requirements of the case.

### Additional Settings

The options in this data framework are particularly relevant for the calculation of the discounted cash flow.

- **Calculatory Interest Rate**

Specifies the discount rate used in the net present value calculations. In other words, it is the interest rate used to obtain the discounted value of future cash flows.

- **Tolerance for Calculation Points (in days)**

Defines a tolerance for the activation of Expansion Stages. If, for example, a calculation is to be performed *once per year*, and all Expansion Stages with activation times within January of that year are to be considered as in-service for the entire year, a tolerance of "31 days" could be specified here.

- **Incorporate load growth**

Allows to consider the growth in load within each calculation interval. In contrast to the case where no load growth is incorporated and costs for a calculation period are calculated only at the beginning of calculation time point, enabling this flag will lead to a second cost calculation at the end of the current calculation time point. Corresponding costs are then calculated using a linear estimation based on both values. Load growth is defined via parameter characteristics (see Chapter 18: Parameter Characteristics, Load States, and Tariffs for details of how to define parameter characteristics).

---

**Note:** It should be noted that the *Incorporate load growth* option is only available if the calculation points are calculated *for every expansion stage* or *for user-defined dates*.

---

#### 44.2.2.2 Costs

On this page of the command, the user can configure which costs are to be considered in the Techno-Economical Calculation. The function supports the consideration of the following costs:

- *Losses*
- *Interruption costs*
- *User defined costs*
- *Annual additional costs*

In addition, as long as at least one of the first three options mentioned above has been activated, it is possible to enable a check box to include a Tie Open Point Optimisation during the Techno-Economical Calculation. Each of these costs is described in more detail below.

### Losses

The options relate to technical losses in the network. Following options for considering network losses are available:

- ***Not Considered***  
If selected, costs due to technical losses of the network under investigation are disregarded.
- ***Evaluate by Load Flow Calculation***  
If selected, the losses of the investigated network are considered and evaluated by means of a Load Flow Calculation (*ComLdf*) that is carried out at each of the points in time according to the configuration made in *Calculation Points*.  
A detailed description of the calculation options is given in Section [25.3](#).
- ***Evaluate by Quasi-Dynamic Simulation***  
With this option, the technical losses can be taken into account and calculated based on a Quasi-Dynamic Simulation (*ComStatsim*), whose *Time period* is defined by the Techno-Economical Calculation Command (*ComTececo*).  
For a detailed description of these calculation options, refer to Section [28.3](#).

---

**Note:** Since the Quasi-Dynamic Simulation allows to consider the time dependencies of the variables that determine the load flow and thus the losses in the network (generation dispatch, load demand, etc.), more accurate results are obtained for the magnitude of the technical losses of the investigated system.

---

### Cost for losses

Provided that grid losses are taken into account, it is possible to enter the following parameters for the calculation of loss costs:

- ***Costs for Losses (Load)***  
Indicates the costs due to the load-dependent part of the technical losses.
- ***Costs for Losses (no Load)***  
Specifies the costs due to the part of the losses that is not load-dependent.
- ***Consider user-defined set of substations/feeders only***  
Through this check box, the user can choose whether the losses of the whole network are used or only a specific part is considered to calculate the costs. It should be noted that only Substations (*ElmSubstat*) and/or Feeders (*ElmFeeder*) can be added to *Selection*.

When a feeder, a substation or a set of feeders and substations is selected, the cost of losses is based on the elements belonging to the feeders and/or substations.

### Interruption Costs

If the check box is enabled, the costs of service interruptions are considered in the Techno-Economical Calculation. These costs are evaluated by means of a *Reliability Assessment* (*ComRel3*).

For details on the Reliability Assessment calculation command options, refer to Chapter [46](#).

When executing a Techno-Economical Calculation and considering interruptions costs, the following options must be specified in the Reliability Assessment command:

- *Basic Options* → *Calculation* → *Load Flow Analysis*
- Tariffs for either “Cost for energy not supplied” or “Costs for loads” should be globally specified on the tap Costs. Tariffs are described in Section [18.5](#) (Tariffs).

---

**Note:** Tariffs can also be provided on the reliability tap of the loads. If values configured in the load should be used, then the option for “Costs for loads” should be set to “Individual cost curve per load”. Further information is given in Section [46.3.4](#) (Load Modelling)

---

### User-defined Costs

If this checkbox is enabled, a DPL Script can be linked next to *Cost Assessment Script* to consider fully user-defined costs. This functionality may be required for detailed analysis, where factors besides losses and interruption costs are to be considered in the calculation.

For a general description of DPL Scripts, refer to Section [4.8.1](#).

For a detailed discussion about the *DlgSILENT* Programming Language, refer to Section [23.1](#).

### Optimise Tie Open Points

If selected, a *Tie Open Point Optimisation* (*ComTieopt*) is executed during the Techno-Economical Calculation to re-configure the open points in the selected Feeders (*ElmFeeder*), in order to minimise the costs, according to the selected *Objective Function* in the linked calculation command.

For a detailed description of the *Tie Open Point Optimisation*, refer to Section [42.6](#).

### Additional annual costs

This parameter indicates those annual costs that are independent of the grid development strategy.

---

**Note:** In the cost assessment, the *Additional annual costs* are added to the *Additional Costs* of the considered Expansion Stages.

---

### 44.2.2.3 Results/Output

On this command tab, the user can configure various options relating to the outputs of the calculation.

#### Results

The results of the Techno-Economical Calculation are stored in a *Results* (*ElmRes*) object and can be exported with the **Export** button in the Results object dialog. The *Result Export* (*ComRes*) command enables the definition of the format and the file type used to export the results.

For a general description of the object *Results* (*ElmRes*), refer to Section [19.7](#).

#### Output

Here, the extension of the outputs of the *Techno-Economical Calculation* command can be defined. The following options are available:

- **Short**

If selected, a limited output report of the calculation process is shown.

- **Detailed (full output of invoked commands)**

If selected, the report provides detailed information on the calculation process for each of the functions used by the *Techno-Economical Calculation* and for each calculation point.

As well as the output in the output window, reports can be generated using the  icon. See Section 44.2.3 for details.

#### 44.2.2.4 Parallel Computing

*PowerFactory* supports the parallel execution of the *Techno-Economical Calculation*. This is achieved via the Parallel Computing page of the command dialog. A number of options are provided in this page and described further below.

- **Parallel computation**

By ticking this checkbox, the user switches from the sequential execution to the parallel task processing. If the checkbox is not asserted, the sequential execution of tasks is adopted.

- **Minimum number of calculation points**

This parameter defines the minimum number of calculation points necessary to start a parallel calculation. This means, if the number of calculation points is less than the entered number, the calculation is run sequentially.

- **Parallel computation settings**

Provides a link to the Parallel Computing page in the User Options edit dialog. The options on the General tab and the Advanced tab are described in detail in Section 7.11. Further information on the *Parallel computation settings* and the *Parallel Computing Manager* is available in Section 6.6.1.

- **Additional objects to transfer**

This option may be needed to transfer some external objects which are not stored in the project or the global library itself, since not all external references are automatically resolved by the sub-processes.

#### 44.2.3 Handling of Results

After executing a *Techno-Economical Calculation*, reports can be generated using the *Techno-Economical Calculation Report* icon , in the *Economic Analysis Tools* toolbox.

This brings up the Report Generation command (*ComReport*), where the available reports are listed and can be selected. Reports can be generated as separate documents or combined into one. By default, reports are generated as PDFs and presented in *PowerFactory*'s inbuilt PDF viewer, but it is also possible to export reports from *PowerFactory* in various different formats.

For more information about reports and the Report Generation command, see Chapter 19: Reporting and Visualising Results, Section 19.5. It should be noted that reports viewed internally in *PowerFactory* are stored in the desktop of the study case, even after they have been closed, until they are actively deleted by the user.

The following calculation-specific reports are available:

- **Calculation Settings**

This report shows the settings used for the *Techno-Economical Calculation*.

- **Costs of calculation periods**

If this option is selected, a report of the cash flows for each of the calculation points is created.

This includes the discounted values of the considered variables such as Investment Costs, Depreciation of assets, Costs for Losses, User-defined Costs, etc. In addition, a summary report displaying the Net Present Value of the considered network expansion strategy is included.

- **Variation Input Data**

This report shows the configurations of the active Expansion Stages containing economic data.

---

**Note:** Irrespective of the calculation option selected, the results are reported annually. This provides user-flexibility to optimise the performance of the Techno-Economical Calculation, whilst retaining the ability to compare annual results with different calculation options.

---

#### 44.2.4 Example Calculation

Consider the following Techno-Economical Calculation example, which also consolidates functionality presented on the following topics:

- Project Variations: Discussed in Chapter [17](#) (Network Variations and Expansion Stages).
- Reliability: Discussed in Chapter [46](#) (Reliability Assessment).
- Parameter Characteristics and Tariffs: Discussed in Chapter [18](#) (Parameter Characteristics, Load States, and Tariffs).

The current year is “2010”. There are four 12 MW loads connected to DoubleBusbar/A and DoubleBusbar/B. In the current arrangement the line “Existing Line” from “Sub 1” is lightly loaded (see Figure [44.2.1](#)).

High load growth is expected from 2010 to 2016, with constant demand thereafter. To model the changes in demand, a *One Dimension - Vector Characteristic* from 2010 to 2020 has been defined for each load. By setting the *Study Time* to 2014, it has been determined that “Existing Line” will be loaded at close to the thermal rating in this year (see Figure [44.2.2](#)).

Based on this, it has been determined that a new substation is required in 2015 to off-load the existing line. Figure [44.2.3](#) shows the case with the *Study Time* set to 2015, and the new substation “Sub 2” in service. Half of the load from “Sub 1” has been transferred to “Sub 2”. Note that the new substation has been implemented as a *PowerFactory* Variation, and hence is shown with yellow dashed lines in cases where the Study Time is prior to 2015.

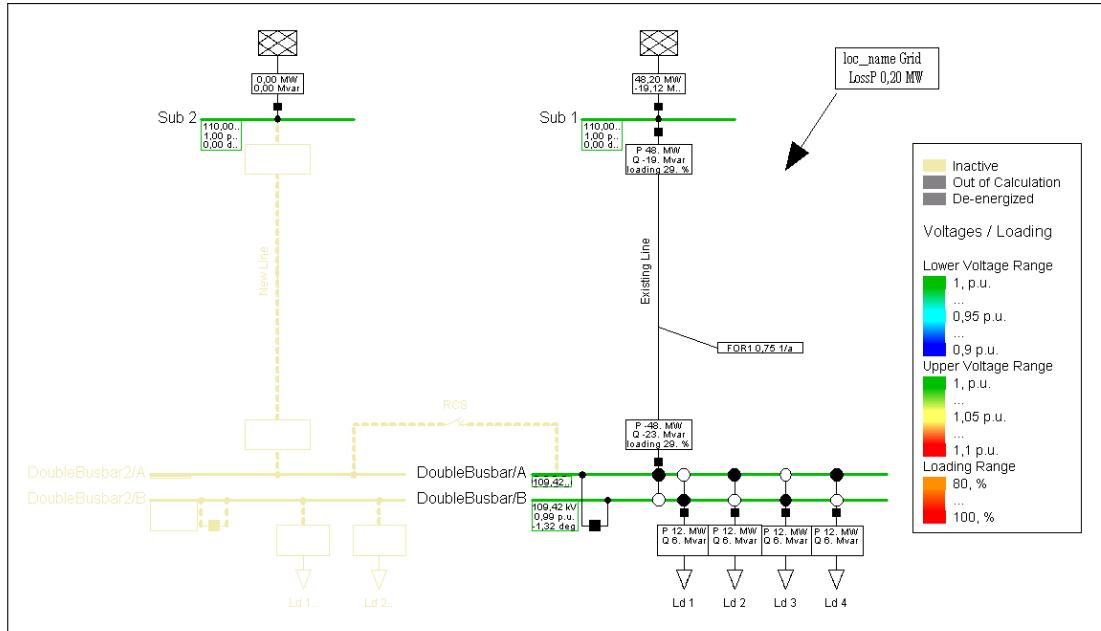


Figure 44.2.1: Example case, study time “2010”

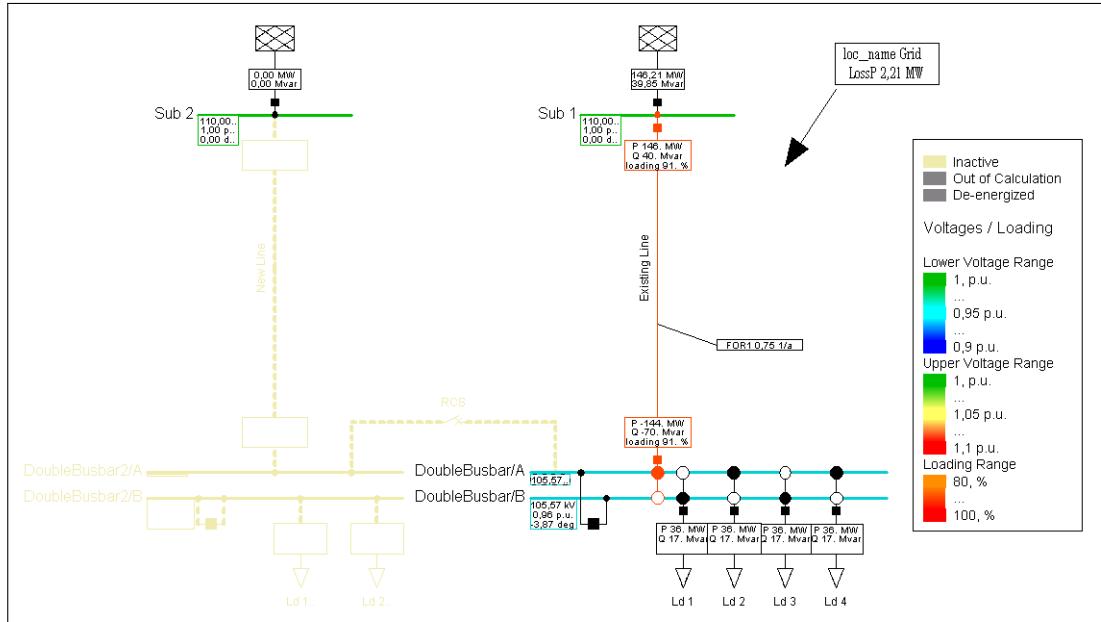


Figure 44.2.2: Example case, study time “2014”

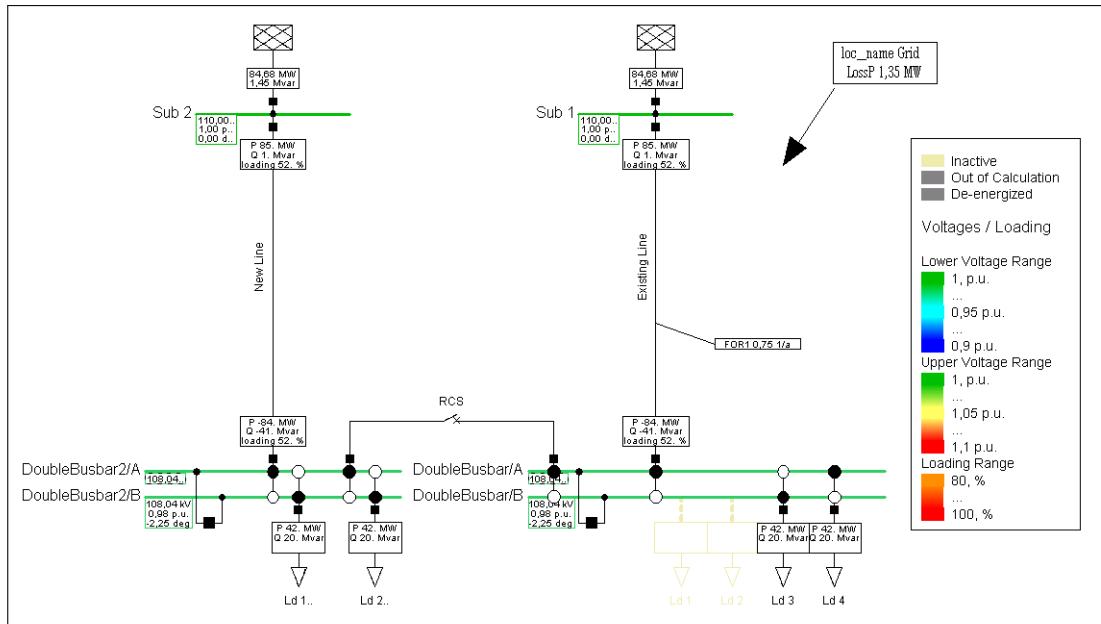


Figure 44.2.3: Example case, study time “2015”

However, the previous analysis has not considered the economic impact of interruption costs. In the “2010”, when there is an outage of the line from “Sub 1” there are no alternative paths to re-establish supply to either load. With the new line and DoubleBusbar/A and B in service, there is an alternative path to re-establish supply to loads in the event of an outage on either “New Line” or “Existing Line”. To understand the economic implications of commissioning the project prior to 2015, in particular the sensitivity of the cost of losses and cost of interruptions to the project commissioning date, a Techno-Economical Analysis is performed for a number of *Activation Times*.

To perform the analysis, the Variation activation time  $T(\text{act.})$  is varied from 2010 to 2015, and the Net Present Value (NPV) of the Strategy is calculated over the period 2010 to 2020. In the example, outage data has been entered for the lines “New Line” and “Existing Line”, and a Global Energy Tariff has been defined for loads from the Reliability command *Costs* page. Due to the trade-off between Energy Interruption Costs (increasing in this example due to load growth) and cost-benefits associated with delaying the project (based on the specified interest rate), the optimum year for project commissioning is determined to be 2011, and not 2015. The NPV is around 11 % lower in 2011 than in 2015. Table 44.2.1 below summarises the results of the Techno-Economical Calculations.

$T(\text{act.})$	NPV
01.01.2010 00:00	69743,68
01.01.2011 00:00	69604,76
01.01.2012 00:00	69664,02
01.01.2013 00:00	71214,65
01.01.2014 00:00	74048,42
01.01.2015 00:00	77982,59

Table 44.2.1: Summary of Calculation Results

## 44.3 Techno-Economical Study Case Comparison

Whether maintenance, replacements, improvements or expansions of power systems, the Techno-Economical Calculation (TEC) tool enables utilities to analyse the impacts of network expansion on the cost they necessitate over the years. The Techno-Economical Study Case Comparison (TECC) enables the comparison of different planning strategies using Net Present values and other economical quantities to be examined in this section.

The calculation of the quantities given in Section 44.3.1 has been implemented as an internal command (\*.ComTececoCmp), which is at the moment not available on the tool bar. This command enables the cost evaluation and comparison of different strategies based on the NPV and the economical quantities described in Section 44.3.1. The TECC Command has several options, which are configured and driven by the script “EvaluateTececoCalcForStudyCases” located in the *DlgSILENT* library.

### 44.3.1 Additional Techno-Economical Calculation Quantities

*PowerFactory*'s Techno-Economical Calculation (TEC) can be used to analyse investment costs, cost of losses, interruption costs, additional and user defined costs as well as cost benefits of expansion stages using the Efficiency Ratio. The Efficiency ratio expresses the benefits of grid extension with respect to the investment costs - for each expansion stage defining the grid extension process. In addition to the Net Present Value the following techno-economical quantities are also supported:

- Internal rate of return (IRR)
- Estimated payback period
- Discounted estimated payback period (DPP)

To calculate these quantities an evaluation of costs (such as costs for losses or expected interruption costs) with and without application of the investment/grid-expansion strategy have to be computed. The strategy without applying the grid expansions is referred to as the “Base Case strategy”. At times this is also referred to as the “Do Nothing” strategy. A strategy defines a plan of action to attain an objective or set of objectives such as network expansion, comprising a number of steps. The strategy is modelled in *PowerFactory* using a “Study Case”.

The following terminology will be used:

- T: Total number of time points for evaluating the costs (typically on a yearly basis).
- t: Current time period index.
- $Cost_t^{Invest}$ : The investment costs for period t.
- $Cost_t^{Benefit}$ : The benefits in costs for period t. This includes, e.g., the expected interruption costs, costs for losses of the individual periods, as well as the depreciated value of the equipment.
- IRR: Resulting internal rate of return.
- IR: Calculatory interest rate.
- K: Pay Back Period.

#### Internal rate of return:

The internal rate of return (IRR) can be computed using the following equation:

$$\sum_{t=0}^{T-1} \frac{Cost_t^{Benefit}}{(1 + IRR)^t} - \sum_{t=0}^{T-1} \frac{Cost_t^{Invest}}{(1 + IRR)^t} = 0 \quad (44.2)$$

The above equation has to be solved to get the IRR. An internal rate of return of an investment is a calculation interest rate, which results in a net present value of zero. In other words, an internal rate

of return is the discount factor, the use which leads to discounted future payments corresponding to today's price or the initial investment. When this interest rate is greater than the calculation interest rate then the investment is economical over the total investment period. Generally speaking, the higher an internal rate of return, the more desirable an investment is to undertake.

#### **Estimated payback period:**

The estimated payback period is the minimum amount of time it takes until the investment costs, for example of the grid expansion, are amortised by the benefits. It is calculated as minimum for values of  $K > 0$ :

$$\min_{K>0} \left( \sum_{t=0}^{K-1} Cost_t^{Benefit} - \sum_{t=0}^{T-1} Cost_t^{Invest} \geq 0 \right) \quad (44.3)$$

In the formula,  $K$  is the first year (between  $0 \leq K \leq T$ ), so that the entire term  $\geq 0$ . In other words, the first year in which the benefits dominate the total costs from year 0 to  $K$ . The term payback period refers to the amount of time it takes to recover the cost of an investment. Therefore, the estimated payback period can help to decide whether an investment project will be carried out or not. In essence, the shorter payback period an investment has, the more attractive it becomes.

#### **Discounted estimated payback period:**

The discounted estimated payback period is similar to the above estimated payback period, but all costs are discounted using the calculatory interest rate. It is calculated as minimum for values of  $K > 0$ :

$$\min_{K>0} \left( \sum_{t=0}^{T-1} \frac{Cost_t^{Benefit}}{(1 + IRR)^t} - \sum_{t=0}^{T-1} \frac{Cost_t^{Invest}}{(1 + IRR)^t} \geq 0 \right) \quad (44.4)$$

The discounted payback period is a method of investment calculation that can be used to determine the profitability of a project. The discounted payback period indicates the number of years required to recover the initial expenditure by discounting future cash flows and taking into account the time value of money.

### **44.3.2 Techno-Economical Study Case Comparison Command**

The TECC Command is configured and driven by the script "EvaluateTececoCalcForStudyCases" located in the *DlgSILENT* library. This script should be used for comparison because it also provides a report with the economic quantities described in Section [44.3.1](#) for the various strategies.

#### **44.3.2.1 Basic Options**

**Base case:** The "Base case" study case is the study case to which literature as a "Do Nothing" strategy where no network enhancements, which will result in investments are done.

**Study Cases:** A list of study cases (strategies) where different network enhancements or different sources of costs or influence of parameters are being investigated. Study cases can be added or removed from the list of study cases. They can also be ignored during the assessment.

The Button **Add** enables adding a new study case to the list of study cases and button **Remove All** clears the list of study cases.

**Tasks:** A series of tasks to be executed can be configured.

- *Run Techno-Economical Calculations;* After running Techno-Economical Calculations results are stored in the result file. So it is not necessary to re-run the calculation if the results of previous calculations are to be analysed. If this option is selected, the Techno-Economical Calculation will be carried out for all study cases. If not selected, existing results will be analysed.

- *Create summary result file*; For each calculation TEC delivers results for the various NPVs along the time line. However, to compare different strategies it is important to use the sum of all these values. With this option selected, values are written to a result file and can be assessed like other parameters.
- *Create time sweep result files in study cases*; Usually, the Techno-Economical Calculation does not store results for the various calculation time points for each strategy. Since it is at times interesting to know that the NPV changes over the year, the time sweep results can be written into a result file.
- *Create plots for summary results*; The summary result file contains the sum of the various TEC quantities for each strategy. With this option these results can be plotted as bar diagrams
- *Create plots for timesweep results*; TEC quantities are calculated at each time point. This option enables the display of changes of TEC quantities with time.

#### 44.3.2.2 Plots

##### Results reporting and visualisation

The creation of a report and the visualisation of results in plots are core elements of this new tool. In the *Techno-Economical Study Case Comparison* command, the user can select the plots to be created. For the Basic analysis, the following plot categories are available:

- *Net present value of total costs*
- *Net present value of total investments*
- *Net present value of cost for losses*
- *Net present value of expected interruption costs*
- *Net present value of user-defined costs*

#### 44.3.3 Techno-Economical Study Case Comparison Example

Techno-economical studies can not only be carried out to compare different strategies but can also be geared for example at studying the impact of a parameter, e.g. interest rates on the NPV and other quantities. For comparison studies based on TEC, it is always useful to create three groups of study cases:

- Base case study case for comparison. This is known in most cases as the “Do Nothing” case.
- A set of study cases reflecting different investments and network enhancements.
- A third study case, an evaluation case, is the case where the Techno-Economical Comparison study is configured and the results are represented. It could be one of the study cases above, but it is recommended to have a separate study case for this purpose.

Unlike many functions in *PowerFactory*, which generally involve carrying out studies for a given study case, the techno-economical comparison has to be carried out over a series of study cases as given above. Therefore the “EvaluateTececoCalcForStudyCases” located in the *DlgSILENT* library is used to automate this process. It configures and executes the TECC Command and generates a report for the various TEC quantities.

Figure 44.3.1 and Figure 44.3.2 visualise an extract of the result plots that are created based on the techno-economic analysis of an example network using the comparison command. Two strategies having the same network expansions with the same investments, differing only by the fact that in one case the Tie Open Point Optimisation is not considered, are studied. The interruption costs, the cost of losses and user defined costs are considered in these two strategies and compared with the Base

Case (“Do Nothing”). For the x-axis 0 → Base Case, 1 → Strategy with tie open point optimisation and 2 → Strategy with no tie open point optimisation

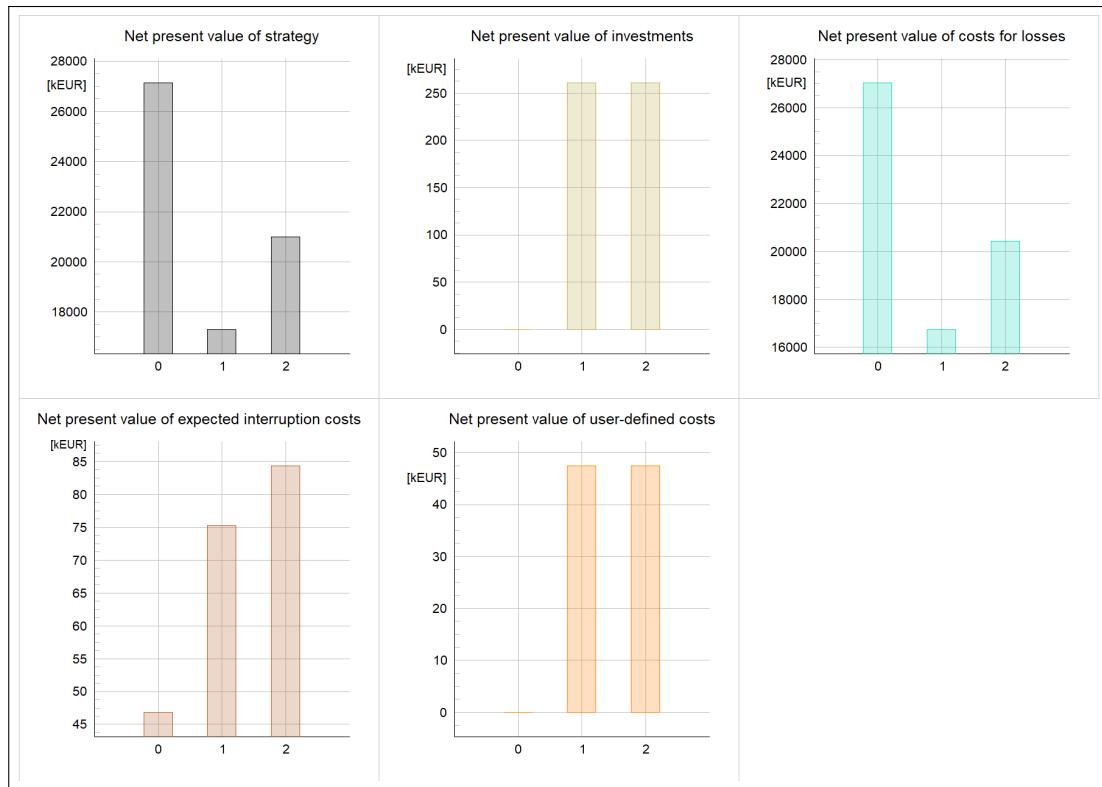


Figure 44.3.1: Example case, summary results

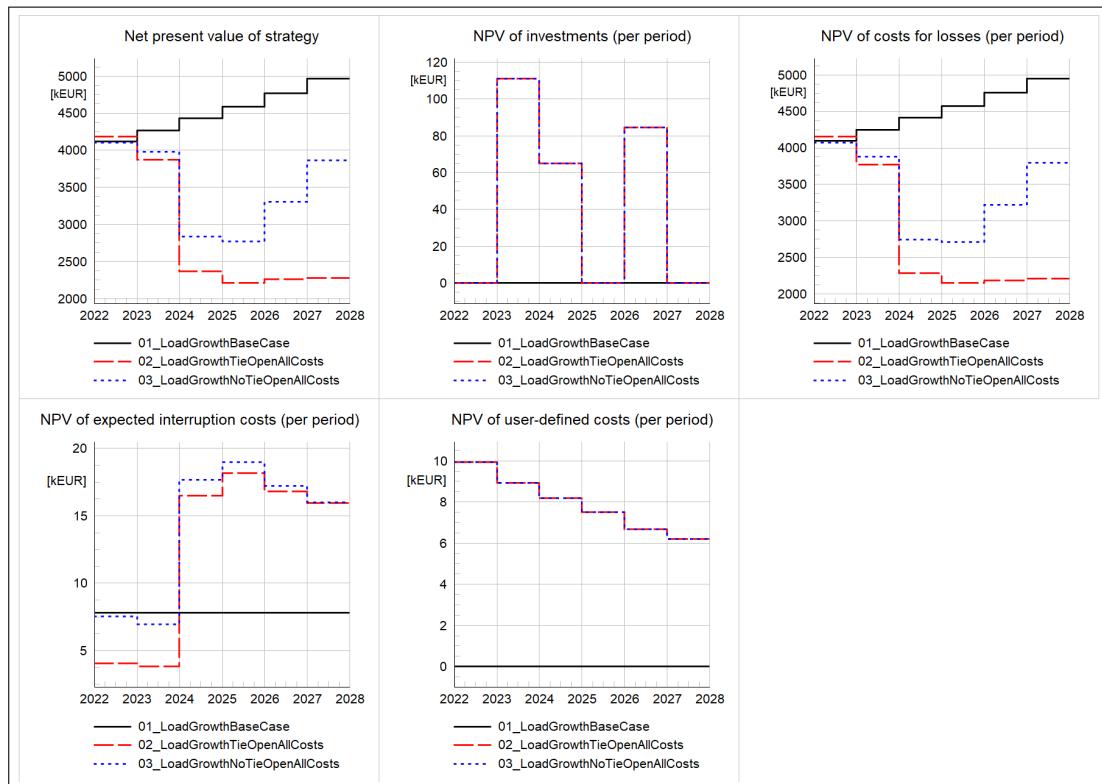


Figure 44.3.2: Example case, time sweep

It is possible to create plots for all the supported and derived quantities. As well as these plots, a flexible and adaptable tabular report is created.

Figure 44.3.3 shows an extract of an example results report based on the TEC comparison of two strategies with a base case. This report is expandable and therefore the types and number of the shown parameters can be configured by removing or adding more variables.

The screenshot shows a software interface for comparing study cases. At the top, it displays the title 'Reports - Results of technico-economical comparison of study cases' and the current view 'Results of technico-economical comparison of study cases'. Below this, there are fields for 'Base case:' (01\_LoadGrowthBaseCase), 'Starting year of calculation period:' (2010), and 'End year of calculation period:' (2015). A toolbar at the top right includes icons for 'html' and 'xlsx'. The main area is a table with the following data:

Study Case	Study Case Index	Internal Rate of Return [%]	Estimated Payback Period [a]	Discounted Estimate [a]	Total net present value benefit [kEUR]	Net present value of strategy [kEUR]	Calculatory Interest Rate [%]
01_LoadGrowthBaseCase	0	-	-	-	-	2521,66	0,00
02_LoadGrowthTieOpenAllCosts	1	203,86	1,24	1,27	9721,02	15491,64	10,00
03_LoadGrowthNoTieOpenAllCosts	2	168,22	1,36	1,40	6602,35	18610,31	10,00

At the bottom of the table, status information is displayed: 'Ln 2', '3 Line(s) of 3', and '0 Line(s) selected'.

Figure 44.3.3: Example case, Report

This user-adaptable report compares the Base Case of the techno-economical quantities with the benefits of grid expansion.

## 44.4 Power Park Energy Analysis

The Power Park Energy Analysis function provides an evaluation of profitability of power plants based on load flow calculations.

Network calculations are essential when it comes to determining the network losses of a power park, based on the possibly volatile infeed of the generating units, and the topology as well as the mode of operation of the power plant internal network. The Power Park Energy Analysis tool combines the powerful network calculation functions of *PowerFactory* with an economical evaluation. With this tool, the user can conveniently determine important quantities such as energies, profit and loss for a power plant, derived from network calculation results.

In the Power Park Energy Analysis tool, the following functions are available:

- *Power Park Energy Analysis* (see Section 44.4.1)
- *Power Park Energy Analysis Report* (see Section 44.4.2)
- *Edit Result Variables* (see Section 44.4.3.1)
- *Create Curve Plot* (see Section 44.4.3.2)

### 44.4.1 Power Park Energy Analysis Configuration

The Power Park Energy Analysis can be accessed by clicking on the *Power Park Energy Analysis* button in the Economic Analysis Tools toolbox. In the following subsections, the settings for the configuration of the Power Park Energy Analysis are described.

#### 44.4.1.1 Basic Options - General

There are three calculation methods available for the Power Park Energy Analysis:

- Basic Analysis
- Time-series Analysis
- Probabilistic Analysis

After choosing the desired calculation method for the Power Park Energy Analysis, the corresponding command of the selected method is accessible via the button → underneath the selection area of the calculation methods.

With regard to the Basic Analysis method, the Load Flow Calculation command (see Section 25.3) is linked in the Power Park Energy Analysis command. For the Time-series Analysis, the Quasi-Dynamic Simulation command (see Section 28.3) is accessible. Likewise for the Probabilistic Analysis, the calculation command (see Section 45.3.4) of the same name is linked. Further settings can be made in these respective calculation commands, which are taken into account accordingly in the Power Park Energy Analysis.

#### Basic Analysis

The Basic Analysis is particularly designed for power plants consisting of wind generators and is carried out by using load flow calculations.

In order for the Basic Analysis to be executed, the user needs to define the corresponding generating units in the power plant as wind turbines. This can be achieved by selecting the plant category "Wind" in the basic options of the generating unit element. In addition, the load flow model of the wind generator then needs to be configured with the "Wind speed input" and an associated "Wind Power Curve".

Generator elements that can be configured accordingly are static generators, asynchronous generators and synchronous generators.

At least one generating unit in the power park has to be modelled with wind speed input.

For the wind speed distribution of the entire power park a single Weibull distribution is used. This implies that the wind speed distribution is identical for all wind turbines. The required data for the Weibull distribution can be entered on the "Wind speed" tab (see Section 44.4.1.2).

Using the Basic Analysis method, the wind speed is swept from 0 m/s up to a value which is defined via the "Confidence level" on the "Wind speed" tab (see Section 44.4.1.2). For each wind speed step, the active power output of the corresponding wind turbines is determined using their individual wind power curves. Based on load flow calculations, the electrical losses in the power plant internal network as well as relevant energy key figures are determined for each wind speed step. The corresponding probabilities for the respective wind speeds from the specified Weibull distribution are used to determine further quantities such as average powers and average electrical losses.

For the Basic Analysis, the use of the balanced as well as unbalanced AC load flow calculation is supported.

#### Time-series Analysis

The Time-series Analysis provides the option to perform the energy yield analysis of power parks combined with an economical evaluation over a user-defined period of time.

This calculation method of the Power Park Energy Analysis is based on the Quasi-Dynamic Simulation (28). It uses a series of load-flow calculations with various network model parameters considered time dependent. Time based parameter characteristics can be provided in order to define time dependencies for many input parameters of numerous elements. For example, to consider a variable active power

injection of a generating unit over time, a time characteristic with regard to the active power input of the unit can be used (refer to Chapter 18 for more information about characteristics).

In addition, within the Time-series Analysis user defined load flow and Quasi-Dynamic Models (QDSL Models) of various power system equipment can be considered. For example, for a battery energy system a Quasi-Dynamic Model can be created in order to represent the time dependent behaviour of the battery system including its state of charge. For information about how to develop a Quasi-Dynamic Model, refer to Section 28.5.

Following the link in the Power Park Energy Analysis to the Quasi-Dynamic simulation command (), there are further options available for setting up the calculation using the Time-series Analysis. These include the determination of the user defined time period for the analysis as well as the step size and other calculation settings. Moreover, maintenance outages and simulation events can be created via the 'Maintenance & Events' tab in the Quasi-Dynamic simulation command. To consider maintenance outages and simulation events refer to Sections 28.3.2 and 28.3.3.

### Probabilistic Analysis

The Probabilistic Analysis method enables the user to carry out the energy analysis taking probability distributions of various quantities into account. For this reason, the Probabilistic Analysis calculation method of the Power Park Energy Analysis uses the Probabilistic Analysis module functionality (45).

For many input parameters, probability distributions can be provided to be considered in the Probabilistic Analysis calculation method. For example with respect to a wind turbine configured with the load flow model "Wind Speed Input", a Weibull distribution can be provided for statistical modelling of wind speeds. Further information about the assignment of distributions and available types can be found in Section 45.2.1.

For cases where random quantities in the power system are not independent from each other, correlations can be defined using copula elements. A copula element can, for example, be used in a wind farm, where the wind turbines are located relatively close to each other. In such case, it could be assumed that the wind speeds at the individual turbines do not differ fundamentally but are correlated to each other. For a description of how to define a copula, see Section 45.3.2.

When selecting the Probabilistic Analysis calculation method in the Power Park Energy Analysis command, the calculation command of the Probabilistic Analysis module can be accessed via the button. Here, further settings are available to the user. This includes the choice of either Monte Carlo or Quasi-Monte Carlo method, the maximum number of samples, and the seeding type of the random number generation.

### Temperature Dependency: Line/Cable Resistances

For each calculation method of the Power Park Energy Analysis, the temperature dependency of line/cable resistances can be configured using the dedicated drop-down menu. This allows the influence of the cable and line temperature on the network losses to be investigated closely. The following temperature options are available:

- 20 °C
- Maximum operating temperature
- Operating temperature
- Temperature (specific temperature value)

Further information regarding the consideration of temperature dependency of lines and cables can be accessed in Chapter 25.4.4.

### Power park

The power park can be specified by using one boundary element (). To select a boundary, click on the button next to the words "Power park" and then *Select...*. This will open a list with the existing

boundaries in the active project from which the user can select a boundary element.

In general, with boundary objects topological regions can be defined. This is used in the Power Park Energy Analysis to distinguish between the power plant and the rest of the network. For a general description of how a boundary can be defined, see Section 15.4.

Since boundaries are defined by the cubicles that determine the cut through the network, these cubicles together with the orientation define the corresponding “Interior Region” which is in this case the power park. For the proper definition of the power park, it is especially important to configure the orientation of the boundary so that it points to the power plant. The busbar(s) at which the boundary is defined represent the point of connection of the corresponding power park. All elements that are inside of the interior region of the boundary are seen as part of the power plant and are considered in the evaluation of the Power Park Energy Analysis.

#### **Agreed active connection power**

The “Agreed active connection power” represents the rated value of the active power of the power park. It is used as a reference value for the calculation of the full load hours of the power plant.

##### **44.4.1.2 Basic Options - Wind speed**

The input data on the Wind speed tab is exclusively available for the Basic Analysis method.

###### **Weibull curve data**

For the Basic Analysis method, the user can define one Weibull distribution as a probability density function for the wind speeds in the power park. The associated data for the Weibull curve that can be provided are the scale factor and the shape factor. The input of the scale factor can also be changed to the average wind speed via the adjacent button .

###### **Confidence level**

The confidence level is used to define the range between 0 m/s and maximum wind speed in which the true wind speed is present with the specified probability of the confidence level. For example, a confidence level of 0.99 can be interpreted such that there is a 99 % probability that the true wind speed is less than or equal to the wind speed value calculated with the cumulative Weibull distribution function using the confidence level of 0.99. Up to this certain wind speed value, the wind speed sweep is carried out within the Basic Analysis method

###### **Wind speed step size**

The wind speed step size indicates the intervals at which the wind speed is swept over the given Weibull distribution curve, up to the maximum wind speed value determined by using the confidence level. For each step of the wind speed, a load flow calculation is executed.

##### **44.4.1.3 Tariffs**

###### **Feed-in tariff**

Based on the feed-in tariff and the amount of energy fed into the network to which the power park under consideration is connected, the feed-in remuneration is calculated.

###### **Consumer tariff**

The consumer tariff can be provided for the determination of the costs of the consumed energy of the power plant.

#### 44.4.1.4 Advanced Options

For the study year, there are three different options available for the Basic Analysis and the Probabilistic Analysis calculation methods:

- Common year (total hours: 8760 h)
- Leap year (total hours: 8784 h)
- Mean year (total hours: 8765,82 h)

The study period for the Time-series Analysis can be configured via the associated command of the Quasi-Dynamic Simulation.

#### 44.4.1.5 Output/Results

##### Results

The results of the Power Park Energy Analysis are written in a result object, which can be selected next to the “Results” label. By default, when the command of the Power Park Energy Analysis is opened and no associated result object has been created yet, a new result object for the selected calculation method of the Power Park Energy Analysis will be created automatically. This simplifies the handling of the function.

The result objects used for the Power Park Energy Analysis contain sub-result objects, where the calculated data is stored. In one sub-result object the results for the Power Park Energy Analysis report is written. The other sub-result objects contain data series that, for example, can be visualised using diagrams.

##### Output

For displaying messages of the Power Park Energy Analysis in the output window, the user can choose between the following output options:

- Off
- Short output
- Detailed output (full load flow output)

#### 44.4.2 Power Park Energy Analysis Report

##### 44.4.2.1 Main reports

After the Power Park Energy Analysis has been carried out, the dedicated Power Park Energy Analysis report command can be executed using the  icon.

This will bring up the Report Generation (*ComReport*) command, where reports can be selected from the list of available reports, with the generally used reports selected by default. Reports can be generated as separate documents or combined into one. By default, reports are generated as PDFs and presented in *PowerFactory*'s inbuilt PDF viewer, but it is also possible to export reports from *PowerFactory* in various different formats.

On the Basic Options page of the command, the result object from which the results of the Power Park Energy Analysis are to be displayed in the report can be selected by clicking on the button  and then *Select....* By default, the result object of the last execution of the Power Park Energy Analysis will automatically be selected.

For more information about reports and the Report Generation command, see Chapter 19: Reporting and Visualising Results, Section 19.5. It should be noted that reports viewed internally in *PowerFactory* are stored in the desktop of the study case, even after they have been closed, until they are actively deleted by the user.

#### 44.4.2.2 Quasi-Dynamic Simulation report

If the Time-series Analysis method of the Power Park Energy Analysis is selected, it is also possible to generate the Quasi-Dynamic Simulation report after the calculation has been run. This is done using the  icon.

### 44.4.3 Visualisation of Power Park Energy Analysis Results

#### 44.4.3.1 Edit Result Variables

Before executing the Power Park Energy Analysis, the user can click on the button “Edit Result Variables”  in the Economic Analysis Tools toolbox. This opens a dialog where the desired method for the variable selection can be chosen. After confirming with *OK* another window opens for the result variable selection. Click on the *New Object* button  to create a new Variable Selection object and choose the result variables for previously selected method of the Power Park Energy Analysis or edit the Variable Selection objects, if they already exist. The selected result variables are recorded when the command of Power Park Energy Analysis is executed and are then available for visualisation in diagrams.

The Variable Selection object is described in Section 19.3: [Variable Selection](#).

#### 44.4.3.2 Create Curve Plot

For the visualisation of the results from the Power Park Energy Analysis, the user has access to various diagram types. By clicking on the button  in the Economic Analysis Tools toolbox, the dialog for inserting a plot is opened. Pre-configured plots are available in the Power Park Energy Analysis category. For results from the Time-series Analysis and the Probabilistic Analysis calculation methods, plots from the “Quasi-Dynamic Simulation” and “Probabilistic Analysis” categories respectively can also be selected. In this case, when selecting the variables for the plot curves in the “Data Series” window, make sure to select in the *Result File* column a sub-result object from the result object of the Power Park Energy Analysis (i.e. a sub-result object containing the recorded data series, which is stored inside the result object of the Power Park Energy Analysis).

# Chapter 45

# Probabilistic Analysis

## 45.1 Introduction

The probabilistic analysis tool allows network assessment based on probabilistic input data rather than assessment of individual operation scenarios or time sweep analysis. It becomes important as soon as input parameters are known to be random or if one wants to simulate the grid at some time in the future with forecast errors.

With this functionality *PowerFactory* takes account of recent trends in power system studies, where stochastic assessment is seen as an alternative to pure worst-case assessment of grid capabilities.

Generally speaking, a probabilistic assessment processes probabilistic data input and produces stochastic results, i.e. each result quantity will no longer be a fixed number but a distribution from which statistic quantities (e.g. mean values, standard deviations, min, max, etc.) can be derived.

The new *PowerFactory* approach for probabilistic data input is very generic, which means that distribution curves can be assigned to any arbitrary input parameter. In this sense, the concept of distributions is very similar to that of characteristics.

Also, our approach is generic in terms of the calculation functions to be supported. Probabilistic Analysis is offered for:

- Load Flow Analysis;
- Optimal Power Flow.

The Probabilistic Analysis of Load Flow could be used, for example, in order to determine the distributions of the loadings of lines, given some forecast errors of a predefined weather situation. The Optimal Power Flow may be interpreted as a dispatch strategy under some given objective. In this respect, the Probabilistic Analysis of OPF allows the distribution of controls to be calculated.

The fundamental functions of this module, i.e.

- Processing of input data;
- Executing a simulation;
- Defining result variables for the analysis;
- Detailed investigation of results including visualising via stochastic plots;

are grouped in *PowerFactory* in the main toolbar “Probabilistic Analysis”.

## 45.2 Technical Background

### 45.2.1 Distributions

For the probabilistic analysis distributions are necessary, because they provide the probabilities of occurrence of different quantities. In *PowerFactory* several types of distributions on parameters are available, which are described in more detail in the following sections:

- Predefined distributions, such as Normal, Weibull, Lognormal, etc.
- Distributions based on available characteristics.
- Distributions based on wind power curve and Weibull distribution.

Information about the dialogs of these objects can be found in Section 45.3.1, whereas the technical background is described here.

#### 45.2.1.1 Representation of Distributions

For most of the available distributions two different types of representation can be selected:

- **Probability density function:**

This function specifies the infinitesimal probability at any point  $x$ . The area between the probability density function  $f(t)$  and the  $x$ -axis from a point  $a$  to a point  $b$  corresponds to the probability of having a value between  $a$  and  $b$ :

$$F(a \leq X \leq b) = \int_a^b f(t)dt \quad (45.1)$$

- **Cumulative distribution function:** this representation can be obtained by integrating the probability density function. From this distribution the probability that the random variable is less than a given value  $x$  can be extracted.

#### 45.2.1.2 Predefined Distributions

##### Bernoulli distribution:

This distribution (*RndBernoulli*) is a probability distribution of a random variable with two possible outcomes: 1 and 0. Value 1 occurs with a probability of  $p$  (*Probability of Success*) and value 0 with a probability of  $q = 1 - p$ . The probability mass function  $f$  of the Bernoulli distribution can be expressed as:

$$f(k; p) = p^k(1 - p)^{1-k} \quad for k \in \{0,1\} \quad (45.2)$$

Mean and standard deviation of this function are calculated by:

$$\mu = p, \quad \sigma = \sqrt{p(1 - p)} \quad (45.3)$$

##### Discrete finite distribution:

In this distribution (*RndFinite*) probabilities for discrete values can be defined. The probability  $p_k$  of a discrete value results from the specified weightings ( $w$ ):

$$p_k = \frac{w_k}{\sum_{i=1}^N w_i} \quad (45.4)$$

**Exponential distribution:**

A continuous random variable  $X$  satisfies the exponential distribution (*RndExp*) with the parameter  $\lambda$  ( $\lambda > 0$ ) (*Rate*), if it has the probability density function:

$$f(t) = \begin{cases} 0 & \text{for } t < 0, \\ \lambda e^{-\lambda t} & \text{for } t \geq 0 \end{cases} \quad (45.5)$$

Mean and standard deviation of this function are calculated by:

$$\mu = \frac{1}{\lambda}, \quad \sigma = \frac{1}{\lambda} \quad (45.6)$$

This distribution is often used to answer questions related to durations of random time intervals, like the lifetime of components, if signs of ageing are not being considered.

**Geometric distribution:**

This distribution (*RndGeo*) is derived from independent Bernoulli experiments. In *PowerFactory* the probability distribution of the number  $Y$  of failures before the first success is used. The probability mass function can be expressed as:

$$f(k; p) = (1 - p)^k p \quad \text{for } k \in \{0, 1, 2, 3, \dots\} \quad (45.7)$$

Mean and standard deviation of this function are calculated by:

$$\mu = \frac{1 - p}{p}, \quad \sigma = \sqrt{\frac{1 - p}{p^2}} \quad (45.8)$$

**Log-normal distribution:**

This distribution (*RndLognormal*) might be used for modelling household loads probabilistically. The probability density function of this distribution with the parameters  $\mu_L$  (*Mean of logarithm*) and  $\sigma_L$  (*Standard deviation of log.*) can be expressed as:

$$f(t) = \begin{cases} 0 & \text{for } t \leq 0, \\ \frac{\log e}{t\sigma_L \sqrt{2\pi}} \exp -\frac{(\log t - \mu_L)^2}{2\sigma_L^2} & \text{for } t > 0 \end{cases} \quad (45.9)$$

Mean and standard deviation of this function are calculated by:

$$\mu = \exp(\mu_L + \frac{\sigma_L^2}{2}), \quad \sigma = \sqrt{(\exp \sigma_L^2 - 1) \exp (2\mu_L + \sigma_L^2)} \quad (45.10)$$

**Normal distribution:**

This distribution (*RndNormal*) can be applied in order to represent real-valued random variables (like quantities with measurement errors), for which the distribution is not known. The probability density function of this distribution with the parameters  $\mu$  (*Mean*) and  $\sigma$  (*Standard deviation*) can be expressed as:

$$f(t) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(t-\mu)^2}{2\sigma^2}} \quad (45.11)$$

If  $\mu = 0$  and  $\sigma = 1$  the standard normal distribution with the probability density function is obtained:

$$f(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}t^2} \quad (45.12)$$

**Uniform distribution**

This distribution (*RndUnif*), which is also called rectangular distribution, has a constant probability

density in the interval  $[a,b]$ . In other words: all subintervals with the same length have the same probability. The probability density function can be written as:

$$f(t) = \begin{cases} \frac{1}{b-a} & \text{for } a \leq t \leq b, \\ 0 & \text{for } t < a \text{ or } t > b \end{cases} \quad (45.13)$$

The interval boundaries  $a$  and  $b$  can be calculated from the input parameters *Mean* ( $\mu$ ) and *Magnitude* in *PowerFactory* by:

$$a = \text{Mean} - \text{Magnitude} \quad b = \text{Mean} + \text{Magnitude} \quad (45.14)$$

The standard deviation can then be calculated by:

$$\sigma = \sqrt{\frac{1}{12}(b-a)^2} = \sqrt{\frac{1}{12}(2 \cdot \text{Magnitude})^2} \quad (45.15)$$

### Weibull distribution:

This distribution *RndWeibull* is often used to represent wind speed distributions. Its probability density function with the parameters  $\alpha$  (*Shape*) and  $\beta$  (*Scale*) is:

$$f(t) = \begin{cases} 0 & \text{for } t < 0, \\ \frac{\alpha}{\beta} \left(\frac{t}{\beta}\right)^{\alpha-1} \exp[-(\frac{t}{\beta})^\alpha] & \text{for } t \geq 0 \end{cases} \quad (45.16)$$

Mean and standard deviation of this distribution are calculated by:

$$\mu = \beta \Gamma(1 + \frac{1}{\alpha}), \quad \sigma = \sqrt{\beta^2 \left[ \Gamma\left(1 + \frac{2}{\alpha}\right) - \Gamma^2\left(1 + \frac{1}{\alpha}\right) \right]}, \quad (45.17)$$

in which  $\Gamma(x)$  is the Gamma function:

$$\Gamma(x) = \int_0^{\infty} t^{x-1} e^{-t} dt \quad \text{for } x > 0. \quad (45.18)$$

Parameter values of this distribution, in order to represent a wind speed distribution in a coastal area, might be  $\alpha = 2$  and  $\beta = 9$ . For those values, the mean is  $\mu = 7.976 \text{ m/s}$  and the standard deviation is  $\sigma = 4.169 \text{ m/s}$ .

#### 45.2.1.3 Transformed Distribution

This distribution (*RndTransformed*) can be used to transform any distribution using a Wind Power Curve (*TypPowercurve*, see Section 49.3.2).

Figure 45.2.1 shows the process of transforming a distribution. In this example a Weibull distribution (see upper left of the figure) is transformed with a wind power curve (upper right of the figure). The result in form of a cumulative distribution function is shown in the lower plot of the figure. Since wind speeds higher than 15 m/s lead to constant maximum power production of the wind turbine, a step can be seen at 2.5 MW (rated power) of the cumulated distribution curve.

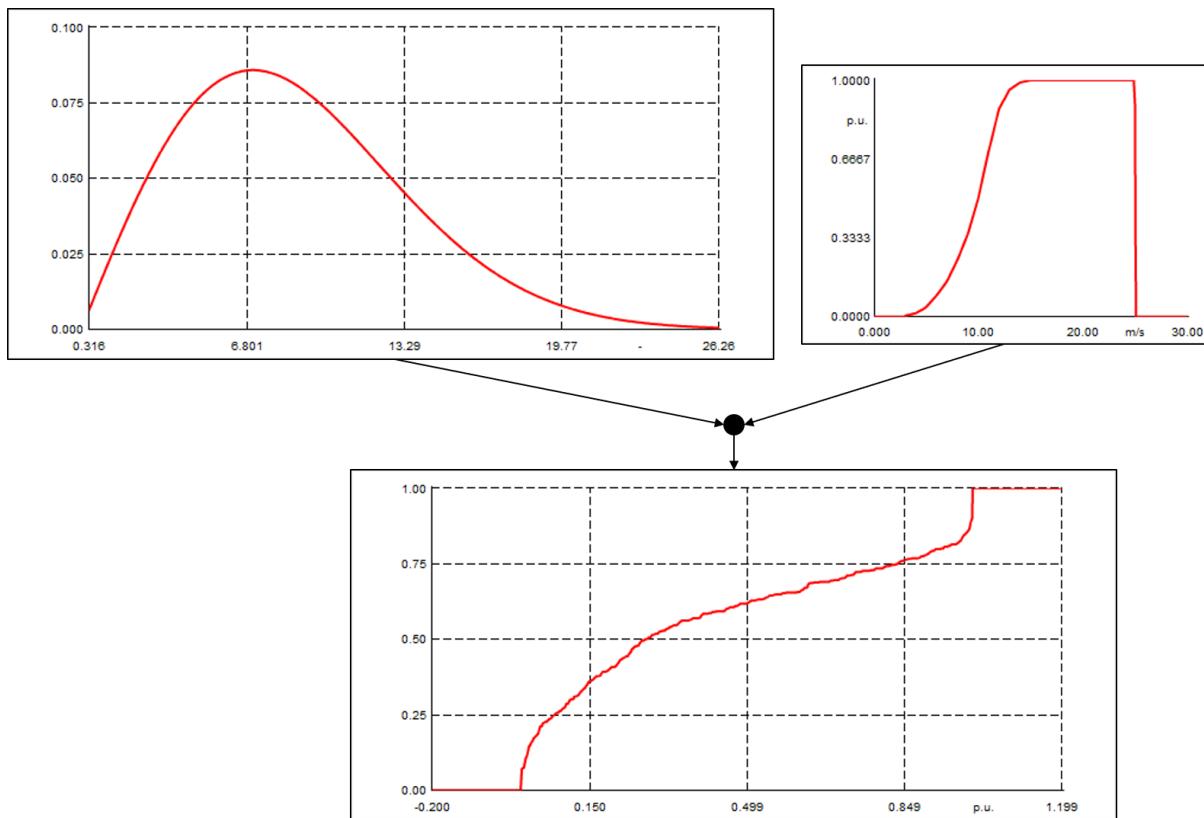


Figure 45.2.1: Process of transforming a distribution

#### 45.2.1.4 Distribution Based on Characteristics

The *Distribution Based on Characteristics* object (*RndCha*) estimates from an assigned characteristic a distribution. Two estimation methods are available, which are described in Section [45.2.5](#).

## 45.2.2 Modelling Dependencies

In many cases, random quantities in power systems are not independent, but have some underlying dependencies.

- Wind turbines which are not too far from each other.
- PV generation units which are located close together.
- Consumption of different households.

Correlations describe the linear dependency structure. Given some correlations, the space of possible couplings of marginals leading to these correlations is still infinite. A further notion is required in order to be able to determine dependency structures fully:

A  $d$ -dimensional copula is the joint cumulative distribution function of some random vector on the unit cube  $[0,1]^d$  with uniformly distributed marginals. In particular, a copula fully describes the distribution of a random vector on the unit cube with uniformly distributed marginals.

More generally, let  $X = (X_1, \dots, X_n)$  be a random vector. Let  $F_1, \dots, F_n$  be the corresponding distribution functions of the marginals. Then the random variables  $F_1(X_1), \dots, F_n(X_n)$  are uniformly distributed on  $[0,1]$  and the joint cumulative distribution function of  $F_1(X_1), \dots, F_n(X_n)$  is called the copula of  $(X_1, \dots, X_n)$ .

An elliptic copula, which is used in *PowerFactory*, is the dependency structure of elliptic multidimensional distributions.

In order to draw random values from a random vector  $X = (X_1, \dots, X_n)$  of given marginals and a dependency structure defined by an copula coming from an elliptic multidimensional distribution  $Y = (Y_1, \dots, Y_d)$ , the following procedure is chosen:

- Generate random sample of the elliptic multidimensional distribution  $Y = (Y_1, \dots, Y_d)$ .
- Transform the vector  $Y$  to coupled uniform distributions on  $[0,1]^d$ :  $U_i = F_{Y_i}(Y_i)$ .
- Transform the vector  $U$  of uniforms back to the desired distribution:  $X_i = F_{X_i}^{-1}(U_i)$ .

Therefore, the following copulas are provided in *PowerFactory*.

- The **Gaussian Copula** is the elliptic copula corresponding to multidimensional normal distributions.
- The **t-Copula** is the elliptic copula corresponding to multidimensional t-distribution.

The difference between these two Copulas is mainly defined by the tail dependence:

Let  $X_1, X_2$  be two random variables defined on a common probability space. Let  $F_1, F_2$  be the distribution functions respectively. Set

$$\lambda_u = \lim_{q \rightarrow 1} P[X_1 \geq F_1^{-1}(q) | X_2 \geq F_2^{-1}(q)],$$

provided that the limit  $\lambda_u \in [0,1]$  exists.  $\lambda_u$  is called the coefficient of upper tail dependence of  $X_1$  and  $X_2$ . If  $\lambda_u = 0$ ,  $X_1$  and  $X_2$  are called asymptotically independent in the upper tail. Otherwise, they are called asymptotically dependent in the upper tail.

Where random variables are coupled according to the **Gaussian Copula**, the upper and lower tails are asymptotically independent.

Conversely, the components of a distribution coupled according to **t-Copula** are asymptotically dependent in the tails, where

- with increasing “Degree of Freedom” the tails become more and more independent and
- with increasing correlation factors, tails become more dependent.

#### 45.2.2.1 Correction of non-positive definite correlation matrices

Usually, when estimating correlations from given data, it is not guaranteed that the resulting matrix is nonnegative definite. Since it is a pre-requirement for applying elliptic copula, a correction-algorithm is provided, which calculates the nearest (Euclidean distance) positive definite matrix to some given matrix.

As matrices with eigenvalues 0 introduce numerical instabilities, the user may give some lower bound for the eigenvalues of the resulting matrix. In this case, the algorithm calculates the nearest (Euclidean distance) positive matrix satisfying the lower bound of the eigenvalues. The implemented algorithm is iterative and stops on reaching some allowed error threshold or after a given maximum number of iterations. As the user may want to keep some entries of the correlation fixed, the algorithm supports keeping specified entries fixed. In this case, there might not exist a nearest positive definite matrix, and the algorithm will then return no solution.

#### 45.2.3 Probabilistic Analysis Methods

The Probabilistic Analysis supports the following analyses probabilistically:

- Load Flow Calculation
- Optimal Power Flow

Two main algorithms, each offering different advantages, have been implemented:

- Monte Carlo method
- Quasi-Monte Carlo method

#### 45.2.3.1 Random Number Generation

The random number generation is according to the commonly used Mersenne Twister pseudorandom number generator where the Seed can be defined within the Probabilistic Analysis Command.

#### 45.2.3.2 Monte Carlo Method

The Monte Carlo method draws individual samples randomly, as Figure 45.2.2a shows. Thus, there are some regions with more and others with less information.

The Monte-Carlo method is a way to calculate numerically the expectation of some function  $f : \mathbb{R}^d \rightarrow \mathbb{R}$  which is subject to random input  $X$ , with  $X$  taking values in  $\mathbb{R}^d$ . It generates  $N$  identically and independently distributed samples of  $X : X_1; \dots; X_N$ . Then the integral / expectation is approximated by:

$$E[f(X)] = \int f(X) dP \approx \sum_{n=1}^N f(X_n) \quad (45.19)$$

where the convergence rate is of order

$$O(1/\sqrt{N}) \quad (45.20)$$

Generally this means that in order to get one digit more precision, the Monte-Carlo method requires a factor 100 more samples.

An important property of the Monte-Carlo Method is that the rate of convergence does not depend explicitly on the dimension  $d$  of the underlying space. Related to network models, this means that the convergence is independent of the actual number of network elements.

When running Probabilistic Analysis according to the Monte-Carlo method, various well known statistic parameters are automatically generated (see Section 45.2.4 for detailed information). These are:

- Mean,
- Standard deviation,
- Maximum,
- Minimum,
- Variance,
- 1st - 5th Moment and
- confidence interval for mean and standard deviation.

#### 45.2.3.3 Quasi-Monte Carlo Method

The Quasi-Monte Carlo method uses special sequences, in order to cover the space more uniformly (see Figure 45.2.2b). The rate of convergence of a Quasi-Monte-Carlo simulation is given by:

$$O(\ln(N)^d/N) \quad (45.21)$$

which for a large number of samples  $N$  behaves like  $1/N$ . Thus, in order to get one digit more precision, 10 times more samples are required. Using this method, it has to be mentioned that the convergence rate now depends on the dimension  $d$ , meaning the number of relevant elements within the network.

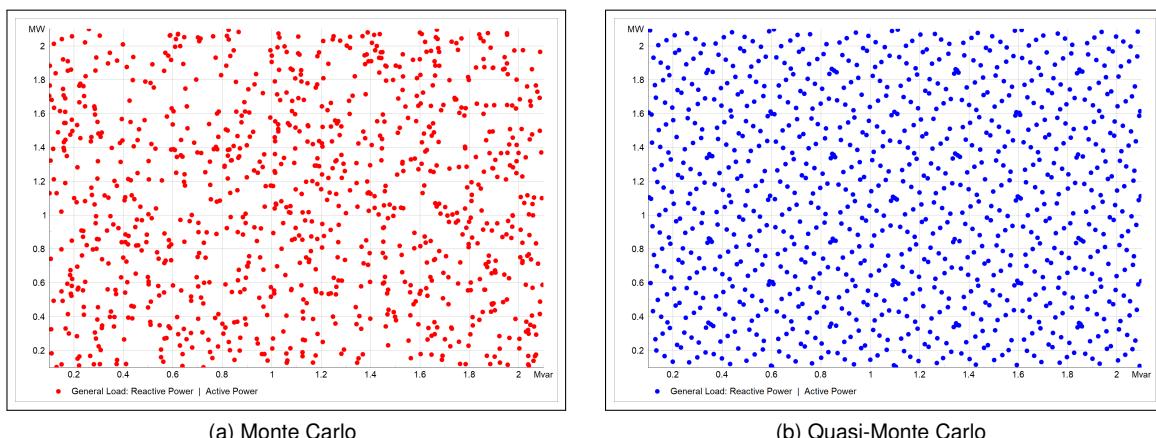


Figure 45.2.2: Comparison of methods

#### **45.2.3.4 Comparison of Methods**

The difference between the methods lies in the sequence of samples as shown in Figure 45.2.2. Due to that, the Quasi-Monte Carlo method usually converges faster than the Monte Carlo method, but because of the dependence of individual samples the confidence interval estimation is not possible (refer to Section 45.2.4 for more information about the confidence interval).

<b>Method</b>	<b>MC</b>	<b>QMC</b>
Calculation of confidence intervals	Yes	No
Convergence rate	$1/\sqrt{N}$	$1/N$
One digit more precision: factor $x$ more samples	$x=100$	$x=10$
Convergence depends on dimension	No	Yes

Table 45.2.1: Comparison between Monte Carlo (MC) and Quasi-Monte Carlo (QMC)

## 45.2.4 Statistics

Statistics are available for two different use cases:

- Statistical results files: the statistics, defined in the Probabilistic Analysis command, will be calculated and stored during Probabilistic Analysis in the *Statistical Result* file.
  - Plots: statistics may be plotted for variables stored in the *Samples Result* file.

Let  $x = x_1, \dots, x_n$  be the observed samples of a random quantity  $X$  during Monte Carlo or Quasi-Monte Carlo Simulation. The statistics are defined in the following way:

- **Mean:**

The empirical mean  $\bar{x}$  of a variable is calculated by:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i \quad (45.22)$$

- **Variance:**

The empirical variance  $s^2(x)$  of a variable is calculated by:

$$s^2(x) = \frac{1}{n-1} \sum_{k=1}^n (x_k - \bar{x})^2 \quad (45.23)$$

- **Standard deviation:**

The empirical standard deviation  $s(x)$  is calculated by:

$$s(x) = \sqrt{s^2(x)} \quad (45.24)$$

- **Moments (up to fifth order):**

The empirical  $l$ -th moment  $m_l(x)$  is calculated by:

$$m_l(x) = \frac{1}{n} \sum_{k=1}^n x_k^l. \quad (45.25)$$

- **Confidence interval of mean:**

General meaning: the probability, that the true value is in the confidence interval, is greater than  $1 - \alpha$  ( $\alpha$  = level), as the number of observations goes to infinity.

Usually, the true probability distribution of the random quantity  $X$  is unknown. Often, it is not even known whether it is normally distributed or not. In such cases the estimation of a confidence interval is possible, provided that the true probability distribution satisfies the central limit theorem.

Note: the central limit theorem relies on independent and identically distributed observations of the random quantity. This is not the case for the Quasi-Monte Carlo Method.

The values for the upper and lower levels both have to be greater than 0 and less than 50.

- **Confidence interval of standard deviation:**

See Confidence interval of mean, but in this case for the standard deviation.

- **Empirical probability:**

The empirical probability for  $X \in I$  for some given interval  $I$ :  $p(I,x)$  is calculated by:

$$p(I,x) = \frac{1}{n} \sum_{k=1}^n 1_{x_k \in I} = \frac{1}{n} \#\{x_k \in I : 1 \leq k \leq n\} \quad (45.26)$$

The following intervals are supported:

- Upper bounded:  $I = (-\infty, b]$ , meaning that  $b$  is inside the interval.
- Lower bounded:  $I = (a, \infty)$ , meaning that  $a$  is not inside the interval.
- Upper and lower bounded:  $I = (a, b]$ .

- **Maximum:**

The maximum value of the observed variable within the complete analysis.

- **Iteration with max. value:**

The sample number of the maximum value of the observed variable within the complete analysis.

- **Minimum:**

The minimum value of the observed variable within the complete analysis.

- **Iteration with min. value:**

The sample number of the minimum value of the observed variable within the complete analysis.

---

**Note:** Confidence intervals are not available for a Quasi-Monte Carlo simulation (see Section 45.2.3.3).

---

## 45.2.5 Distribution Estimation

Two estimation methods are available, which are explained in the following subsections.

- Bootstrapping
- Histogram

### 45.2.5.1 Bootstrapping

Bootstrapping is commonly used to re-sample the data in order to refine the estimation of the underlying distribution. The approach used here is the most basic one, which does not re-sample and draws randomly from the given sample.

In the case of a characteristic that means that all values from this characteristic are converted into a cumulative distribution function, from which random numbers are drawn during the analysis.

This process of drawing samples is illustrated in Figure 45.2.3. A number between zero and one is randomly drawn. The corresponding x-value is then taken for the analysis.

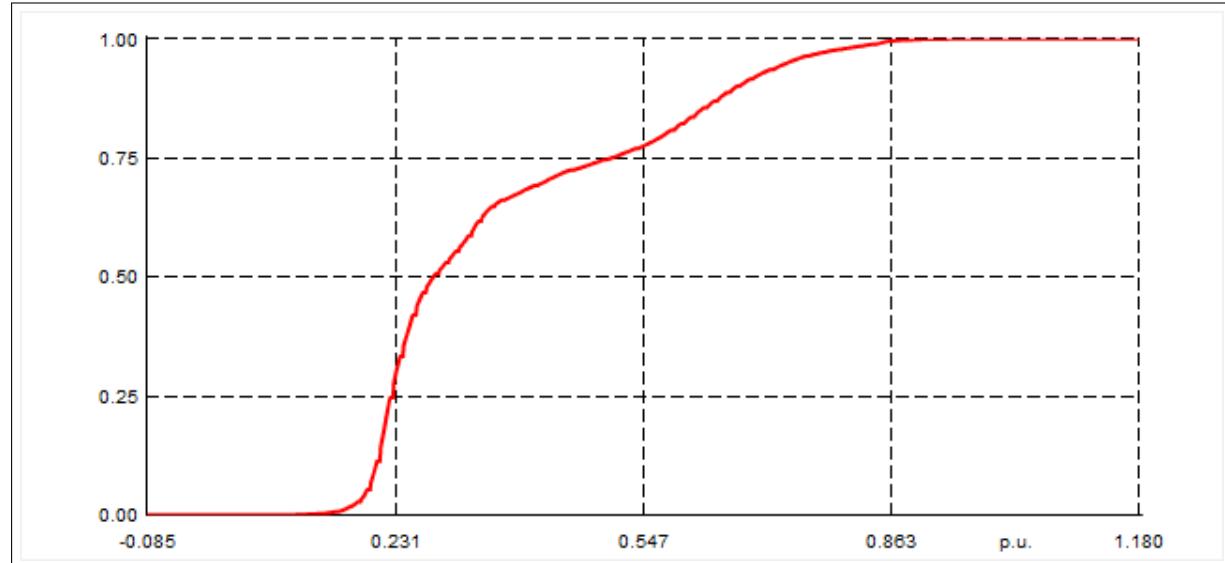


Figure 45.2.3: Bootstrapping - Cumulative distribution function

For photovoltaic power curves, the bootstrapping method may be more suitable than the histogram, since this method uses each single value from the characteristic and therefore represents night hours, in which the production of PV systems is zero, more realistic. If the histogram method were to be used, it would be rather improbable that zero would be drawn from the sample.

Mean and Standard deviation of the distribution are calculated by:

$$\mu = \frac{1}{n} \sum_{i=1}^N x_i, \quad \sigma = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2} \quad (45.27)$$

### 45.2.5.2 Histogram

For the generation of a histogram from a sample (with  $n$  = number of samples), either the number of bins  $k$  or the bin width  $h$  has to be defined. Note that the bin width is fixed, once the number of bins is determined (and vice-versa), if the first bin begins at the first sample and the last bin ends at the last sample. For the definition of the bins the following options are available:

**User-Defined:** the user is able to specify either the number of bins or the bin width.

**Automatic:** will choose the maximum number of bins obtained using the Rice Rule and the Freedman-Diaconis choice.

**Rice Rule:**  $k = 2 * n^{1/3}$

**Scott's Rule:**  $h = \frac{3.5 * \sigma}{n^{1/3}}$ , where  $\sigma$  is the standard deviation of the sample.

**Freedman-Diaconis choice:**  $h = \frac{2 * IQR}{n^{1/3}}$ , where  $IQR$  is the interquartile range of the sample. The interquartile range is the difference between the first and the third quartile.

Figure 45.2.4 shows the probability density function of the histogram with 33 bins and a bandwidth of 0,02986 per unit. From this graphic the frequency of values within a specific interval can be extracted. The histogram can also be represented as a cumulative distribution function, which is obtained by integrating the probability density function.

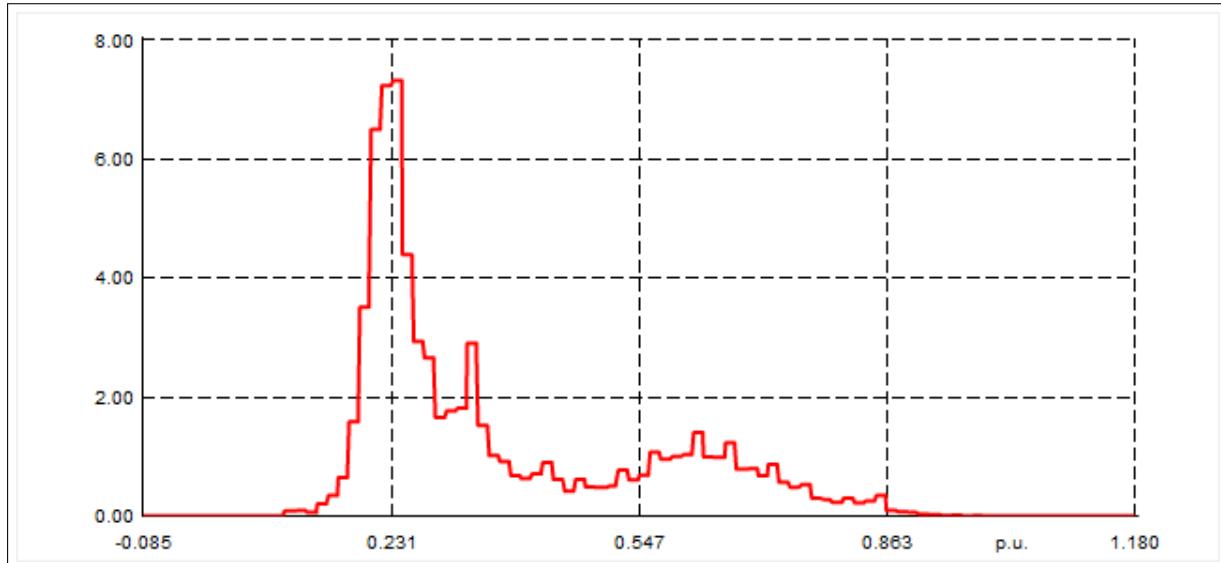


Figure 45.2.4: Histogram - Probability density function

During the probabilistic analysis a value is randomly drawn from the sample. The algorithm checks to which bin this value belongs and draws another value from within that bin, which is then used for the analysis.

## 45.2.6 Distribution Fitting

The goal of distribution fitting is to determine the distribution of some random quantity based on some statistical data. In *PowerFactory*, two fitting procedures are supported:

- Gram-Charlier A series
- Edgeworth expansion

Both fittings calculate distributions based on given moments  $m_1, \dots, m_k$  of the underlying distribution.

Note, that in both cases, the outcome is not necessarily a distribution function / probability density, but just an approximation of the true distribution function / probability density.

In both cases, the approximate cdf / pdf are given as follows:

$$pdf : f(x) = \sum_{j=0}^n c_j \phi^{(j)}(x) \quad (45.28)$$

$$cdf : F(x) = \sum_{j=0}^n c_j \Phi^{(j)}(x), \quad (45.29)$$

where  $\phi$  is the probability density of the standard normal distribution, and  $\phi^{(j)}$  is  $j$ -the derivative of  $\phi$ . Analogously  $\Phi$  is the cumulative distribution function of the standard normal distribution, and  $\Phi^{(j)}$  is  $j$ -the derivative of  $\Phi$ . The coefficients  $c_j$  are different for the different expansions, and depend on the given moments of the underlying distributions. The order of the expansion is  $n$ .

The Gram-Charlier A series is based on the series representation of the characteristic function of a distribution. In contrast, although having the same summands, the Edgeworth expansion uses a different ordering of the summands which allows (with the help of the central limit theorem) for a more precise control of the fitting error.

A rigorous treatment of series expansions for distributions may be found e.g. in [42] or [50].

## 45.3 Object Settings

In the following subsections the settings of the commands and objects related to the Probabilistic Analysis are described.

### 45.3.1 Distributions

#### 45.3.1.1 Assignment of distributions

**Manual assignment of distributions:** the manual assignment of distributions to parameters is possible in two different ways:

- **Object dialog:** right-click on a parameter and select *Add project distribution...* and then choose one of the provided distributions. Already assigned distributions can be edited by right-clicking on the parameter and selecting *Edit distribution(s)...*
- **Class Filters (e.g. Network Model Manager):** select one or more cells (of one column) of a desired parameter, right-click the selection and choose *Characteristic/Distribution → Add project distribution...* From the provided list, select the desired distribution.

**Automatic assignment of distributions based on characteristics:** if characteristics have been assigned to parameters, distributions can be estimated automatically using the *Distribution Estimation* command (see Section 45.3.3).

If a distribution has been assigned to a parameter, its input field or the corresponding cell in the browser will be coloured, depending on the user settings (see Section 7.8).

#### 45.3.1.2 Application of random values

For all available distributions two options for the *Application of random values* are available:

- **Usage:** in general, all types of distributions can be applied to parameters with a selectable *Usage*. The available options are explained below. The description also contains a formula, in which  $p$  is the value of the parameter,  $d$  the value of the distribution and  $r$  the resulting value.
  - **Multiplicative in %:** drawn percentage value is multiplied with the set parameter value.

$$r = \frac{d}{100} \cdot p \quad (45.30)$$

- **Multiplicative:** drawn value is multiplied with the set parameter value.

$$r = d \cdot p \quad (45.31)$$

- **Absolute:** drawn value (absolute value) is directly applied to the parameter.

$$r = d \quad (45.32)$$

- **Additive in %:** from the drawn percentage value and the set parameter value an absolute value is calculated, which is added to the set parameter.

$$r = \frac{d}{100} \cdot p + p \quad (45.33)$$

- **Additive:** from the drawn value and the set parameter value an absolute value is calculated, which is added to the set parameter.

$$r = d + p \quad (45.34)$$

- **Based on:** the drawn value from the distribution can be applied to the “Original value” (the entered value) or the “Characteristics value” (actual value).

---

**Note:** If *Usage* is set to *Absolute*, the *Based on* option is disabled.

---

#### 45.3.1.3 Standard Distributions

The following predefined distribution functions are available:

- Bernoulli distribution (*RndBernoulli*)
- Exponential distribution (*RndExp*)
- Geometric distribution (*RndGeo*)
- Log-normal distribution (*RndLognormal*)
- Normal distribution (*RndNormal*)
- Uniform distribution (*RndUnif*)
- Weibull distribution(*RndWeibull*)

For more information about the theoretical background of these distributions (like adjustable parameters), refer to Section 45.2.1.2.

#### 45.3.1.4 Special Distributions

**Transformed distribution (*RndTransformed*):** two references have to be assigned to this distribution:

- *Distribution:* any kind of distribution (*Rnd\**) can be assigned.
- *Transformation:* a *Wind Power Curve* (*TypPowercurve*) can be assigned.

Further information about this distribution can be found in Section 45.2.1.3.

**Discrete finite distribution (*RndFinite*):** in order to define desired states, first the *Number of states* has to be specified. By changing this number, rows are appended or deleted from the table on the right. In this table values and their weights can be defined. Depending on the weights the probability is automatically calculated, as described in Section 45.2.1.2.

#### 45.3.1.5 Distribution based on characteristics

In the following, the input options of the *RndCha* object are described:

**Characteristics:** reference to a time characteristic (*ChaTime*, *ChaProfile* and *ChaVec* with time trigger) or objects, which internally have a characteristic, like PV Systems (*ElmPvsys*) with a Solar Calculation model or MV (*ElmLodmv*) or LV loads (*ElmLodlv*) with a Consumption Profile. A characteristic can be assigned by clicking on and *Select...* and choosing a desired object of the previously mentioned classes.

**Time Range of data:** two options are available:

- *Take whole range*: the complete time range of the characteristic will be used.
- *Restrict time range*: a Time Range object (*SetTimerange*) can be assigned, in order to specify the time range of the data. For more information about this object refer to Section [45.3.1.7](#).

**Distribution Estimation:** one of the methods described in Section [45.2.5](#) can be selected.

---

**Note:** If a PV System with Solar Calculation Model has been assigned as characteristic, the Bootstrapping method will automatically be applied.

---

#### 45.3.1.6 Parameter Distribution - Reference

This reference (*RndRef*) is the equivalent to the Characteristic Reference (*ChaRef*). It connects the distribution with the element's parameter.

#### 45.3.1.7 Time Range

This object (*SetTimerange*) can be assigned to a *RndCha* object, in order to convert a specific time range from a characteristic into a distribution. Within this object (*SetTimerange*) a start and end time (date and time) can be defined.

### 45.3.2 Dependencies

Dependencies, representing for example the correlation in terms of power infeed of locally clustered wind turbines, can be defined in different ways:

- Through Single Line Diagram  
It is possible to multi select several elements within the SLD choosing *Network Groupings* → *Distribution correlation* → *New...*
- Through Element Filter  
Within the Element Filter, it is also possible to mark several elements *Network Groupings* → *Distribution correlation* → *New...*

There is the possibility to choose between the correlation definitions, explained within the following subsections.

### 45.3.2.1 Elliptic Copula - equally

Within this element the correlation between elements and/or parameters can be defined.

Once several elements are defined according to Section 45.3.2, the table containing elements is filled. In this stage, all parameters of the defined elements, where distributions are defined, will be correlated. The **column “Dists. (Distributions)**” shows how many parameters are considered for every element within the table. This can be enhanced by choosing a parameter within the **column “Parameter”**. In this case, the correlation is only valid for this single parameter.

The correlation itself can be entered in the field **“Correlation”** and is valid for all selected Parameters. Note, that when entering a correlation value lower than 1, the Copula type, “Gaussian” or “t-copula”, according to Section 45.2.2 can be selected.

### 45.3.2.2 Elliptic Copula - detailed

Within this correlation element, coupled parameters can be defined, where a correlation parameter can be entered for each individual pair of parameters, given within one row. This correlation can be set within the table column **“Correlation”**. In addition, one **“Default correlation factor”** can be entered to define the correlation between the parameters, where no explicit correlation is given within the table. Again, it is possible to choose between the Copula type “Gaussian” or “t-copula” according to Section 45.2.2.

On the Basic options page, there is also the possibility to **“Show Matrix”**, which is the overall correlation matrix, containing every parameter to be correlated.

On the **“Copula Correlation Matrix”** page, there are several options to define the automatic correction of Copula correlation matrix (if option “Automatically correct to pos. def. matrix” is enabled). The usage of these parameters is explained in Section 45.2.2.1. The option **“Keep specified entries fixed”** enables the specified correlation values defined within the Basic Options to be fixed, when the auto correction tries to find a proper Copula Correlation matrix by adapting the non-specified entries.

### 45.3.2.3 Elliptic Copula - characteristics

This Copula element allows the user to automatically define copulas according to time characteristics. To estimate the correlation between different characteristics, the user can tick the corresponding option within the Distribution Estimation command, or link different characteristics manually within the Basic Options page of the “Elliptic Copula - characteristics” Element. Further available options are explained below:

- Correlation of parameters using same cha.:  
The user can define whether all parameters using the same characteristics shall be “Fully correlated” or correlated with a “User-defined” correlation factor, which can be entered within the “Characteristics” table on the same page.
- Restrict time range:  
This setting allows the user to define a restricted time range out of which the correlation will be estimated. More information about the time range object can be found in Section 45.3.1.7.
- Copula correlation Matrix:  
These options refer to the separate page “Copula Correlation Matrix” and are explained below.
- The button **Show correlation matrix of characteristics**:  
This button allows the user to access the correlation matrix as estimated by this “Elliptic Copula - characteristics” Element.

The page “Copula Correlation Matrix” offers the possibility to define the automatic correction of the copula correlation matrix. The procedure of this is explained in Section 45.2.2.1.

### 45.3.3 Distribution Estimation Command

This command *Distribution Estimation* (*ComRndest*) can be opened by clicking the *Distribution Estimation* button  available in the Probabilistic Analysis toolbar. It creates *Distribution based on characteristics* objects, which convert time characteristics or specific time ranges of those into distributions. For more information about distributions based on characteristics and estimation methods, refer to Sections [45.2.1.4](#) and [45.2.5](#).

#### Characteristics

Within the first field called *Characteristics* the user can select which characteristics are to be converted into distributions. There are two options:

- **All active:** for all active<sup>1</sup> characteristics, distributions are created and assigned. If this option is selected, there must be active characteristics in the project. Otherwise an error message will occur when the command is executed.  
The button **Show input** shows all active characteristics. Apart from time characteristics (*ChaTime*, *ChaProfile* and *ChaVec* with time trigger), objects which internally have a characteristic, like PV Systems (*ElmPvsys*) with a Solar Calculation model or MV (*ElmLodmv*) or LV loads (*ElmLodlv*) with a Consumption Profile, will also be listed.
- **User-defined selection:** by clicking on  and *Select...*, a browser window with all active characteristics will open, from which desired characteristics can be chosen. By confirming with **OK**, a set (*SetSelect*), containing the selected characteristics, will automatically be created in the active study case. If the Distribution Estimation command is executed, all parameters which refer to the characteristics stored in this selection, will obtain a distribution.  
By clicking on  and *Select Parameter...* desired parameters, to which a characteristic has been assigned, can be chosen. This selection is stored in a set (*SetSelect*), as well. For all parameters contained in this set, distributions will be created and assigned to the corresponding parameters.

Two *Estimation methods* are available:

- Bootstrapping
- Histogram

The technical background information for these two methods can be found in Section [45.2.5](#).

By clicking the button **Edit all references and distributions based on cha.**, a browser window will open that shows all available references and distributions based on characteristics. This browser allows the user to modify or delete existing distributions and references.

#### Time range

By activating the option *Restrict time range* a Time Range (*SetTimerange*) object can be assigned to the command, which enables the possibility to convert only a specific time range from a characteristic into a distribution. The reference to the time range object is applied to the *Distribution based on characteristics* object. For more information about the time range object refer to Section [45.3.1.7](#). If the option *Restrict time range* is deactivated the complete time range of the characteristic is used.

#### Advanced Options

On this page the *Assignment of created distributions to parameters* can be defined. Three options are available:

- **Do not assign:** distributions are created but not assigned to the parameters.
- **Ignore parameters which already have distributions assigned:** new distributions will be created, but for all parameters which already have a distribution assigned no new references will be created. For all remaining parameters new references will be created and assigned to the new distributions.

<sup>1</sup>A characteristic is called *active*, if it has been assigned to a parameter within the network.

- **Always assign and replace existing references:** new distributions are created. Old references are replaced by new ones, which refer to the newly created distributions.

If the check box *Estimate correlations* is selected, two options are available:

- **Create new correlation:** a new correlation (Elliptic Copula - characteristics (*ElmCopellcha*), see Section 45.3.2.3) will be created and all characteristics which have been selected on the *Characteristics* page, will be assigned to this copula.
- **Add to existing correlation:** an existing copula (*ElmCopellcha*, see Section 45.3.2.3) can be selected, to which the characteristics defined on the *Characteristics* page will be added.

#### 45.3.4 Probabilistic Analysis Command

The Probabilistic Analysis Command (🎲) is the main Command in Probabilistic Analysis function within *PowerFactory*. These are the settings for the command:

##### Basic Options:

- Analysed calculation:  
Within this setting, it is possible to choose the calculation type to be Load Flow or Optimal Power Flow.
  - **Load Flow:**  
Load Flow calculation is used to analyse power systems under steady-state non-faulted conditions. All its available calculation settings are considered by the Probabilistic Analysis. More information about the load flow and the settings of this command are available in Chapter 25.
  - **Optimal Power Flow**  
The Probabilistic Analysis will analyse the Optimal Power Flow probabilistically. The Optimal Power Flow (OPF) module optimises a certain objective function (e.g. Maximisation of reactive power reserve) in a network whilst fulfilling equality constraints (the load flow equations) and inequality constraints (i.e. generator reactive power limits). All its settings are considered by the Probabilistic Analysis. More information about OPF and its settings are available in Chapter 39.  
The Probabilistic Analysis of the OPF can be used in order to determine the distributions of controls for a dispatch strategy considering a given objective.
- Method:  
As described in 45.2.3, it is possible to choose between the Monte Carlo (45.2.3.2) or Quasi-Monte Carlo (45.2.3.3) methods.
- Max. number of samples:  
Within this field, the number of samples can be defined.
- Pointer to Statistical results:  
According to the chosen Method and Calculation type settings, the relevant Statistical results file is given here. For more information about results files of the Probabilistic Analysis refer to Section 45.3.7.
- Pointer to Sample results:  
According to the chosen Method and Calculation type settings, the relevant Sample results file is given here. For more information about results files of the Probabilistic Analysis refer to Section 45.3.7.

##### Recorded Statistics

- Statistics recorded for all variables:  
Within this table, all statistics to be recorded can be defined. Please note that the available

statistics depend on the simulation method. A list of available statistics together with possible settings is given in Section 45.2.4. The settings can be entered by editing the Statistics. The Button “Set statistics to default” resets the “Statistics recorded for all variables” selection to contain the default values.

- Statistics recorded for specific variables:

An additional option is to record statistics only for specific variables. Therefore, it is possible to choose one single object or a class of objects within the column “Variables”. If the variable consists of a class, a filter can be defined according to Section 45.3.7. The statistics to be recorded has to be chosen within the column “Statistics”.

### Advanced Options

- Random number generation:

The Seeding type can be selected to be “Automatic” or “User defined”. When set to be “User defined”, a Seed value between 0 and  $(2^{32} - 1)$  can be entered. This makes it possible to exactly reproduce all samples drawn as described in Section 45.2.3.1.

- Output per sample:

If turned Off, no output will be shown during calculation of samples. Otherwise the output will contain the information as described within the title.

- Consider distribution correlations:

This checkbox determines whether or not defined correlations will be considered during Probabilistic analysis.

During the execution of Probabilistic Analysis command, the user can interrupt the calculation using this button:  (Stop Probabilistic Analysis).

### 45.3.5 Continue Probabilistic Analysis

Within the *Continue Probabilistic Analysis* command ()<sup>2</sup>, the user can enter the Max. number of samples, which should be the resultant number of samples on completion of the simulation. The currently executed and stored number of samples is given in the information field “Number of calculated samples”.

### 45.3.6 Probabilistic Analysis Player

**Investigation of worst and mean cases** The *Probabilistic Analysis Player*  can be used to reload the results of a specific sample. This might be useful, for example, to analyse a sample in which a specific line is overloaded. The sample number can be extracted from the statistics results file. After the execution of this command the results can then be observed for example in the single line diagram or on the Flexible Data page of the Network Model Manager.

### 45.3.7 Results File Handling

As for many other modules in *PowerFactory*, result variables have to be defined for the Probabilistic Analysis<sup>2</sup> as well, before the command is executed. For this analysis a distinction has to be made between a:

- *Sample* results file and a
- *Statistical* results file.

<sup>2</sup>Some standard parameters for selected object classes have already been predefined.

**Sample results:** for the *Sample result* file, result parameters of the load flow calculation or the OPF analysis like the “loading” of a line or the “active power dispatch” of a generator can be selected.

**Statistical results:** such parameters can also be selected for the *Statistical result* file. However, for this, statistics (see Section 45.2.4) like the mean value or the standard deviation are recorded (dependent on the settings on the *Recorded Statistics* page of the Probabilistic Analysis command, see Section 45.3.4). The results files (*ElmRes*) for the Probabilistic Analysis are stored within a folder called *Probabilistic assessment* inside the study case.

Within each iteration of the Probabilistic Analysis, samples are randomly drawn from the assigned distributions and a Load Flow or Optimal Power Flow is executed. The results of each iteration for the parameters to be recorded are stored within the *Samples* results file. After the maximum number of samples has been reached, statistics are calculated for the selected variables and stored within the *Statistical result* file.

Dependent on the *Calculation Type* and the *Method* selected in the Probabilistic Analysis as well as the Load Flow or Optimal Power Flow command, different results files are used. For each of them, variables can be defined individually.

Result variables can be defined by clicking the *Edit Result Variables* button  from the Probabilistic Analysis toolbar. A selection dialog will appear, in which the type of results file (Statistical or Sample results) to be edited can be selected. After the confirmation of this dialog, a browser window opens, which shows the content of the results file that has been selected in the Probabilistic Analysis command. Information about the Variable Selection can be found in Section 19.3.

For Variable Selections that are applied to a complete class, a filter (see Chapter 11: [Data Manager and Network Model Manager](#), Section 11.3.3) can be defined. This makes it possible for example to record parameters for elements of that class which are inside a specific area.

## 45.3.8 Representation of results

### 45.3.8.1 Single Line Diagram

After the execution of the Probabilistic Analysis, statistics can be observed in the single line diagram in two different ways.

#### Result Boxes:

By right-clicking on a result box, different predefined statistics like the mean value and the standard deviation for node voltages and branch flows can directly be selected (e.g. *Format for Nodes* → *L-L Voltage, Angle* or *Format for Edge Elements* → *Branch flow*).

Of course every other variable recorded in the Statistical results file can be displayed in the result box as well, by clicking on *Edit Format for ...*.

#### Colouring:

Statistics can also be analysed by colouring the single line diagram as shown in Figure 45.3.1.

The use of colouring in network diagrams is described in Section 10.3.10.1, and more detail about the use of colours and colour palettes can be found in Section 4.7.8.

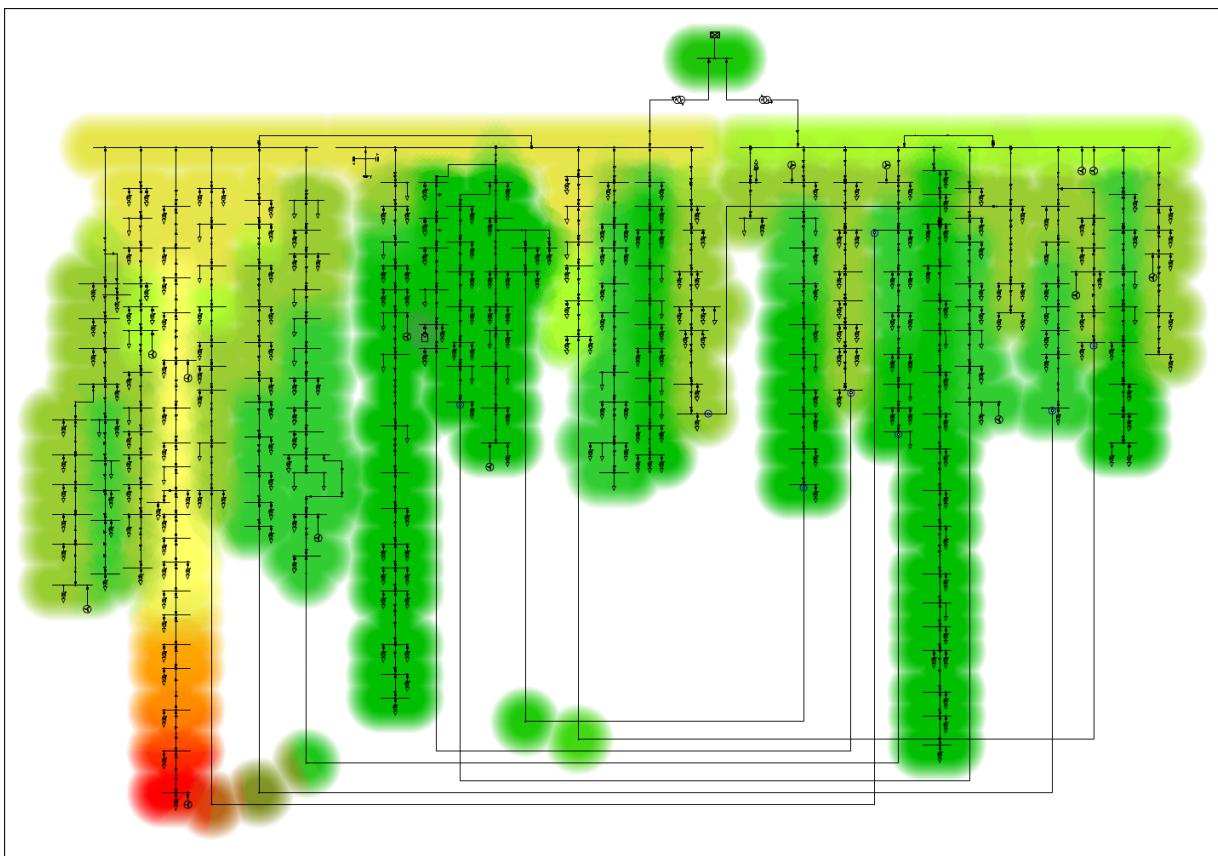


Figure 45.3.1: Colouring of statistics (mean value) of Load Flow quantities in the Single Line Diagram

The colouring can be enabled by clicking on the *Diagram Colouring...* button  and opening the corresponding Probabilistic Analysis page. Activate 3. Others and select *Results → Probabilistic Load Flow*. By clicking on **Colour Settings...**, the statistic to be used for the colouring as well as the colour gradation can be defined.

For *Statistical results*, the following options are available:

- Mean values;
- Maximum or minimum;
- Standard deviation;
- Empirical probability.

For more information about statistics, refer to Section 45.2.4.

Depending on the selected statistic, different colour gradations can be defined.

---

**Note:** In order to colour the diagram according to the Empirical probability, the corresponding statistic has to be recorded in the statistics results file (by default it is not recorded). E.g. if the *Empirical probability* for the interval *Lower unbounded* with an *Upper bound* of 0.95 has been recorded for the node voltage "m:u", the variable "m:u:empPrLow\_-Inf\_-0\_95" should exist and can therefore be used for the graphical analysis.

### 45.3.8.2 Network Model Manager

Statistics can be displayed on the Flexible Data page in the Network Model Manager. This makes it possible to sort parameter values according to their size or apply existing filter functions. By clicking on the *Variable selection* button , statistics and other variables can be chosen for the selected class to be displayed on the Flexible Data page. For more information regarding variable selections refer to Section 19.3.

### 45.3.8.3 Convergence of Statistics Plot

Choosing this plot, the convergence behaviour of single element's parameter can be analysed. It is possible to analyse the convergence behaviour of Mean and Standard deviation. In these cases, the plots will show the corresponding value as well as the confidence interval for a user defined percentage level. More information about statistics can be found in Section 45.2.4.

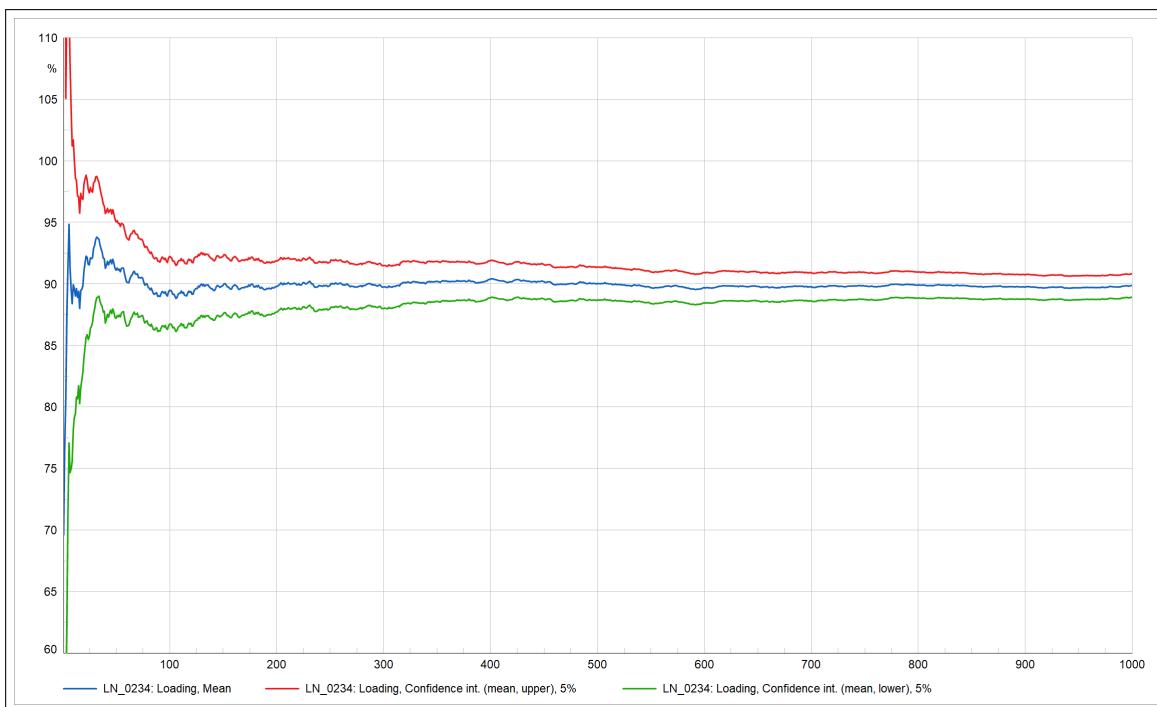


Figure 45.3.2: Convergence Plot of Line Loading Mean value (red) and upper (green) and lower (blue) 5% Confidence interval

### 45.3.8.4 Correlation Plot

The correlation gives the resulting correlation between two parameters. By default, two parameters of one element can be selected. However, if "Show x-Element in table" is selected, the correlation between two parameters of two different elements can be plotted.

### 45.3.8.5 Distribution Estimation Plot

The Distribution Estimation plot returns the resulting distribution of one sample result variable. For this purpose, the sample results file is analysed. The methods that are used are Bootstrapping and Histogram, as explained in Section 45.2.5, the only difference being that the input data is not a time characteristic but the result of all samples available after Probabilistic analysis. For Histogram method, the bin width is estimated automatically but can also be set manually by selecting

“User-defined” within the *Distribution estimation* dialog.

A Distribution Estimation plot is shown in Figure 45.3.3 (red curve), which is estimated from the samples of the recorded quantity. From the curve shape it can be supposed that the underlying distribution was a uniform one.

#### 45.3.8.6 Distribution Fitting Plot

The Distribution Fitting Plot gives the distribution fitting according to the methods, explained in Section 45.2.6. In context with these methods, an order of fitting can additionally be entered for fitting.

The blue cumulative distribution function of Figure 45.3.3 is the result of the Edgeworth Expansion (Ord. 4) distribution fitting method, which uses the statistical results of the same element parameter selected for the distribution estimation.

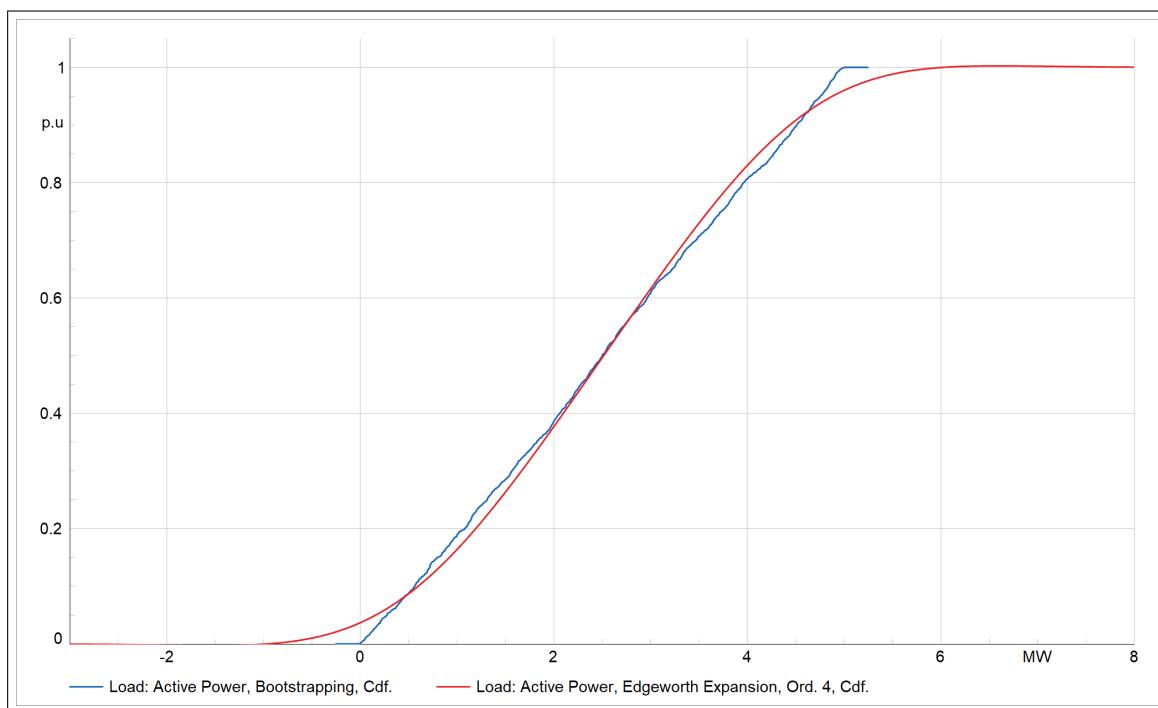


Figure 45.3.3: Cumulative distribution function. Comparison between Distribution Estimation - Bootstrapping (red curve) and Distribution Fitting - Edgeworth Expansion, Ord. 4 (blue curve)

#### 45.3.8.7 Load Probabilistic Analysis Results

Depending on the size of the network, the number of active distributions and correlations and the number of samples of the analysis, the Probabilistic Analysis might take a long time. Execution of other commands or changes in element parameters will cause the results of the Probabilistic Analysis to be reset. In order to avoid another execution of this analysis to obtain the same results, the *Load Probabilistic Analysis results* button can be used to reload an existing probabilistic results file. The results can then again be analysed in the single line diagram or the Network Model Manager.

---

**Note:** Users reloading earlier results should be aware that they may not be compatible with the current network state, if changes have been made since the analysis was executed.

---

# Chapter 46

## Reliability Analysis

### 46.1 Introduction

Reliability assessment involves determining, generally using statistical methods, the total electric interruptions for loads within a power system during an operating period. The interruptions are described by several indices that consider aspects such as:

- The number of customers N;
- The connected load, normally expressed in kW;
- The duration of the interruptions, normally expressed in h = 'hours';
- The amount of power interrupted, expressed in kW;
- The frequency of interruptions, normally expressed in 1/a = 'per annum';
- The repair times, which are normally expressed in h = 'hours';
- The probabilities or expectancies, expressed as a fraction or as time per year (h/a, min/a).

Network reliability assessment is used to calculate expected interruption frequencies and annual interruptions costs, and to compare alternative network designs. Reliability analysis is an automation and probabilistic extension of contingency evaluation. For such analysis, it is not required to pre-define outage events, instead the tool can automatically choose the outages to consider. The relevance of each outage is considered using statistical data about the expected frequency and duration of outages according to component type. The effect of each outage is analysed automatically such that the software simulates the protection system and the network operator's actions to re-supply interrupted customers. Because statistical data regarding the frequency of such events is available, the results can be formulated in probabilistic terms.

---

**Note:** Reliability assessment tools are commonly used to quantify the impact of power system equipment outages in economic terms. The results of a reliability assessment study may be used to justify investment in network upgrades such as new remote controlled switches, new lines / transformers, or to assess the performance of under voltage load shedding schemes.

---

This chapter deals with probabilistic Network Reliability Assessment. For information on *PowerFactory*'s deterministic Contingency Analysis, refer to Chapter 27 (Contingency Analysis).

The structure of this chapter is as follows:

- Technical Background, in Section 46.2;

- Model Setup for Reliability Assessment, in Section 46.3;
- Description of Reliability Assessment Calculation Command, in Section 46.4;
- Analysis of Reliability Assessment Results, in Section 46.5;
- Description of Loss of Grid Assessment, in Section 46.6.

The Reliability Assessment functions can be accessed by selecting the *Reliability Analysis* toolbar from the *Change Toolbox* ▾ icon. The Toolbox “Reliability Analysis” is shown in Figure 46.1.1.

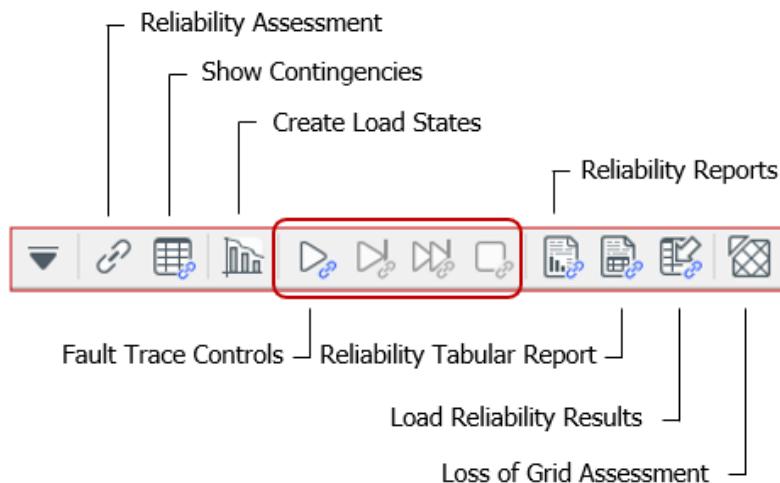


Figure 46.1.1: Reliability Analysis Toolbox

The basic user procedure for completing a reliability assessment consists of the following steps as shown in Figure 46.1.2. Steps on the left are compulsory, while steps on the right are optional and can be used to increase the detail of the calculation.

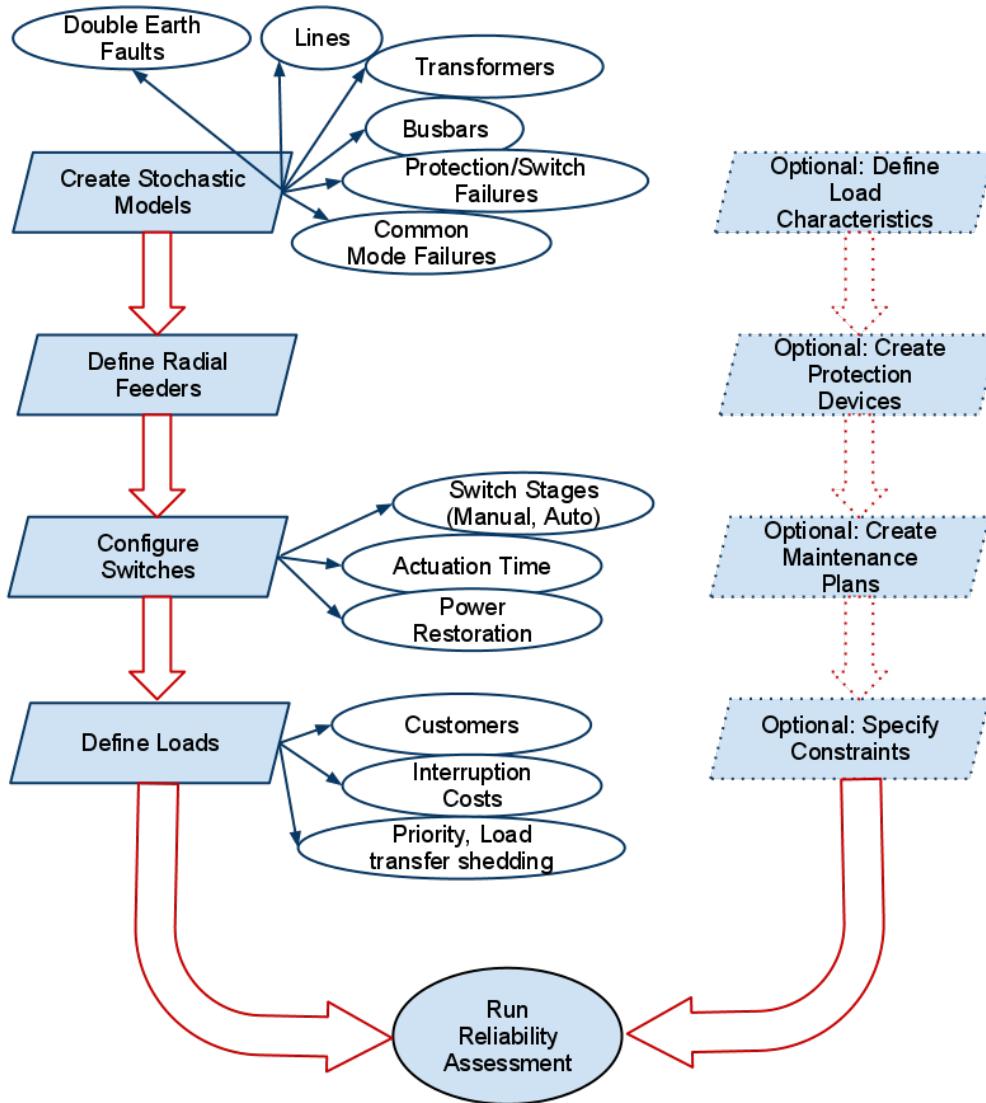


Figure 46.1.2: Reliability Assessment User Procedure

These procedures are explained in detail in the following sections.

## 46.2 Technical Background

The Reliability Assessment procedure considers the network topology, protection systems, constraints and stochastic failure and repair models to generate reliability indices. The technical background of the procedure and Stochastic Models is described in this section.

**Note:** A quantity is said to be stochastic when it has a random probability distribution. A simple example of a stochastic quantity is the expected repair duration for an item of equipment, which is based on the total number of repairs and repair duration. This measured data can be used to build Stochastic Models, and perform analysis using statistical calculation methods.

### 46.2.1 Reliability Assessment Procedure

The generation of reliability indices, using the Reliability Assessment tool also known as 'reliability analysis', consists of the following:

- Failure Modelling
- Load Modelling
- System State Creation
- Failure Effect Analysis (FEA)
- Statistical Analysis
- Reporting

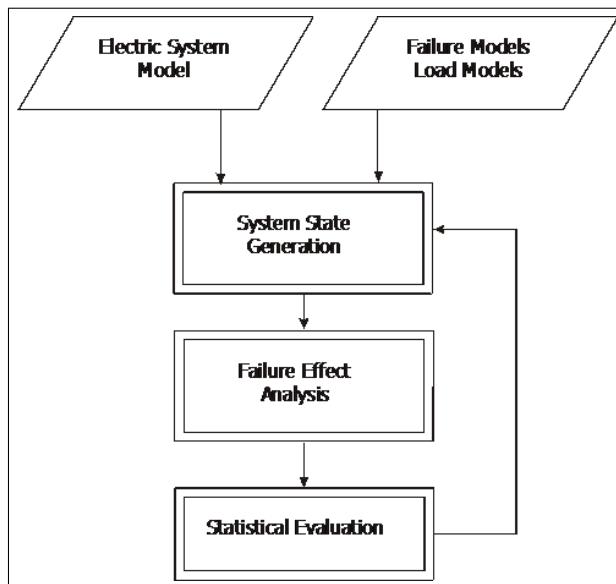


Figure 46.2.1: Reliability Analysis: Basic Flow Diagram

The reliability analysis calculation flow diagram is depicted in Figure 46.2.1. The failure models describe how system components can fail, how often they might fail and how long it takes to repair them when they fail. The load models can consist of a few possible load demands, or can be based on a user-defined load forecast and growth scenarios.

The combination of one or more simultaneous faults and a specific load condition is called a "system state". Internally, *PowerFactory*'s system state generation engine uses the failure models and load models to build a list of relevant system states. Subsequently, the Failure Effect Analysis (FEA) module analyses the faulted system states by simulating the system reactions to these faults.

The FEA takes the power system through a number of post-fault operational states that can include (some of them are only considered for the Distribution option in the reliability command, for a detailed description refer to Section 47.1):

- Fault clearance by tripping of protection breakers or fuses;
- Fault separation by opening separating switches;
- Power restoration by closing normally open switches, reconnecting busbars in substations or establishing a backward recovery;
- Overload and voltage constraint alleviation by load transfer, load shedding and busbar transfer.

The objective of the FEA function is to determine if system faults will lead to load interruptions and if so, which loads will be interrupted and for how long.

The results of the FEA are combined with the data that is provided by the system state generation module to create the reliability statistics including indices such as SAIFI, SAIDI and CAIFI. The system state data describes the expected frequency of occurrence of the system state and its expected duration. However, the duration of these system states should not be confused with the interruption duration. For example, a system state for a line outage, perhaps caused by a short-circuit on that line, will have a duration equal to the time needed to repair that line. However, if the line is one of two parallel lines then it is possible that no loads will be interrupted because the parallel line might be able to supply the full load current.

Even if the loads are interrupted by the outage, the power could be restored by network reconfiguration - by fault separation and closing a back-feed switch. The interruption duration will then equal the restoration time, and not the repair duration (equivalent to the system state duration).

### 46.2.2 Stochastic Models

A stochastic reliability model is a statistical representation of the failure rate and repair duration time for a power system component. For example, a line might suffer an outage due to a short-circuit. After the fault clearance, repair will begin and the line will be put into service again after a successful repair. If two states for line A are defined as 'in service' and 'under repair', monitoring of the line could result in a time sequence of outages and repairs as depicted in Figure 46.2.2.

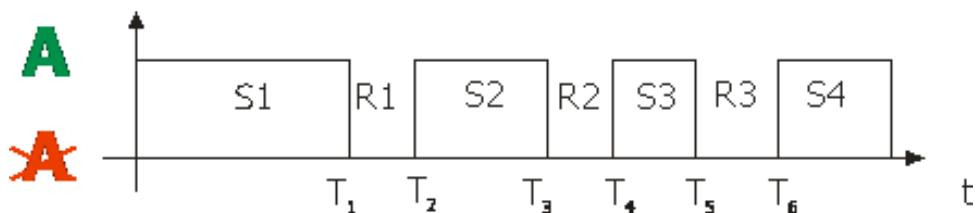


Figure 46.2.2: Line availability states are described by the status of the line (in service or under repair). Each of these states lasts for a certain time.

Line A in this example fails at time  $T_1$  after which it is repaired and put back into service at  $T_2$ . It fails again at  $T_3$ , is repaired again, etc. The repair durations are also called the 'Time To Repair' or 'TTR'. The service durations  $S_1 = T_1$ ,  $S_2 = T_3 - T_2$ , etc. are called the 'life-time', 'Time To Failure' or 'TTF'.

Both the TTR and the TTF are stochastic quantities. By gathering failure data about a large group of similar components in the power system, statistical information about the TTR and TTF, such as the mean value and the standard deviation, can be calculated. The statistical information is then used to define a Stochastic Model.

There are many ways in which to define a Stochastic Model. The so-called 'homogeneous Markov-model' is a highly simplified but generally used model. A homogeneous Markov model with two states is defined by a constant Failure Rate  $\lambda$  and a constant Repair Rate  $\mu$ . These two parameters can be used to calculate the following quantities:

- Mean Time to Failure,  $TTF = 1/\lambda$ ;
- Mean Time to Repair,  $TTR = 1/\mu$ ;
- Availability,  $P = TTF/(TTF+TTR)$ ;
- Unavailability,  $Q = TTR/(TTF+TTR)$ ;

The availability is the fraction of time when the component is in service; the unavailability is the fraction of time when it is in repair; and  $P+Q = 1.0$ .

### Example

If 7500 monitored transformers were to show 140 failures over 10 years, during which a total of 7360 hours was spent on repair, then:

$$\lambda = \frac{140}{10 \cdot 7500} = 0.00187 \frac{1}{a} \quad (46.1)$$

$$TTF = \frac{1}{\lambda} = 536 a \quad (46.2)$$

$$TTR = \frac{7360}{140} = 52.6 h = 0.006 a \quad (46.3)$$

$$\mu = \frac{1}{TTR} = 167 \frac{1}{a} \quad (46.4)$$

$$P = \frac{536}{536 + 0.006} = 0.999989 \quad (46.5)$$

$$Q = \frac{0.006}{536 + 0.006} = 0.000011 \approx 6 \frac{\text{min}}{a} \quad (46.6)$$

i.e. the expected outage duration is approximately 6 minutes per annum.

### 46.2.3 Calculated Results for Reliability Assessment

The network reliability assessment produces two types of indices:

- Load Point Indices
- System Indices

These indices are separated into frequency/expectancy indices and energy indices. Furthermore, there are indices to describe the interruption costs. In addition, a dedicated set of indices is available for the reliability evaluation of terminals and busbars.

Load point indices are calculated for each load, and are used in the calculation of many system indices. This section describes the simplified equations for the reliability indices. However, note that the *PowerFactory* reliability assessment calculations use more complex calculation methods. Nevertheless, the simplified equations shown here can be used for hand calculations or to gain insight into the reliability assessment results.

The reliability indices are driven by contingencies and their probability of occurrence, which lead to the loss of supply of parts or the entire modelled network. For the interruptions and their consideration in the index calculation, a differentiation is made between a sustained and a momentary loss of supply. The reason for the one or the other may come from the source of the fault being of sustained or transient nature or from the attempt of the network protection (reclosers) to restore the supply after detection of a fault. The following results refer to sustained interruptions where not explicitly stated otherwise.

#### 46.2.3.1 Parameter Definitions

In the definitions for the reliability indices, the following parameters are used:

$i$	Load point index
$k$	Contingency index
$t_{i,k}$	Interruption Time at load point $i$ for contingency case $k$
$frac_{i,k}$	The fraction of the load which is lost at load point $i$ , for contingency $k$
$C_i$	Number of customers supplied by load point $i$
$A_i$	Number of affected customers for an interruption at load point $i$
$Fr_k$	Frequency of occurrence of contingency $k$
$Pr_k$	Probability of occurrence of contingency $k$
$C$	Number of customers
$A$	Number of affected customers
$L_m$	Total connected apparent power interrupted, for each interruption event, $m$
$r_m$	Duration of each interruption event, $m$
$L_T$	Total connected apparent power supplied
$Pc_i$	Contracted active power at load point $i$
$IM_i$	Number of momentary interruptions at load point $i$

For unsupplied loads, or for loads that are shed completely,  $frac_{i,k} = 1$ . For loads that are partially shed,  $0 <= frac_{i,k} < 1$ .

#### 46.2.3.2 Load Point Frequency and Expectancy Indices

These indices are defined as follows:

**ACIF:** Average Customer Interruption Frequency, in  $[1/a]$ .

$$ACIF_i = \sum_k Fr_k \cdot frac_{i,k} \quad (46.7)$$

**ACIT:** Average Customer Interruption Time, in  $[h/a]$ .

$$ACIT_i = \sum_k 8760 \cdot Pr_k \cdot frac_{i,k} \quad (46.8)$$

**LPIF:** Load Point Interruption Frequency, in  $[1/a]$ .

$$LPIF_i = \sum_k Fr_k \quad (46.9)$$

**LPIT:** Load Point Interruption Time, in  $[h/a]$ .

$$LPIT_i = \sum_k 8760 \cdot Pr_k \quad (46.10)$$

**Note:** The parameters ACIF, ACIT, LPIF, and LPIT are only calculated and considered if the duration of the outage is longer than the time value “Calculation of SAIFI/SAIDI according to IEEE 1366”, that is set within the Advanced Options of the Reliability Assessment command.

**AID:** Average Interruption Duration, in [h].

$$AID_i = \frac{ACIT_i}{ACIF_i} \quad (46.11)$$

**TCIF:** Total Customer Interruption Frequency, in [C/a].

$$TCIF_i = ACIF_i \cdot C_i \quad (46.12)$$

**TCIT:** Total Customer Interruption Time, in [Ch/a].

$$TCIT_i = ACIT_i \cdot C_i \quad (46.13)$$

**TPCONTIF:** Total Contracted Power Interruption Frequency, in [MWh/a].

$$TPCONTIF_i = \sum_k Fr_k \cdot frac_{i,k} \cdot P_{ci} \quad (46.14)$$

**TPCONTIT:** Total Contracted Power Interruption Time, in [MWh/a].

$$TPCONTIT_i = \sum_k 8760 \cdot Pr_k \cdot frac_{i,k} \cdot P_{ci} \quad (46.15)$$

#### 46.2.3.3 System Indices

**SAIFI:** *System Average Interruption Frequency Index*, in [1/Ca]. It indicates how often the average customer experiences a sustained interruption during the period specified in the calculation.

$$SAIFI = \frac{\sum ACIF_i \cdot C_i}{\sum C_i} \quad (46.16)$$

**SAIFI\_P:** *Yearly Average Interruption Frequency (Contracted Power)*, in [1/a]. It indicates how often there are contracted power interruptions during the period of the calculation.

$$SAIFI\_P = \frac{\sum TPCONTIF_i}{\sum PCONTRACT_i} \quad (46.17)$$

**CAIFI:** *Customer Average Interruption Frequency Index*, in [1/Ca]. It is the mean frequency of sustained interruptions for those customers experiencing sustained interruptions. Each customer is counted once regardless of the number of times interrupted for this calculation.

$$CAIFI = \frac{\sum ACIF_i \cdot C_i}{\sum A_i} \quad (46.18)$$

**SAIDI:** *System Average Interruption Duration Index*, in [h/Ca]. It indicates the total duration of interruption for the average customer during the period in the calculation. It is commonly measured in customer minutes or customer hours of interruption.

$$SAIDI = \frac{\sum ACIT_i \cdot C_i}{\sum C_i} \quad (46.19)$$

**SAIDI\_P:** *Yearly Average Interruption Duration (Contracted Power)*, in [h/a]. It indicates the total duration of contracted power interruptions during the period of the calculation.

$$SAIDI\_P = \frac{\sum TPCONTIT_i}{\sum PCONTRACT_i} \quad (46.20)$$

**CAIDI:** *Customer Average Interruption Duration Index*, in [h]. It is the mean time to restore service.

$$CAIDI = \frac{SAIDI}{ASIFI} \quad (46.21)$$

**ASAI:** *Average Service Availability Index*. It represents the fraction of time that a customer is connected during the defined calculation period.

$$ASAI = 1 - ASUI \quad (46.22)$$

**ASUI:** *Average Service Unavailability Index*. It is the probability of having all loads supplied.

$$ASUI = \frac{\sum ACIT_i \cdot C_i}{8760 \cdot \sum C_i} \quad (46.23)$$

**ASIFI:** *Average System Interruption Frequency Index*, in [1/a]. The calculation of this index is based on load rather than customers affected. ASIFI can be used to measure distribution performance in areas that supply relatively few customers having relatively large concentrations of load, predominantly industrial/commercial customers

$$ASIFI = \frac{\sum L_m}{L_T} \quad (46.24)$$

**ASIDI:** *Average System Interruption Duration Index*, in [h/a]. It is the equivalent of SAIDI but based on load, rather than customers affected.

$$ASIDI = \frac{\sum (r_m \cdot L_m)}{L_T} \quad (46.25)$$

**MAIFI:** *Momentary Average Interruption Frequency Index*, in [1/Ca]. It evaluates the average frequency of momentary interruptions. The calculation is described in the IEEE Standard 1366 'IEEE Guide for Electric Power Distribution Reliability Indices'.

$$MAIFI = \frac{\sum IM_i \cdot A_i}{\sum C_i} \quad (46.26)$$

#### 46.2.3.4 Load Point Energy Indices

**LPENS:** Load Point Energy Not Supplied, in [MWh/a].

$$LPENS_i = ACIT_i \cdot (\widehat{Pd}_i + \widehat{Ps}_i) \quad (46.27)$$

**LPES:** Load Point Energy Shed, in [MWh/a].

$$LPES_i = ACIT_i \cdot \widehat{Ps}_i \quad (46.28)$$

Where  $Pd_i$  is the weighted average amount of power disconnected at load point  $i$  and  $Ps_i$  is the weighted average amount of power shed at load point  $i$ .

#### 46.2.3.5 System Energy Indices

**ENS:** *Energy Not Supplied*, in [MWh/a]. It is the total amount of energy on average not delivered to the system loads.

$$ENS = \sum LPENS_i \quad (46.29)$$

**SES:** *System Energy Shed*, in [MWh/a]. It is the total amount of energy on average expected to be shed in the system.

$$SES = \sum LPES_i \quad (46.30)$$

**AENS:** *Average Energy Not Supplied*, in [MWh/Ca]. It is the average amount of energy not supplied, for all customers.

$$AENS = \frac{ENS}{\sum C_i} \quad (46.31)$$

**ACCI:** *Average Customer Curtailment Index*, in [MWh/Ca]. It is the average amount of energy not supplied, for all affected customers.

$$ACCI = \frac{ENS}{\sum A_i} \quad (46.32)$$

#### 46.2.3.6 Load Point Interruption Cost

**LPIC:** *Load Point Interruption Cost*, in [USD/a].

$$LPIC_i = \sum_k LPIC_{i,k} \quad (46.33)$$

where  $LPIC_{i,k}$  is the average interruption cost for load point  $i$  and contingency case  $k$ , considering the load point interruption costs function and the assessed distribution of the durations of the interruptions at this load point for contingency case  $k$ .

Interruption costs are calculated differently depending on whether an Energy Tariff ( $IntTariffenergy$ ) or a Time Tariff ( $IntTarifftime$ ) is used.

On the one hand, if an Energy Tariff is selected:

$$LPIC_{i,k} = Fr_k \cdot E_{i,k} \cdot Tariff_{energy}(t_{i,k}) \quad (46.34)$$

where  $E_{i,k}$  is the energy not supplied at load point  $i$  due to the contingency  $k$ .

On the other hand, if a Time Tariff is selected:

$$LPIC_{i,k} = Fr_k \cdot P_i \cdot frac_{i,k} \cdot Tariff_{time}(t_{i,k}) \quad (46.35)$$

where  $P_i$  is the active power at load point  $i$  before the contingency  $k$  occurs.

For cost functions expressed in money per interrupted customer, the number of interrupted customers is estimated for each interruption as the highest number of customers interrupted at any time during the whole interruption duration.

#### 46.2.3.7 System Interruption Costs

**EIC:** *Expected Interruption Cost*, in [ $MUSD/a$ ]. It is the total expected interruption cost.

$$EIC = \sum LPEIC_i \quad (46.36)$$

**IEAR:** *Interrupted Energy Assessment Rate*, in [ $USD/kWh$ ]. It is the total expected interruption cost per not supplied kWh.

$$IEAR = \frac{EIC}{ENS} \quad (46.37)$$

#### 46.2.3.8 Indices for Busbars/Terminals

**AID:** Average Interruption Duration, in [h].

$$AIT = \sum_k 8760 \cdot Pr_k \quad (46.38)$$

**AIF:** Yearly Interruption Frequency, in [1/a].

$$AIF = \sum_k Fr_k \quad (46.39)$$

**AIT:** Yearly Interruption Time, in [h/a].

$$AID = \frac{AIT}{AIF} \quad (46.40)$$

Where the sum iterates through all contingencies leading to an unsupplied busbar (analogous to LPIF and LPIT).

#### 46.2.4 System State Enumeration in Reliability Assessment

In *PowerFactory*, Reliability Assessment uses a System State Enumeration to analyse all possible system states, one by one. A fast 'topological' method is used which ensures that each possible system state is only analysed once. State frequencies (average occurrences per year) are calculated by considering only the transitions from a healthy situation to an unhealthy one and back again. This is important because the individual system states are analysed one by one, and the (chronological) connection between them is therefore lost.

The enumerated calculation method is fast for quick investigation of large distribution networks, but does not compromise accuracy. Exact analytic averages are calculated. Distributions of reliability indices, however, cannot be calculated. For example, the average annual unavailability in hours/year can be calculated, but the probability that this unavailability is less than 15 minutes for a certain year cannot be calculated.

The state enumeration algorithm can include independent failures, simultaneous ( $n-2$ ) failures, common mode failures, numerous load states and planned outages.

An overview flow diagram for the reliability assessment by state enumeration is shown in Figure 46.2.3.

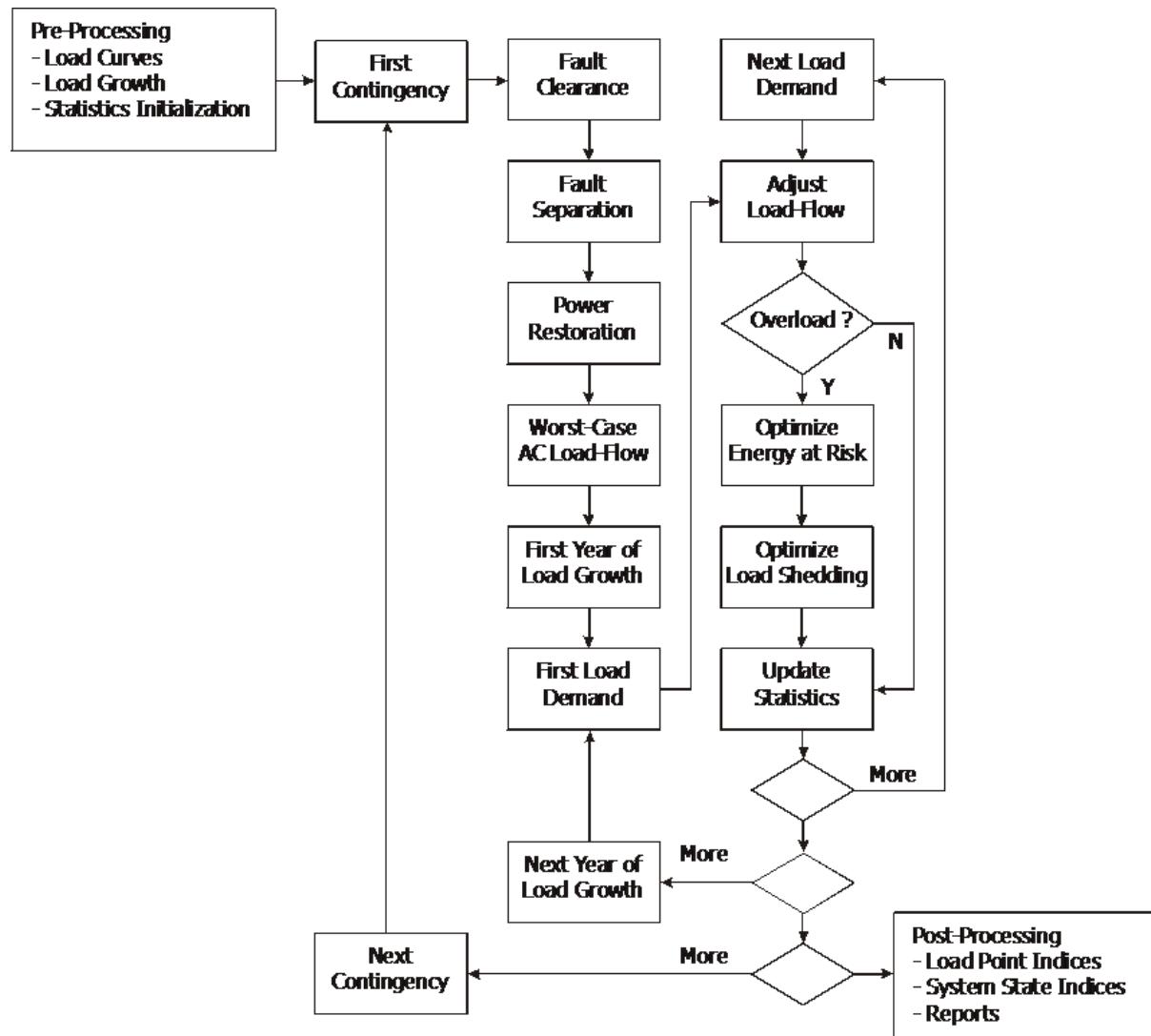


Figure 46.2.3: Overview Flow Diagram for Reliability Assessment by State Enumeration

After the State Enumeration is complete, each simulated system state can be viewed using the 'tracing

tool' on the Reliability Toolbar, see Section [47.2](#) for more information.

## 46.3 Setting up the Network Model for Reliability Assessment

Prior to starting a Reliability Assessment Calculation, you must setup the Network Model with specific reliability data models. This section discusses the following procedures:

- Defining Stochastic Failure and Repair Models, described in Section [46.3.1](#)
- Defining Feeders for Reliability Assessment, described in Section [46.3.2](#)
- Configuring Switches for Reliability Assessment, described in Section [46.3.3](#)
- Load Modelling for Reliability Assessment, described in Section [46.3.4](#)
- Considering Multiple System Demand Levels, described in Section [46.3.6](#)
- Defining Fault Clearance Based on Protection Device Location, described in Section [46.3.7](#)
- Considering Planned Maintenance, described in Section [46.3.8](#)
- Specifying Individual Component Constraints, described in Section [46.3.9](#)
- Considering Switching Rules, described in Section [46.3.10](#)

### 46.3.1 Defining Stochastic Failure and Repair Models

*PowerFactory* allows the user to define models to capture the stochastic failure and repair behaviour of system components. Stochastic failure models () define the probability that a component will fail and when it does fail, the mean time to repair the component. The following stochastic failure models are supported by *PowerFactory*:

- Busbar/Terminal Stochastic Model (*StoTypbar*), described in Section [46.3.1.1](#)
- Line/Cable/Branch Stochastic Model (*StoTypne*), described in Section [46.3.1.2](#)
- Transformer and Shunt Stochastic Model (*StoTyptrf*), described in Section [46.3.1.3](#)
- Distribution Transformer Stochastic Model for MV Loads (*StoTyptrf*), described in Section [46.3.1.4](#)
- Generator Stochastic Model (*StoGen*), described in Section [46.3.1.5](#)
- Common Mode Stochastic Model (*StoCommon*), described in Section [46.3.1.6](#)
- Protection/Switch Failure Model, described in Section [46.3.1.7](#)
- Double Earth Fault Failure Model, described in Section [46.3.1.8](#)

This section describes each of these Stochastic Models and the procedure for defining them.

#### 46.3.1.1 Bar Type Failures (*StoTypbar*)

The Bar Type Failures model (*StoTypbar*) is a stochastic failure and repair model for nodes of a network. It can be assigned to a Terminal (*ElmTerm*) or to its Busbar Type (*TypBar*).

---

**Note:** Bar Type Failures can only be mapped to Terminals that are defined as Busbar or Junction Node.

The following parameters can be defined on the Basic Data page:

- **Failure frequency for terminal in 1/a.**
  - **Additional failure frequency per connection in 1/a.**
  - **Repair duration in h.**
- 

**Note:** For a description of the parameters on the Earth Fault page, refer to Section [46.3.1.8](#).

---

The *Additional failure frequency per connection* allows the user to consider an incremental failure rate due the failure of the circuit breakers connected to the terminal, without modelling this mode explicitly in the associated circuit breakers. The forced outage rate of a given terminal then results from considering its own failure frequency as well as the additional failure frequency per connection.

$$FOR1 = DnFr1 + NC_{CB} \cdot DnFr2 \quad (46.41)$$

where:

- *FOR1* is the forced outage rate of the terminal in 1/a.
- *DnFr1* is the failure frequency for terminal in 1/a.
- *DnFr2* is the additional failure frequency per connection in 1/a.
- *NC<sub>CB</sub>* is the number of connected circuit breakers.

It should be noted that the additional failure frequency is added to the failure frequency of the Terminal only if the connection is made via a circuit breaker or a disconnecting circuit breaker. The following cases can be considered:

- Simple node (Terminal): The switching devices are modelled using Switch (*StaCubic*) objects. Additional failure frequency is only added if the switch type is cbk (circuit breaker) or dcb (disconnecting circuit breaker).
  - Composite node (Substation or Secondary Substation). The switching devices are modelled using Breaker/Switch (*ElmCoup*) objects. The additional frequency is only added to the failure frequency for terminal if the switch type is cbk (circuit breaker) or dcb (disconnecting circuit breaker).
- 

**Note:** Concerning the failure data, it should be noted that is possible to define the stochastic failure model using either failure frequencies or outage expectancies. The *Options* button (⚙) can be used to select the most convenient input mode.

---

As mentioned above, the bar type failure model can be added to a Terminal element (*ElmTerm*) or to its type (*TypBar*). If you want to use the same stochastic model for several terminals of the same type, then you should define it via the type. On the other hand, if you want to use a stochastic model for a single element, then you should define it via the element.

---

**Note:** If you define a stochastic element model for a terminal that also has a stochastic model within its corresponding type, the element model overrules the type model.

---

### 46.3.1.2 Line/Cable/Branch Stochastic Model (*StoTypLine*)

It is possible to define a Stochastic Model for every line, cable or branch within the network. The Stochastic Model can be defined either through the object type or through the object element. If you want to use the same Stochastic Model for a number of different lines/cables then you should define it through the object type reliability page. Alternatively, if you want to use a Stochastic Model for only one element, then you should define it through the element reliability page.

#### Cable type definition

To define a Stochastic Model for a line or cable type follow these steps:

1. Open the dialog for the line *type* and select the Reliability tab.
2. Click on the arrow button *Select Type* (  ) next to *Stochastic model* and select the option *New Project Type*.... The dialog for the *Line Type Failures* will open up.
3. Enter the *Sustained Failure Frequency*. Note that the probability of the line failure is determined using this value and the length of the line. For example, a 12 km line with a Sustained failure frequency of  $0.032(1/(a \cdot km))$  will have a failure probability of  $12 \cdot 0.032 = 0.384(1/(a))$ .
4. Enter the mean repair duration in hours.
5. Enter the Transient Fault Frequency. Note this parameter is used for the calculation of the MAIFI.
6. Click on **OK** twice to return to the element dialog.

#### Cable element definition

To define a Stochastic Model for a line, cable or branch element follow these steps:

1. Open the dialog for the line or branch *element* and navigate to the Reliability tab.
2. Click on the arrow button *Select Type* (  ) next to *Element model* and select the option *New Project Type*.... The dialog for the *Line Type Failures* will open up.
3. In case of branches, if the stochastic data for one of its components is defined, then you will only see a summary of the stochastic data. If this is not the case, then an 'element model' can be selected for branches. This stochastic model is applied to all the branch components, i.e lines, cable systems and tower models.
4. If reliability data are available for branches, then branches are treated like other *elements*
5. Enter the failure data and repair time data as described above for the line type.
6. Click on **OK** to return to the element dialog.

---

**Note:** For a description of the parameters on the Earth Fault page, refer to Section [46.3.1.8](#).

---

### 46.3.1.3 Transformer and Shunt Stochastic Model (*StoTyptrf*)

It is possible to define a Stochastic Model for every transformer and Shunt within the network. The Stochastic Model can be defined either through the object type or through the object element. If you want to use the same Stochastic Model for a number of different transformers or Shunts then you should define it through the object type reliability page. Alternatively, if you want to use a Stochastic Model for only one transformer or Shunt element, then you should define it through the element reliability page.

#### Transformer type definition

To define a Stochastic Model for a transformer type follow these steps:

1. Open the dialog for the transformer *type* and select the Reliability tab.
2. Click on the arrow button *Select Type* (▼) next to *Stochastic model* and select the option *New Project Type*.... The dialog for the *Transformer Type Failures* will open up.
3. Enter the failure frequency data (1/a).
4. Enter the mean repair duration in hours.
5. Click on **OK** twice to return to the element dialog.

#### Transformer and Shunt element definition

To define a Stochastic Model for a transformer or shunt *element* follow these steps:

1. Open the dialog for the transformer or shunt *element* and select the Reliability tab.
2. Click on the arrow button *Select Type* (▼) next to *Element model* and select the option *New Project Type*.... The dialog for the *Transformer Type Failures* will open up.
3. Enter the failure data and repair time data as described above for the transformer type.
4. Click on **OK** to return to the element dialog.

---

**Note:** For a description of the parameters on the Earth Fault page, refer to Section [46.3.1.8](#).

---

#### 46.3.1.4 Distribution Transformer Stochastic Model for MV Loads (*StoTyptrf*)

In *PowerFactory*, MV Loads can provide the functionality of a built-in distribution transformer. The fault behaviour of the distribution transformer is the same as for other transformers, except for the fact that the load connected behind the transformer is not supplied until the end of the repair duration.

To define a Stochastic Model for a distribution transformer within the MV Load element or type, open the dialog for the MV Load element (*ElmLodmv*) or its type (*TypDistrf*) and select the Reliability tab. As the failure model is based on the transformer (*StoTyptrf*), the steps for the definition of this failure model are equivalent to the ones described in Section [46.3.1.3](#).

#### 46.3.1.5 Generator Stochastic Model (*StoGen*)

Within a network, it is possible to define a Stochastic Model for Generation (*StoGen*) for every generator class (synchronous machines, static generators, PV systems, etc.) which can be used by both Reliability Analysis and Generation Adequacy Analysis calculation functions. For further information refer to Section [49.3.1](#). The Stochastic Model can only be defined via the network element. The failure model can contain any number of load level states; each state representing the availability of the generator over a year. This way, complete and/or partial outages can be modelled.

Upon execution of Reliability Assessment, *PowerFactory* creates a separate contingency for each defined state. A load flow is calculated considering the reduced availability (including 0%) of the generator, and depending on constraint violations, load shedding and/or re-dispatch of alternative generators may result.

The *Stochastic Model for Generation* includes an unlimited number of states with each defined according to:

- **State:** Name of the state
- **Availability in %:** Percentage of the rated power available

- **Probability in %:** Probability that this state is valid (the sum of all probabilities must be 100 %)
- **Duration in h:** Time needed to solve the given failure
- **Frequency in 1/a:** Number of incidents that cause the given state per year
- **Total Duration in h/a:** Total duration of the given state per year

### Generator element definition

To define a Stochastic Model for a generator *element* follow these steps:

1. Open the dialog for the Generator *element* and select the Reliability tab.
2. Click on the arrow button *Select Type* (  ) next to *Stochastic Model* and select the option *Select...*. The dialog for the *Equipment Type Library* of the project will appear.
3. Click the *New Object* button (  ) to create a Stochastic Model for Generation object (*StoGen*). The dialog for the object should appear.
4. Enter the data according to one of the following:
  - Probability and repair duration
  - Repair duration and frequency per year
  - Probability and frequency per year
5. Click on **OK** to return to the element dialog.

#### 46.3.1.6 Common Mode Stochastic Model (*StoCommon*)

A common mode failure (*StoCommon*) involves the simultaneous failure of two or more power system components. An example is a distribution feeder where two lines with different voltages share the same poles. If one or more poles fail, for example a car hits a pole, then both lines will be interrupted simultaneously: these lines have a 'common failure mode'. Such a failure will usually be more likely than the probability of the two lines failing independently at the same time.

In *PowerFactory*, it is possible to define a common mode failure object to consider such failures in the reliability calculation. These Stochastic Models consider the common mode failure probability in addition to the independent failure mode of each component within the model.

To define a common mode failure Stochastic Model through the single line diagram follow these steps:

1. Mark two or more network objects.
2. Right-click on one of the marked elements and chose *Operational Library* → *Common Mode Failure* → *New...*.
3. To add a network element, add a cell below the last full cell by right-clicking within an empty area of the dialog and selecting the option *Append Row(s)*.
4. Double-click in the first empty cell of the 'Name' column, to open an object selection browser.
5. Use the browser to find the object that is part of the Common Mode Failure that you are trying to define.
6. Click **OK** to return to the Common Mode Failure dialog.
7. Repeat steps 3-6 to add more objects to the Common Mode Failure.
8. Click the 'Failure Data' tab and enter the Sustained Failure Frequency, Mean Outage duration and Transient Fault Frequency data.
9. Click **OK** to save the changes.

**Note:** Meaningful names are created for common modes based on the names of elements in the common modes.

---

To define a common mode failure Stochastic Model through the Data Manager (not suitable for the first Common Mode Stochastic Model) follow these steps:

1. Using the Data Manager, select the 'Common Mode' failures folder within the 'Operational Library'.
2. Click the *New Object* button ( ) to create a Stochastic Common Mode failure object (*StoCommon*). The dialog for the object should appear.
3. Double click in the first empty cell of the 'Name' column, to open an object selection browser.
4. Use the browser to find the object that is part of the Common Mode Failure that you are trying to define.
5. Click **OK** to return to the Common Mode Failure dialog.
6. Add a cell below the last full cell by right-clicking within an empty area of the dialog and selecting the option 'Append Rows'.
7. Repeat steps 3-6 to add more objects to the Common Mode Failure.
8. Click the 'Failure Data' tab and enter the Sustained Failure Frequency, Mean Outage duration and Transient Fault Frequency data.
9. Click **OK** to save the changes.
10. Common mode failures can be defined between all elements with stochastic data, e.g.between branches and other branches or between branches and other components.
11. System failure states created by *PowerFactory* could consist of common mode failure, maintenance and protection failures.

#### 46.3.1.7 Protection/Switch Failures

*PowerFactory* can consider the failure of the protection system to clear the fault as a stochastic probability within the reliability calculation. This is enabled by entering a 'Probability of Failure' into the switch object. To enter this data:

1. Open the dialog for the switch object where you want to enter the switch failure probability. Normally switches are accessed by right-clicking their containing cubicle and selecting the option *Devices* → *Show All Devices*.
2. On the Reliability page of the switch object, enter the 'Fault Clearance: circuit breaker fails to open probability' in percent. For example, a 5% failure rate means that on average 1 out of 20 attempted fault clearance operations will fail.
3. "Unnecessary backup protection maloperation" gives the probability of the backup protection operating unnecessarily. That is, the backup protection tripping in addition to the main protection device.
4. "Frequency of spurious protection operation" gives the probability of a relay tripping spuriously, without any indication.

---

**Note:** *PowerFactory* does not distinguish between a protection system failure and a switch failure. For example, the reason that a switch fails to open could be caused by a faulty relay, a protection mal-grading or a faulty circuit breaker. The cumulative probability of all these events should be entered into the switch failure probability.

---

#### 46.3.1.8 Double Earth Faults

In ungrounded or resonant grounded systems, a single earth fault does not lead to tripping of overcurrent protective devices and thus to an immediate disconnection of the faulted part of the system. Therefore, it is generally said that the system can continue to operate for a short time despite the single earth fault. However, if, due to the overvoltage on the healthy phases that appears during a single earth fault, a second earth fault occurs on the same feeder or on another feeder connected to the same earth system, high fault currents will flow. These fault currents are extinguished by disconnection of the faulted elements and thus lead to an interruption of service.

In *PowerFactory*, it is possible to define models for double earth faults in order to consider them for reliability assessment. This can be done on the Earth Fault page of the *StoTypbar*, *StoTypne* and *StoTyprf* stochastic failure models () after enabling the *Model Earth Faults* option. The following parameters can be defined:

- **Frequency of single earth faults.**

Refers to the frequency of single earth faults of the system component to which the stochastic failure model is assigned.

- **Conditional probability of a second earth fault.**

Refers to the probability that a single earth fault will occur in the element to which the stochastic failure model is assigned, given that a single earth fault has occurred in another system component that is connected to the same grounding area.

- **Repair duration.**

Refers to the mean time required to repair the system component to which the stochastic failure model is assigned.

---

**Note:** For contingencies considering a double earth fault, *PowerFactory* performs a conservative analysis, i.e. uses the longest repair duration to assess the effects of the fault.

---

If the *Double earth faults* option is enabled in the *Reliability Assessment (ComRel3)* calculation command, *PowerFactory* automatically generates a Contingency (*ComContingency*) for each pair of network elements meeting the following conditions:

- The *Frequency of single earth faults* of a first element is  $> 0$ , and
- The *Conditional probability of a second earth fault* of a second element is  $> 0$ , and
- The network elements are connected to the same grounding area, and
- The star point of the elements supplying that part of the network is either isolated (not connected) or compensated (*Peterson Coil* option enabled).

---

**Note:** The failure frequency of a double earth fault contingency is evaluated using combinatorial properties of event probabilities. That is, the failure frequency is calculated as:

$$FF_{DEFC} = CCEarFr_A \cdot CCEarProb_B + CCEarFr_B \cdot CCEarProb_A \quad (46.42)$$

where:

- $FF_{DEFC}$  is the failure frequency of a double earth fault contingency in 1/a.
- $CCEarFr$  is the frequency of single earth faults of the element in 1/a.
- $CCEarProb$  is the conditional probability of a second earth fault of the element in %.

### 46.3.2 Defining Feeders for Reliability Assessment

When performing a reliability calculation with the *Distribution* option set under 'Basic Options', *PowerFactory* requires that feeders have been defined in the model.

To create a feeder:

- Right-click on the cubicle at the head of the feeder and select the option *Network Groupings* → *Feeder* → *New...*; or
- For fast creation of multiple feeders, right-click the bus that the feeder/s are to be connected to and select the option *Network Groupings* → *Feeder* → *New...*. More information on feeders and feeder creation can be found in Chapter 15: Grouping Objects, Section 15.6(Feeders).

### 46.3.3 Configuring Switches for Reliability Assessment

A critical component of the Failure Effect Analysis (FEA) is the behaviour of the switches and fuses in the network model. Switches in *PowerFactory* are classified into five different categories:

- Circuit Breakers, which are automatic, usually relay-controlled, and may be equipped for remote operation. They are used to clear faults and close back-feeds for power restoration;
- Disconnectors, used for isolation and power restoration;
- Load-Break-Switch, used for isolation and power restoration;
- Switch Disconnector, used for isolation and power restoration;
- Disconnecting circuit breaker, which is a circuit breaker with visual disconnection path.

All switches in *PowerFactory* are modelled using the *StaSwitch* or *ElmCoup* objects. The switch category (CB, disconnector, etc.) is selected on the basic data page of the switch. This selection has an impact on the options available on the Reliability page.

The actions that the FEA analysis takes depends on the configuration of these switches and, optionally, the location of protection devices.

#### Configuration steps

To configure a switch for reliability analysis follow these steps:

1. Open the dialog for the switch and select the reliability page. This can be done directly by editing switches modelled explicitly on the single line diagram, or for switches embedded within a cubicle, by right-clicking the cubicle and selecting the option *Devices* → *Show All Devices*, to access the switch.
2. Select the 'Sectionalising' option. The following choices are available:
  - Remote controlled (Stage 1); This option means that the actuation time of this switch is taken from the global 'remote controlled' switch actuation time. The default time is 1 min but this can be adjusted within the reliability command, see Section 46.4.1. Typically remote controlled switches are circuit breakers controlled by relays or with communications from a control room.
  - Indicator of Short Circuit (Stage 2); This option represents a switch that has an external indication of status on the outside of the switch enclosure. This allows the operator/technician to easily identify the switch status and actuate the switch.
  - Manual (Stage 3); These switches need direct visual inspection to determine their status and therefore take longer to actuate than either stage 1 or stage 2 switches.
3. Select the 'Power Restoration' option. The following choices are available:

- Do not use for power restoration; If this option is selected the switch can only be used for isolation of equipment or load shedding. It will not be used by the FEA calculation to restore power.
  - Direction dependency of the restoration:
    - For switches (*StaSwitch*) and breakers (*ElmCoup*) in bays of substations feeding branches, the following options are available for a direction dependency:
      - \* From Branch to Node: If this option is selected, the switch will only be used for the restoration if the branch side of the breaker is supplied. The switch will not be used for power restoration in the opposite direction.
      - \* From Node to Branch: If this option is selected, the switch will only be used to restore power if the node side of the breaker is supplied. The switch will not be used for power restoration in the opposite direction.
    - For breakers (*ElmCoup*) connecting two terminals:
      - \* From j to i: If the connection side j of the breaker is supplied, the breaker may be used by the algorithm to restore the unsupplied side i of the breaker.
      - \* From i to j: If the connection side i of the breaker is supplied, the breaker may be used by the algorithm to restore the unsupplied side j of the breaker.
  - Independent of direction; If this option is selected the switch will be used to restore power flow regardless of the direction of the post restoration active power flow.
4. Enter the time to actuate switch (Stage 2 and 3 switches only); This field specifies the time taken by the operator to actuate the switch. Note, this excludes the local access and access time if the switch is within a substation. The total actuation time of such a switch is therefore the switch actuation time + the substation access time + the local access time.
- Fuses in *PowerFactory* are classified into four different categories:
- Fuse: A fuse, an electrical safety device, consists of or includes a wire or strip of fusible metal which melts and interrupts the circuit when the current exceeds a specified threshold. Fuses located in cubicles are purely protection devices acting on the corresponding switching device. The corresponding switch will clear faults for the option *Use switches with protection devices*.
  - Fuse Disconnector: A fused disconnector switch is a combination of a switch to disconnect the circuit and a fuse to shut the circuit off in the event of a problem. When installing or maintaining equipment on the circuit, or the circuit itself, the switch provides a method to manually shut off the power. A fuse switch disconnector is a distribution device widely used in industry that can be used to manually switch and interrupt circuits and disconnect the power supply. The function to re-supply power is not yet supported by *PowerFactory* so it is only used to clear faults.
  - Fuse Switch - Disconnector: Fuse load break switches are devices that provide protection against overcurrent. They reinforce and enhance the safety of your electrical installations by ensuring the on-load making and breaking of electrical circuits. A load break switch is a device that disconnects the electrical current from an electric device or circuit when the current exceeds the nominal value. Fuse load break switches have high current-limiting power fuses. These fuses can be replaced safely. They also have an indicator of a blown fuse.
  - Fuse Load-Switch: This is a Fuse Switch Disconnector for switching of loads.

Fuses basically support two functions in the reliability assessment:

1. They melt in order to clear faults.
2. They can be replaced during the power re-supply step if the option “Replace melted fuse during power restoration” is activated.

The complete switching times depend on the following settings:

- Switching procedure for fault separation / power restoration (page *Restoration* of reliability command)
- Remote-controlled switches: the time for the remote-controlled switches is taken from the reliability command

- Access time of switches:
  - “First Manual switch action after the fault”: Access time of terminals.
  - “After a switch action at another busbar”: Local access time of terminals.
- Time to actuate switch:
  - Remote-controlled switches: with time for remote-controlled switches from the reliability command.
  - Switches for the “Short-circuit indicator” and “Manual” phases:
    - \* For the first switch: Time to actuate the switch (as specified in ElmCoup/StaSwitch) + the time for “First Manual switch action after the fault”
    - \* For all other switches: Time to actuate the switch + the time “After a switch action at another busbar”

The final time to actuate a switch is calculated as follows:

- The protection breakers/switches actuate immediately (at 0:00 minutes after the fault was applied).
- Switching procedure for fault separation / power restoration:
  - Concurrently:
    - \* Remote controlled switches: Are actuated at the time entered in the reliability command.
    - \* All switches are operated at the same time at each stage, since each one of them actuated at the same switch time “Time to actuate switch” + “Access times of Switches” of the substation. The final switching time is the maximum of the switching times of all the switches.
  - Sequential (the previous switching time is taken into account):
    - \* Remote controlled switches: Are actuated at the time entered in the reliability command.
    - \* Any other switch: The switching times are added up one after the other, i.e. the first manual switch receives  $t_1 = \text{Time to actuate switch 1} + \text{Access Time of switch 1}$ (= “First manual switch action after the fault”), the second receives  $t_2 = t_1 + \text{Time to actuate switch 2} + \text{Access Time of switch 2}$ (= “After a switch action at another busbar”), etc.

A switch however, will never be closed for power restoration before the corresponding area was separated from the fault. If an area can be separated from the fault after 15 minutes and the switch for restoration is remote controlled (time of remote controlled switches is set to 3:00 minutes), it will be restored after 15 minutes.

---

**Note:** The Sectionalising options are only considered when the ‘Distribution’ reliability analysis option is selected under ‘Basic Options’. If the ‘Transmission’ mode is selected, then all switches are assumed to be remote controlled.

---

#### 46.3.4 Load Modelling for Reliability Assessment

This section provides a general description of the load element parameters that are used by the reliability calculation. The first sub-section describes how to input the number of customers that each load represents and how to classify each load. The second sub-section describes how to define load shedding and transfer parameters.

##### 46.3.4.1 Specifying the Number of Customers for a Load

Many of the reliability indices such as SAIFI and CAIFI are evaluated based on the number of customers interrupted. Therefore, for accurate calculation of these indices it is important to specify the number of customers that each load represents.

For the general load (*ElmLod*) and the MV load (*ElmLodmv*), the Number of connected customers can be entered on the Reliability page.

For the low voltage load (*ElmLodlv*) and the partial LV load (*ElmLodlvp*), the Number of customers can be entered directly on the Basic Data page.

### Load Classification

Every load can be optionally classified into agricultural, commercial, domestic or industrial load. This option does not affect the calculation of the reliability indices and is provided for categorisation purposes only.

#### 46.3.4.2 Specifying Load Shedding and Transfer Parameters

Load transfer and load shedding are used to alleviate violated voltage or thermal constraints caused by the power restoration process. There is a distinction between load transfer for constraint alleviation, such as described in this section, and load transfer for power restoration. Load transfer by isolating a fault and closing a back-stop switch is considered automatically during the fault separation and power restoration phase of the failure effect analysis.

If a violated constraint is detected in the post-fault system condition, a search begins for the loads contributing to these overloads. The overloads are then alleviated by either:

- Transferring some of these loads, if possible; or
- Shedding some of these loads, starting with the lowest priority loads.

To define the load shedding parameters follow these steps:

1. Open the reliability page of the load element.
2. Enter the number of load shedding steps using the 'Shedding steps' list box. For example, four shedding steps means that the load can be shed to 25%, 50%, 75% or 100% of its base value. Infinite shedding steps means that the load can be shed to the exact value required to alleviate the constraint.
3. Enter the 'Load priority'. The reliability algorithm will always try to shed the loads with the lowest priority first. However, high priority loads can still be shed if the algorithm determines this is the only way to alleviate a constraint.
4. Enter the load transfer percentage in the 'Transferable' parameter. This defines the percentage of this load that can be transferred away from the current network. *PowerFactory* assumes that the transferred load is picked up by another network that is not modelled, hence load transferring in this way is equivalent to load shedding in terms of constraint alleviation. The difference is that transferred load is still considered as supplied load, whereas shed load is obviously not supplied.
5. Optional: Use the selection control next to 'Alternative Supply (Load)' to specify an alternative load that picks up all transferred load.

---

**Note:** There is a critical difference between the transmission reliability and distribution reliability functions. In distribution reliability all constraint alleviation is completed using switch actions, so loads can only be fully shed (switched out) or they remain in service. However, by contrast, the transmission reliability option can shed or transfer a percentage of the load.

---

#### 46.3.5 Modelling Load Interruption Costs

When supply to a load is interrupted, there is a cost associated with the loss of supply. *PowerFactory* supports the definition of cost curves for load elements using Energy Tariffs and Time Tariffs. They can

be defined using the 'Tariff' characteristic on the reliability page of the load element, as discussed in Chapter 18: Parameter Characteristics, Load States, and Tariffs, Section 18.5 (Tariffs).

Projects imported from previous versions of *PowerFactory* may include Vector Characteristics for the definition of cost curves, which are discussed in Chapter 18: Parameter Characteristics, Load States, and Tariffs, Section 18.2.10 (Vector Characteristics with Time Scales).

### 46.3.6 System Demand and Load States

#### Considering Multiple System Demand Levels

If time-based characteristics for the feeder loads, generators or both are defined so that the demand changes depending on the study case time, these states can be considered in the reliability analysis. Therefore, the load demand for a one year period can be discretised and converted into several so-called 'load states', and a probability of occurrence for each state. The Reliability Command does not automatically generate the load states. One possibility is the specification of load characteristics for individual loads and generators, and the second is by specification of load distribution states for substations. The procedures for each method is described in Chapter 18: Parameter Characteristics, Load States, and Tariffs; Sections 18.3 (Load States) and 18.4 (Load Distribution States).

### 46.3.7 Defining Fault Clearance Based on Protection Device Location

The Reliability Calculation has two options for fault clearance:

- Use all circuit breakers; or
- Use only circuit breakers controlled by protection devices (fuses or relays).

With the option "Use circuit breakers", only circuit breakers are considered as protection switches when clearing faults. If the network model contains fuses, these fuses will not be considered as 'melted' for fault clearance.

The option "Use switches with protection device" uses protection devices for fault clearance. Switches are only used if they have a protection device. Fuses are used for Fault clearing.

The second option is the more realistic option, because only locations within the network that can automatically clear a fault will be used by the reliability calculation to clear the simulated faults.

---

**Note:** If there is no protection device entered in the network model, there is the possibility to define a circuit breaker to be considered as switch with protection device for reliability calculations. This setting can be found within the circuit breakers reliability page. "Fault Clearance: Consider as switch with protection device"

---

### 46.3.8 Considering Planned Maintenance

The *PowerFactory* reliability calculation supports the definition and automatic consideration of planned network maintenance. Maintenance is implemented using a planned outage object (*IntPlannedout*), as explained in detail in Section 43.2 that also explains the process of their creation.

---

**Note:** When the reliability calculation considers outages it creates a unique contingency case for every contingency with the outage applied and also without the outage. For example, for a network with two planned outages and six contingencies there will be a total of  $6 \cdot 3 = 18$  cases.

---

### 46.3.9 Specifying Individual Component Constraints

The reliability calculation can automatically consider voltage and thermal constraints for the power restoration process. There are two options for specifying constraints applied to branch, terminal, and feeder objects as follows:

Global Constraints; All network constraints are based on the constraints specified on the constraints page of the Reliability Command dialog.

Individual Constraints; If Individual Constraints are selected for branches, terminals, and / or feeders, constraints should be defined by the user for each relevant object by taking the following steps:

1. Open the reliability page of the target terminal, branch (line/transformer), or feeder.
2. Enter the Max and Min Voltage limits, max loading, or voltage drop/rise for the terminal, branch, or feeder respectively.
3. Click **OK** to close the dialog and save the changes.

### 46.3.10 Considering Switching Rules

Reliability Analysis in *PowerFactory* allows the user to consider predefined switching rules within substations according to Section 12.2.7.4. Switching-rules are executed directly after protection operation.

## 46.4 Running The Reliability Assessment Calculation

The procedure for using the *PowerFactory* Reliability Assessment tool and analysing the results generated by the tool is described in this section.

### 46.4.1 How to run the Reliability Assessment

In *PowerFactory* the network Reliability Analysis is completed using the *Reliability Assessment* command (*ComRel3* ). This command is found on the 'Reliability Analysis' toolbar.

Alternatively, the commands can be executed for a single element by right-clicking the element and selecting *Calculation* → *Reliability Assessment...* or *Calculation* → *Optimal Power Restoration...*. The options for the reliability command that are presented within its dialog are described in the following sub-sections.

A reliability assessment is started when the **Execute** button is pressed. The calculation time required for a reliability assessment can range from a few seconds for a small network only considering n-1 contingencies, to several hours for a large network considering n-2 contingencies. A reliability assessment calculation can be interrupted by clicking on the *Break* icon () on the main toolbar.

#### 46.4.1.1 Basic Options

The following options are available on the Basic Options page Reliability Assessment Command dialog.

##### Calculation

This section defines what kind of load flow calculation will be the base for the analysis and for the constraints evaluation. The selection offers either a balanced or an unbalanced calculation. In addition, the load-flow calculation command is linked as an object for the further configuration of specific options

such as the consideration of automatic tap adjustment, the slack assignment or the voltage dependency of the loads.

### Method

- **Connectivity analysis:** this option enables failure effect analysis without considering constraints. A load is assumed to be supplied if it is connected to a source of power before a contingency, and assumed to undergo a loss of supply if the process of fault clearance separates the load from all power sources. Because constraints are not considered, no load-flow is required for this option and hence the analysis will be faster than when using the alternative load-flow analysis option.
- **Load flow analysis:** this option is the same as the connectivity analysis, except that constraints are considered by completing load-flows for each contingency. Loads might be disconnected to alleviate voltage or thermal constraints. For the transmission analysis option, Generator re-dispatch, load transfer and load shedding are used to alleviate overloads.

### Calculation time period

- **Complete year:** the reliability calculation is performed for the current year specified in the 'Date/Time of the Calculation Case'. This can be accessed and the date and time changed by clicking the → button.
- **Single Point in Time:** the Reliability Calculation is completed for the network in its current state at the actual time specified by the 'Date/Time of the Calculation Case'.

---

**Note:** If load states or maintenance plans are not created and considered, then these options make no difference because the reliability calculation is always completed at the single specified time.

---

### Network

- **Distribution:** the reliability assessment will try to remove overloading at components and voltage violations (at terminals) by optimising the switch positions in the system. If constraints occur in the power restoration process, loads will be shed by opening available switches. This option is the recommended analysis option for distribution and medium voltage networks.

---

**Note:** The reliability command optimises switch positions based on load shedding priorities, and not network losses.

---

- **Transmission:** thermal overloads are removed by generator re-dispatch, load transfer and load shedding. First generator re-dispatch and load transfer is attempted. If these cannot be completed or do not remove the thermal overload, load shedding actions will occur. Generator re-dispatch and load transfer do not affect the reliability indices. However, by contrast, load shedding leads to unsupplied loads and therefore affects the reliability indices.

### Automatic Contingency Definition

The command will execute individual contingency objects which are called *ComContingency* and are stored in the *Reliability Assessment* command. They define a network element (or several elements) that is faulted and leads to the potential customer outage.

If the checkbox is selected, new contingencies will be created. If it is unchecked, existing contingencies from previous calculations will be used for reliability assessment.

The 'Selection' list presents two possible options for the contingency definition. These are:

- Whole system: *PowerFactory* will automatically create a contingency event for every object that has a Stochastic Model defined.

- User Defined: Selecting this option shows a selection control. Now you can select a set of objects (*SetSelect*), and contingencies will be created for each of these objects that has a Stochastic Model defined.

In addition to the above contingency definition options, the automatic contingency definition can be further controlled with the following, mostly self explanatory checkboxes:

- Busbars/Terminals
- Lines/Cables/Branch
- Transformers
- Generators: This flag is only available for the Load flow analysis method.
- Shunts/Filters/Ser. Impedances: Covers the parallel and series elements.
- Common Mode; This flag should be enabled for *PowerFactory* to create Common Mode contingencies. See Section [46.3.1.6](#) (Common Mode Stochastic Model) for more information.
- Independent second failures; This flag should be enabled for *PowerFactory* to consider n-2 outages in addition to n-1 outages. Caution: n-2 outages for all combinations of n-1 outages are considered. This means that for a system of n contingencies there are  $(n \cdot (n - 1))/2 + n$ , contingencies to consider. This equation is quadratic, and so to minimise the required time for computation this option is disabled by default.
- Double-earth faults; This flag should be enabled for *PowerFactory* to consider double-earth faults. See Section [46.3.1.8](#) (Double Earth Faults) for more information.
- Protection/switching failures; This flag should be enabled for *PowerFactory* to consider the failure to operate of protection devices or circuit breakers. See Section [46.3.1.7](#) (Protection/Switch Failures) for more information.
- Spurious protection operation; as explained in Section [46.3.1.7](#).
- Backup protection maloperation; as explained in Section [46.3.1.7](#).

#### 46.4.1.2 Outputs

The following options are available on the *Outputs* page of the Reliability command.

##### Results

This option allows the selection of the result element (*ElmRes*) where the results of the reliability analysis will be stored. Normally, *PowerFactory* will create a results object within the active study case.

##### Show detailed output of initial load flow and top level feeders

If this option is checked, a detailed report of the initial load flow will be printed to the output window.

#### 46.4.1.3 Protection

##### Fault Clearance Breakers

- **Circuit breakers:** all switches in the system whose *aUsage* is set to *Circuit Breaker* can be used for fault clearance.
- **Switches with protection device:** all circuit breakers in the system which are controlled by a protection device (fuse or relay) can be used for fault clearance. Circuit breakers which are set to have a protection device are also considered.

Further information is provided in Section [46.3.7](#).

### Create Contingencies

These settings are the same as in “Automatic Contingency Definition”, described in Section [46.4.1.1](#). For convenience they are displayed within this tab as well.

#### 46.4.1.4 Restoration

##### Automatic Power Restoration

The options described below will only be available if Automatic Power Restoration is selected.

##### Load/Generator Priorities

The two settings will be used to evaluate the element's priority value, entered by the user.

- **Lowest priority number refers to most critical load/generator:** this means that higher priorities are shed first.
- **Highest priority number refers to most critical load/generator**

##### Switching procedures for fault separation/power restoration

- **Concurrent Switch Actions:** it is assumed that the switching actions can be performed immediately following the specified switching time. However, a switch can be closed for power restoration only after the faulted element was disconnected. The analogy for this mode is if there were a large number of operators in the field that were able to communicate with each other to coordinate the switching actions as quickly as possible. Therefore, this option gives an optimistic assessment of the ‘smart power restoration’.
- **Sequential Switch Actions:** it is assumed that all switching actions are performed sequentially. The analogy for this mode is if there were only a single operator in the field, who was required to complete all switching. The fault separation and power restoration is therefore slower when using this mode compared with the ‘concurrent’ mode.
- **Consider Sectionalising (Distribution analysis only):** if enabled, the FEA considers the switch sectionalising stage when attempting fault separation and power restoration.

##### Time to open remote controlled switches

The time (in minutes) taken to open remote controlled switches.

##### Tie Open Point Optimisation

After the isolation of failures, parts of the network may be unsupplied. However, the network can be reconfigured by moving the tie open point in order to restore as much power as possible (partial power restoration). This reconfiguration might lead to violations of constraints (e.g. overloading), which should be avoided. For each sectionalising stage of switches, the optimisation method offers three power restoration modes:

- Disabled: no movement of tie open points
- Enabled without load transfer: tie open points can only be moved between the feeder affected by the fault and a directly-bordering feeder. Only unsupplied loads are then transferred to and restored by the neighbouring feeder. An exception applies in the event that constraints are violated after the fault isolation, e.g. due to the loss of generation. In this case, loads would have to be shed in order to comply with constraints. A load transfer can still be carried out for these loads, as they would otherwise remain unsupplied.
- Enabled with load transfer: tie open points can be moved between a bordering feeder and a second-level bordering feeder. In this case supplied loads not affected by the fault can also be

transferred to other feeders to increase the flexibility of the restoration algorithm.

First sectionalising is attempted using only stage 1 switches, if this is not successful then stage 1 and 2 switches are used. Finally, if this is not successful, then stage 1, 2 and 3 switches are used.

If *Consider Sectionalising Actions* is disabled, the stage of each switch is ignored and all switches will be considered equally with one of the above mentioned methods.

### Supplying substations

**Backward recovery:** Whenever a busbar, being the starting point of one or more feeders, is de-energised after a fault (e.g. of an HV/MV-transformer), closing tie open points in one of the feeders can restore the supply of the busbar and the other feeders from a neighbouring substation. This so-called backward recovery is available for networks with explicit substation modelling (*ElmSubstat*, *ElmTrfstat*). The algorithm finds the best feeder for resupplying the substation and the interrupted feeders. This can improve the restoration quality for a loss of a substation significantly (see Figure 46.4.1).

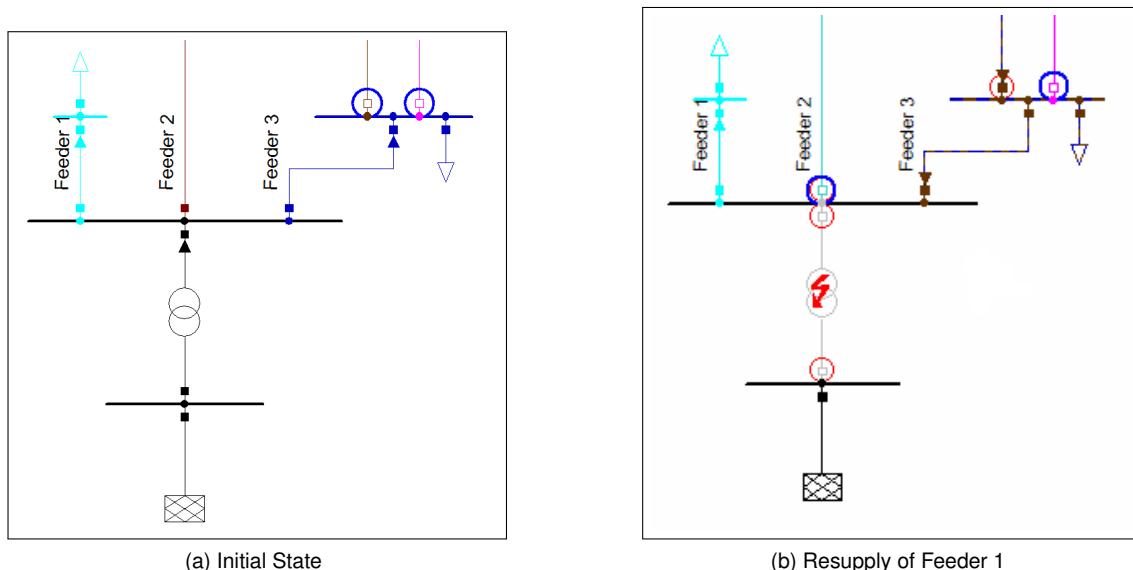


Figure 46.4.1: Example of Backward Recovery

The Backward recovery offers the following options:

- Do not allow: backward recovery will not be considered for restoration
- Allow but prefer standard recovery: backward recovery will only be used if the other restoration options fail.
- Allow with user-defined preference: substations where backward recovery is allowed can be selected here.
- Allow and prefer: whenever possible, backward recovery will be preferred over the other restoration options.

**Busbar transfer:** Whenever a busbar in a multi-busbar substation with multiple transformers is unsupplied as a consequence of a fault, closing coupling breakers within the substation may be an option for restoration. This transfer of a busbar to the supply of another transformer is available in *PowerFactory*, with one of these three options:

- Simple reconnection: an unsupplied busbar will be connected to the adjacent busbar. If constraints violations remain, further measures will be considered, e.g. load shedding within feeders.
- Optimised without additional meshes: the algorithm will optimise the coupling breaker positions in the substations to supply as much as possible of the unsupplied feeders, taking into account

the constraints. The transfer of busbars not directly affected by the fault to other transformers is possible. The switching actions will not lead to additional meshes, e.g. the parallel operation of transformers due to the busbar transfer.

- Optimised with and without meshes: Same as “Optimised without additional meshes” but with the further freedom for the algorithm of creating new meshes, e.g. leading to parallel operation of transformers.

### Enhanced restoration

It may happen that the load flow calculation does not converge for the basic restoration process. If this checkbox is enabled, more restoration strategies are examined in this case to find a solution that converges. Setting this option can decrease the performance of the Reliability assessment if non-converging load flows have to be prevented.

#### 46.4.1.5 Costs

##### Costs for energy not supplied

If this option is selected, an Energy Tariff can be selected. Energy Tariffs are discussed in Chapter 18: Parameter Characteristics, Load States, and Tariffs, Section 18.5.2(Defining Energy Tariffs).

##### Costs for loads

If this option is selected, a Global cost curve for all loads can be selected. Alternatively, 'Individual cost curve per load' may be selected, allowing the user to define tariffs for individual loads. In both cases, a Time Tariff or Energy Tariff may be defined, as discussed in Chapter 18: Parameter Characteristics, Load States, and Tariffs, Section 18.5 (Tariffs).

#### 46.4.1.6 Constraints

For the method “Load flow analysis” (see Basic Options in Section 46.4.1.1), this page allows the user to define the consideration and the limits for various constrained quantities. For the “Transmission” option only the thermal constraints are available, for the “Distribution” option, the whole set of quantities is available to be considered during the Optimal Power Restoration.

##### Consider Thermal Constraints (Loading)

If this option is enabled, thermal constraints are considered by the FEA.

- **Global constraints for all components:** constraints specified in 'Max thermal loading of components' apply to all components in percent value.
- **Individual constraint per component:** the maximum thermal loading limit is considered for each component separately. This loading limit can be found on the Reliability page of each component.

##### Consider Voltage Limits (Terminals)

If this option is enabled terminal voltage limits are considered by the FEA.

- **Global Constraint for all terminals:** constraints specified in Lower and Upper Limit of allowed voltage in p.u. that will apply to all terminals.
- **Individual Constraint per terminal:** voltage constraints are considered for each terminal separately. These constraints can be found on the Reliability page of each terminal.

##### Consider Voltage Drop/Rise

If this option is enabled feeder voltage limits are considered by the FEA.

- **Global Constraint for all feeders:** constraints specified in Maximum Voltage Drop and Rise in percent value that will apply to all feeders.
- **Individual Constraint per feeder:** voltage Drop/Rise constraints are considered for each feeder separately. These constraints can be found on the Reliability page of each feeder.

### Consider Boundary Constraints outside feeders

If this option is set, the boundary constraints, applied on the boundaries “Reliability” settings are considered during restoration.

### Ignore all constraints for

Constraints are ignored for all terminals and components below the entered voltage level.

- **Nominal voltage below or equal to:** the voltage level in kV is specified here if ‘Ignore all constraints for...’ is enabled.

### 46.4.1.7 Maintenance

This page allows you to enable or disable the consideration of Maintenance based on the Planned Outages you have defined. See Section 46.3.9, for more information on defining planned outages. The following options are available on this page:

#### Consider Maintenance

If enabled, all maintenance that falls in the selected time period, whether it's a year or a single point in time, is considered.

- **Show used planned outages:** when clicked, this button will show a list of all planned outages that will be considered by the calculation.
- **Show all planned outages:** when clicked, this button will show a list of all planned outages created in the project, including those not considered by the analysis because they fall outside of the selected time period.

### 46.4.1.8 Load Data

If the Reliability Calculation option ‘Complete Year’ is selected on the basic options page, then the following options are available on the Load Data page.

#### Load Variations

Enable the relevant flag to consider a constant load, load states (include generator states, if created with corresponding settings) or load distribution states according to Section 46.3.6 in the reliability calculation. The reliability calculation does not create load states automatically. If this flag is enabled but the states have not been created, then an error will be written to the output window and the reliability calculation will stop. Otherwise the following two options are available:

#### Update/creation of States

- **Manually:** if selected, a button ‘Create load states’ will be available. When clicked, it launches the ‘Load state creation’ command after closing the reliability command (see Chapter 18 for more information on load state creation).
- **Automatically before running reliability calculation:** when selected, a pointer to the load state creation command is available.

#### 46.4.1.9 Advanced Options

##### Events created during restoration

- **Only store them in the results file:** events will only be stored in the results file and not be saved as separate events in the contingency. This minimises the number of objects created in the database while performing calculations with many contingencies in large networks (e.g. if independent second failures or double earth faults are enabled).
- **Also save them in the corresponding contingency:** switch events, load shedding and load transfer as well as generator redispatch events will be saved in the corresponding contingency.

##### Stop calculation if base case has constraint violations

It can happen that the load flow for the base case already leads to the violation of constraints for some components. If this option is set, the reliability assessment will stop in this case. If it is not activated, a user-defined percentage can be specified to relax the constraints of initially overloaded elements.

##### Calculation of SAIFI/SAIDI according to IEEE 1366

- **Do not consider interruptions shorter than or equal to:** Only interruptions which last longer than the entered duration will be classified as sustained interruptions and considered in the calculation of the corresponding indices (e.g. SAIFI/SAIDI, see Section 46.2.3.3).

For the calculation of the MAIFI, the following cases leading to momentary interruptions are considered:

- transient faults defined in the stochastic models of lines for which contingencies are being evaluated in the reliability assessment
- if a recloser is involved in the fault clearance of a sustained fault, its reclosing attempts will be counted as momentary interruptions

Reclosers can be configured as follows:

- ticked checkbox “Consider as switch with automatic reclosing device” on the Reliability page
- relay including a reclosing block assigned to the breaker

##### Enhanced consideration of automatic reclosing devices:

With this option ticked, transient faults on lines which are not protected through reclosers will lead to sustained interruptions and be taken into account in the corresponding reliability indices.

##### Trace Functionality (Jump to Last Step)

A user-defined ‘Time delay in animation’ can be entered to delay the animation of power restorations when the *Jump to Last Step* icon  is pressed.

##### Switch/Load events

- **Delete switch events:** A click on the button removes all switch events associated with the contingencies stored inside the command.
- **Delete load events:** A click on the button removes all load and generator events associated with the contingencies stored inside the command.

##### Calculation of Load Shedding in Transmission Networks

The parameter “Only consider branch if loading before shedding exceeds” is a performance setting and available only for the Transmission Network selection on the Basic Data page (see Section 46.4.1.1). For the default value of 0 % all branch elements, respectively their loading, will be considered for the optimised load shedding algorithm. In order to reduce the size of the optimisation problem, branches with a comparably low loading in the base case may be excluded without any impact on the results.

#### 46.4.1.10 Parallel Computing

Parallel calculation of *Reliability Assessment* is possible and can be activated on the Parallel Computing page of the *Reliability Assessment* command dialog. The options provided on this page are described below and only available if the parallelisation is activated for the user in the user settings.

**Parallel computation of contingencies:** if the checkbox is ticked, the Reliability Assessment is executed in parallel. Otherwise, the calculation is run sequentially.

**Minimum number of contingencies:** this parameter defines the minimum number of contingencies necessary to start a parallel calculation. This means, if the number of contingencies is less than the entered number, the calculation is run sequentially.

**Parallel Computing Manager:** the parallel computation settings are stored in a *Parallel Computing Manager* object (*SetParalman*). Further information on the particular configuration is found in Section [6.6.1](#).

## 46.5 Results of the Reliability Analysis

### 46.5.1 Load Reliability Results

The Load Reliability Results *Load Reliability Results*  icon can be accessed in the Reliability Analysis toolbox. This will bring up the *Load Reliability Results* (*ComRelresload*) command, for the evaluation of the results file.

The *Reliability Analysis* automatically writes all simulation results to a result object specified above. After completing the Reliability Calculation, *PowerFactory* automatically evaluates the result object to compute the reliability indices. A new reliability calculation will overwrite the existing result object. This can be avoided by making a copy of the result object or using different study cases. The efforts for reliability calculation can be very high depending on the size of the network. To avoid repeated calculations for the same network states, the existing result object can be evaluated once a reliability analysis calculation has been executed. The command *Load Reliability Results* allows you to re-evaluate a result object that has previously been created by this or another reliability calculation command.

The *Load Reliability Results* command has the following options.

- **Result:** Enables the selection of a reliability result object. By default it is set to the result object on the output tab of the reliability command in the active study case.
- **Considered load interruptions:** Option to consider the evaluation for either all load interruptions or for a set of selected loads.

The benefit of this is that you do not have to re-run the reliability calculation (which can be time consuming compared with the results object evaluation) if you only want to recalculate the indices from an already completed calculation. Furthermore, it allows to evaluate contributions to interruptions of specific loads only.

### 46.5.2 Creating Reports using the Report Generation command

The main *Reliability Analysis* reports can be accessed via the *Reliability Reports*  icon in the Reliability Analysis toolbox. This will bring up the Report Generation (*ComReport*) command, where reports can be selected from the list of available reports. Reports can be generated as separate documents or combined into one. By default, reports are generated as PDFs and are presented in *PowerFactory*'s inbuilt PDF viewer, but it is also possible to export reports from *PowerFactory* in various different formats.

For more information about reports and the Report Generation command, see Chapter 19 ([Reporting and Visualising Results](#)), Section 19.5.

It should be noted that reports viewed internally in *PowerFactory* are stored in the desktop of the study case, even after they have been closed, until they are actively deleted by the user.

By default, results for the whole network are reported, but this can be customised on the Filters page, as described in Section 46.5.2.1 below. For example, selecting a set of loads or terminals will restrict the output in the Load Interruptions or Node Interruptions respectively.

The available calculation-specific reports are:

- **Contribution of Component Groups to System Indices:** reports the contribution of Lines, Cables, Transformers, Terminals, Generators, Common Mode Failures and Double-Earth Faults to the system indices.
- **Load Interruptions:** reports the following indices for loads.
  - TCIT
  - TCIF
  - AID
  - LPENS
  - LPIC
  - ACIF
  - ACIT
- **Node Interruptions:** reports the following indices, focused on nodes.
  - AIT
  - AIF
  - AID
- **Recovery Scheme:** reports the recovery scheme for a selected contingency case. The contingency case is selected using the down-arrow next to “Contingency”, underneath the “Available Reports” box.
- **System Summary:** reports reliability indices per grid and for the whole system.

#### 46.5.2.1 Using filters in the Report Generation command

The Report Generation command dialog has a Filters page. As a general rule, as described in Chapter 19 ([Reporting and Visualising Results](#)) in Section 19.5.1.3, this allows the user to restrict the reporting to just elements of interest.

However, for the *Contribution of Component Groups to System Indices* report and the *System Summary* report, some specific additional functionality is implemented. In some cases, the user is not so much interested in the contributions of the various contingencies to the *overall* reliability indices, but rather the contributions to the reliability indices for a *specific load or group of loads*. This is achieved by selecting loads, or grouping objects for loads (.e.g. zones) on the Filters page, which (as well as acting as a general filter) triggers a post-processing of the results for these particular reports.

#### 46.5.3 Tabular report of Contributions

This report is generated using the  icon. It returns the contribution of individual elements to the system indices in a tabular form. This contribution is given as an absolute value and in percent.

The value columns allow the sorting and filtering of rows and corresponding contingency-related elements to quickly identify the most critical components in the network affecting the various reliability indices.

It is possible to generate a report of the contributions to the interruption of all or selected loads. Loads can be specified by selecting individual loads or object groups such as Networks, Areas, Zones and Feeders. Selecting a feeder is the equivalent to selecting all the loads in that feeder. The results displayed in the tabular report or corresponding colouring in single line diagrams therefore do not relate to the overall system indices but to the 'system indices' for the selected loads.

The "Tabular report of Contributions" has the following options.

- "Contributions to": Contributions can be displayed for SAIFI, SAIDI, ASIFI, ASIDI, ENS and EIC.
- "Report Contributions to Load interruptions": Option to consider the evaluation for either all load interruptions or for a set of selected loads.
- "Result": Enables the selection of a reliability result object. By default it is set to the result object on the output tab of the reliability command in the active study case.

#### 46.5.4 Selecting loads for post-processing

As described in Section 46.5.2.1 above, it is possible to create a report of contributions to the reliability indices for specific loads.

In addition to the report, the user may wish to view such post-processed results in a diagram or the Flexible Data page. To achieve this, the user can run the *Contributions to Reliability Indices ComRelpost* command. This is done by going to the study case in the *Data Manager* and using the new object icon  to create a *ComRelpost* command object. The command can be configured as required, then executed. (In fact, a "legacy" ASCII report will also be generated in the output window.)

Running this command before running the reporting command (*ComReport*) will not affect the content of the reports. The content of the reports is instead managed using the filter options in the reporting command, as described above. However, it will affect the diagram colouring and output shown on the Flexible Data page, as described in the following two sections.

To see results for the full network again, the *ComRelpost* command can be re-run with the filter deselected; it is not necessary to re-run the Reliability Assessment.

#### 46.5.5 Viewing Results in the Single Line Diagram

You can view the Reliability Assessment Load Point Indices in three ways: in the load result boxes in single line graphic, in the data browser (Data Manager, Network Model Manager or filtered element lists) or according to the diagram colouring. This sub-section describes the first two of these methods. The third method is described in Section 46.5.5.2.

##### 46.5.5.1 Result Boxes

After you have executed the Reliability Assessment Calculation, all loads within the Network Single Line Graphic, will show the following load point indices:

- LPIF: Load Point Interruption Frequency.
- LPIT: Load Point Interruption Time.
- LPIC: Load Point Interruption Costs.

As usual, with *PowerFactory* result boxes, you can hover the mouse pointer over the result box to show an enlarged popup of the results.

---

**Note:** You can show any of the calculated load point indices in the load result boxes. To do this modify the displayed variables as described in Chapter 19: Reporting and Visualising Results, Section 19.3 (Variable Selection)

---

#### 46.5.5.2 Diagram Colouring

For further analysis, which element contributes most to the reliability of a certain selection of customers, it is possible to use the diagram colouring. The use of colouring when viewing the results of Reliability Analysis is described below, but for general information about using colour in network diagrams please refer to Section 10.3.10.1, and more detail about the use of colours and colour palettes can be found in Section 4.7.8.

---

**Note:** The colouring seen in the diagram will relate to the currently loaded results. That is, if results relating to particular loads have been selected using the command described in Section 46.5.4, the colouring will reflect these results.

---

The diagram colouring, especially for branches, terminals, MV Loads and generators can be according to

- contribution to EIC,
- contribution to ENS,
- contribution to SAIDI,
- and contribution to SAIFI.

The colouring, especially for Loads, can be according to

- average interruption duration,
- load point energy not supplied,
- yearly interruption frequency,
- and yearly interruption time.

In addition, there are several colouring modes that can aid you when using the reliability assessment functions. These are:

- Colouring according to *Feeders*; Use this to identify each Feeder and to see which feeder picks up load when back-feed switches are closed.
- Colouring according to *Connected Grid Components*; Use this to identify de-energised sections of the network during the fault isolation, separation and power restoration.
- Switches, type of usage. Use this mode to check the type of switches in the system when they are not modelled explicitly in the single line diagram.

#### To Colour According to Feeders

1. Click the *Diagram Colouring* button . The Diagram colouring dialog will appear.
2. Select the tab for the function you want to show the colouring mode for. For example, if you want the feeder colouring to appear before a calculation, then select the *Basic Data* tab. If you want the colouring to appear after a load-flow choose the *load-flow* tab.

3. Check the *3. Other* box and select *Topology* from the drop down list.
4. Select *Feeders* in the second drop down box.
5. Optional: To change the feeder colour settings click the *colour settings* button. You can double click the displayed colours in the colour column and select a different colour for each feeder as desired.
6. Click **OK** to close the Diagram Colouring dialog and save your changes.

#### To Colour According to Connected Grid Components

The *Connected Grid Components* colouring mode displays all the network components that are electrically connected together in the same colour. Other components are not coloured. To enable this mode:

1. Click the *Diagram Colouring* button . The diagram colouring dialog will appear.
2. Select the load-flow tab.
3. Check the *3. Other* box and select *Topology* from the drop down list.
4. Select *Connected Grid Components* in the second drop down box.
5. Click **OK** to close the Diagram Colouring dialog and save your changes.

#### To Colour According to Switch Type

The *Switches: type of usage* colouring mode displays all switches in the network with a different colour depending on their *switch type*. For instance circuit breakers will be displayed in a different colour to disconnectors. To enable this mode:

1. Click the *Diagram Colouring* button . The diagram colouring dialog will appear.
2. Select the tab for the function you want to show the colouring mode for. For example, if you want the switch type colouring to appear before a calculation, then select the *Basic Data* tab. If you want the colouring to appear after a load-flow choose the load-flow tab.
3. Check the *3. Other* box and select *Secondary Equipment* from the drop down list.
4. Select *Switches, Type of Usage* in the second drop down box.
5. Optional: To change the switch colour settings, click the *colour settings* button. You can double click the displayed colours in the colour column and select a different colour for each switch type as desired.
6. Click **OK** to close the Diagram Colouring dialog and save your changes.

#### 46.5.6 Viewing Results in the Data Browser

To view the load point and system reliability indices in the Data Browser (as a selectable spreadsheet list), follow these steps:

1. Select the element or grouping element icon from the *Network Model Manager* button .
2. Choose the *Flexible Data* tab.
3. Click the *Define Flexible Data* button , to show all available variables.
4. Add more variables to the *Selected Variables* by double-clicking the variable in the *Available Variables* window.
5. Click **OK** to view the result variables in the data browser.

**Note:** Steps 3-5 are only required the first time you want to view the system reliability indices, or if you want to change the displayed variables. *PowerFactory* 'remembers' these settings within each project.

---

**Note:** The results displayed will be the currently loaded results. That is, if results relating to particular loads have been selected using the command described in Section 46.5.4, this is what will be shown.

---

## 46.6 Loss of Grid Assessment

A power station connection to the grid generally provides both a connection for the generated electricity exported from the station and a separate connection for the supply of grid electricity to the station. Of principal concern to critical power station operators is the potential for loss of the grid connection to the station. Failures of the grid external to the power station are commonly referred to as Loss of Grid events (LoG) and the corresponding frequency as Loss of Grid Frequency (LoGF) in probabilistic analysis studies. Loss of Grid events are in effect "islanding" or isolation of a component from the grid. The Loss of Grid Assessment (LoGA) in *PowerFactory* considers a user-defined time period, with Areas, Zones or Boundaries delimiting the substations to be considered. The tool can analyse the impact of various short-term events such as bad weather. LoGF is calculated for Substations and Secondary Substations.

Failures of the following components are considered by LoGA:

- Lines and Cables
- Transformers
- Busbars
- Shunts

Only those components with reliability failure data are considered for the LoGA. The LoGA command has several options, which are configured depending on data available.

### 46.6.1 Basic Options

**Substation:** The "Substation" refers to the substation (also transformer substation) whose LoGF has to be calculated.

#### Calculation time period

- **Complete year:** the Loss of Grid Assessment is carried out for the current year, according to the Study Case date. This can be accessed and the date and time changed by clicking the (→) button.
  - **User defined time range:** the LoGA can be carried out for a selected time range. This is the case when certain events have to be considered. It is assumed that for that time the failure rate of components.
  - **Begin:** Starting Date and time for time-banded assessment
  - **End:** End Date and time of the period to be considered.
-

**Note:** Historical data indicates that, for example, overhead line related failure rates become highly elevated during the relatively short periods of adverse weather (i.e. single or combined conditions of high winds, rainfall, snowfall and lightning) experienced in different seasons and different regions of a network. For calculations for the whole period *PowerFactory* incorporates the average effect of such weather effects over an annual period. By selecting the time range, the calculation of time-banded risk level of grid supply disconnection while experiencing periods of adverse weather can be assessed.

---

For time-banded calculations the change in failure rates can be modelled by using characteristics on the corresponding failure parameters.

#### Common Mode Failures

- **Using existing:** A common mode failure involves the simultaneous failure of two or more power system components. For the definition of Common Mode see Section 46.3.1.6. For this option, it is possible to “Show”, “Add” or “Remove All” common modes from the list of common modes to be used for the LoGA.
  - **Detect automatically:** With this option *PowerFactory* automatically determines the common modes leading to LoG within the selected region, up to a given order.
- 

**Note:** The common mode failures are created systematically and checked to avoid topological duplicates with those created manually by the user. If topological duplicates are detected, one of them is reported and removed from the list of common modes. Meaningful names are created for common modes based on the names of elements in the common modes.

---

- **Search Region:** This is the region where common modes leading to LoG are automatically detected. This can be a zone, area or a boundary.
- **Maximum Order:** This is the maximum order of common modes.

#### 46.6.2 Protection

For the fault clearance during LoGA, it is possible to select either the option “Use circuit-breakers” or “Use switches with protection device”.

##### Fault Clearance Breakers

- **Use circuit breakers:** All circuits breakers can be used for the clearing of faults.
- **Use switches with protection devices:** With this option *PowerFactory* automatically determines all switches having protection devices as described in Section 46.3.7 and uses these switches for fault clearing.

#### 46.6.3 Output

**Results:** A reference (pointer) to the results object.

**Show load flow message:** During the LoGA a series of load flows are carried out. Messages written to the output window are by default suppressed. These can be enabled by activating this option.

#### 46.6.4 Results

The LoGA is carried out each time for one substation. The LoGA produces two basic result variables for the relevant substation or secondary substation, which can be seen in a Network Model Manager:

- **e:logf\_freq**: Loss of grid frequency
- **e:logf\_year**: Time period between loss of grid events

All the common mode failure objects used or created during the LoGA are stored in the study case and can be explored using the button **Failures** on the Output page of the command. Each common mode contains the various components whose simultaneous failure would lead to a LoG.

# Chapter 47

## Optimal Power Restoration

The optimal power restoration functions can be accessed by activating the Optimal Power Restoration toolbar using the icon on the toolbar selection control as illustrated in Figure 47.0.1

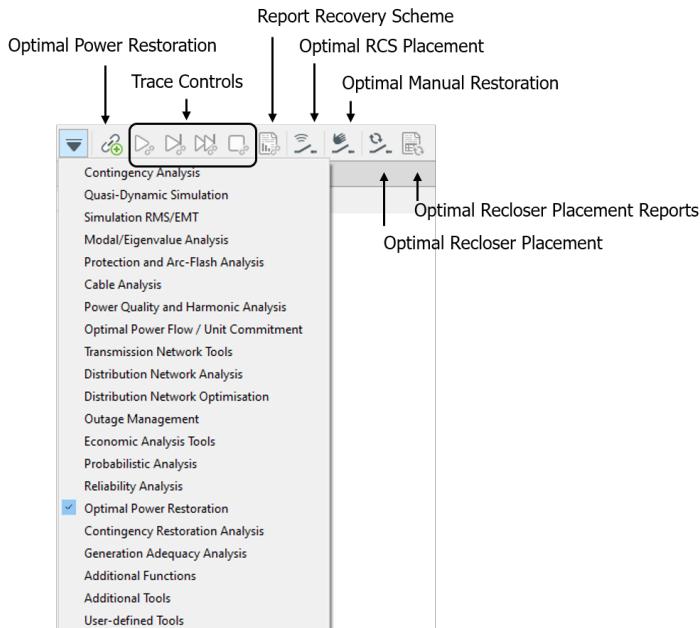


Figure 47.0.1: Optimal Power Restoration Selection

### 47.1 Failure Effect Analysis

The simulation of the system response to specific contingencies (*ComContingency*) is called 'Failure Effect Analysis' (FEA). The System State Enumeration algorithm uses the FEA engine to analyse the following steps after a contingency:

- Fault Clearance;
- Fault Isolation;
- Power Restoration;
- Overload Alleviation;
- Voltage Constraint Alleviation;

- Load Transfer;
- Load Shedding;

FEA analysis for the network assessment can consider or ignore constraints. For overload alleviation, the algorithm uses an AC load flow to search for overloaded branches and if any are identified then it attempts to resolve them, firstly by load transfer and secondly by load shedding. If constraints are not considered by the FEA, then a load-flow for each state is not required and consequently the simulation is much faster.

For every simulated failure, a contingency is created by the FEA algorithm. If the calculation uses load characteristics, a contingency is created for every combination of failure and load state. Likewise, when maintenance (planned outages) are considered, there are more states for each outage and contingency combination.

### Fault Clearance

The fault clearance step of the FEA assumes 100% selectivity of the protection. Therefore, it is assumed that the relays nearest to the failure will clear the fault. If protection/switching failures are considered in the FEA, it is assumed that the next closest protection device (after the failed device) has 100% selectivity. As described in (Protection/Switch Failures), *PowerFactory* does not consider separate switch and protection failures, instead these are lumped together. In the pre-processing phase of the reliability assessment, all breakers in the system that can be tripped by a relay, or fuse are marked as 'protection breakers'.

To clear the fault, the FEA starts a topological search from the faulted component/s to identify the closest protection breaker/s that can clear the fault. These breaker/s are then opened to end the fault clearance phase of the FEA. If it is not possible to isolate the fault because there are no appropriate protection breakers, then an error message will be printed and the reliability assessment will end.

### Fault Isolation

The next step of the FEA is to attempt to restore power to healthy network sections. It does this by separating the faulted section from the healthy section by opening sectionalising switches.

The fault separation procedure uses the same topological search for switches as the fault clearance phase. The fault separation phase starts a topological search from the faulted components to identify the closest switches that will isolate the fault. These switches are subsequently opened. Note, all closed switches can be used to separate the faulted area. The area that is enclosed by the identified fault separation switches is called the 'separated area'. The separated area is smaller than, or equal to, the 'protected area'. It will never extend beyond the 'protected area'.

The healthy section which is inside the 'protected area', but outside of the 'separated area' is called the 'restorable area' because power can be restored to this area.

### Power Restoration

The Power Restoration process of the FEA energises the healthy areas of the system after the fault separation process has isolated the faulted area. Note that only open switches that are enabled for use in power restoration will be considered by *PowerFactory* as candidate switches for power restoration. Additionally, *PowerFactory* uses a 'smart power restoration' procedure that also considers the direction of the power restoration and the priority (stage) of the switch. The fastest candidate switch is always selected when there is more than one restoration alternative. Each restorable area that is reconnected to the supplied network is called a 'restored' area. For more information about the switch configuration for smart power restoration, see Section 46.3.3.

If switching actions are not possible in order to return loads and terminals in a separated area to service, then these loads and terminals will remain interrupted for the mean duration of the repair, which is normally several hours. However, if switching actions are possible to return the loads and terminals to service, they will only be interrupted for the time needed to open all separators and to close all power restoration switches. The effects of network upgrades, including improved automation and remote

control of switches (by lowering switch actuation times), can be analysed.

An Optimal Power Restoration can also be conducted for a single contingency from outside the reliability calculation through the Optimal Power Restoration command shown in Figure 47.0.1, or by right-clicking an element and selecting *Calculation* → *Optimal Power Restoration*....

### Overload Alleviation

If the power restoration does not cause any thermal overloads or voltage violations (if applicable), then the FEA can proceed to calculate the statistics for that state and then analyse the next state. However, if thermal constraints are enabled, then *PowerFactory* will complete load-flows to check that all components are still within their thermal capability after the power restoration is complete. If necessary, load transferring, partial or full load shedding might be required to alleviate the thermal over-load. Note load transferring and partial load shedding are only considered when 'Transmission' is selected in the Reliability command Basic Options. The distribution option considers only discrete switch actions. Therefore, loads must be fully shed or remain in service.

#### Voltage Constraint Alleviation (Distribution Option only)

If the 'Distribution' option is selected in 'Basic Options', voltage constraints for busbars/terminals and feeders can be considered in addition to thermal constraints. The voltage constraint alleviation process is similar to the thermal overload alleviation process, where loads will be shed if necessary to maintain system voltages within the defined limits.

#### Load Transfer (Transmission Option only)

In some cases, load transfer switches and/or the alternative feeders are not included in the network model where reliability assessment is completed. In these cases, the automatic power restoration cannot switch an unsupplied load to an alternative supply. An example is when a (sub-)transmission network is analysed and the connected distribution networks are modelled as single lumped loads. In this scenario, transfer switches that connect two distribution networks will not be visible. Therefore, the possibility of transferring parts of the lumped load model to other feeders can be modelled by entering a transfer percentage at each lumped load. This transfer percentage defines the portion of the lumped load that can be transferred 'away' from the analysed network, without specifying to which feeder/s the portion is transferred.

The use of the load transfer percentage (parameter name: *Transferable* on the load element's *Reliability* tab) is only valid when load transfer is not expected to result in an overloading of the feeders which pick up the transferred loads.

Load transfer is used in the overload alleviation prior to the calculation of power at risk (see the following section for further information). The power at risk is considered to be zero if all overloads in the post-fault condition can be alleviated by load transfers alone.

---

**Note:** Load Transfer, either to a non-explicitly modelled network or the elements defined in each load, is only executed if a post-contingency overload occurs (immediately or during the restoration process).

---

### Load Shedding

There are three basic variations of shedding that can be used:

- Optimal load shedding.
- Priority optimal load shedding.
- Discrete optimal load shedding.

Optimal load shedding presumes that all loads can be shed precisely (an infinite number of steps).

*PowerFactory* attempts to find a solution that alleviates the overload with the lowest amount of load shed.

*PowerFactory* uses linear sensitivity indices to first select those loads with any contribution to overloading. A linear optimisation is then started to find the best shedding option. The resulting minimum amount of shed load is called the 'Power Shed', because it equals the minimum amount of load that must be shed to alleviate overloads after the power restoration. The power shed is multiplied by the duration of the system state to get the 'Energy Shed'. The total energy shed for all possible system states is reported after the reliability assessment is complete, and is referred to as the 'System Energy Shed' (SES).

Loads are shed automatically based on their allocated priority, with *PowerFactory* attempting to shed low priority loads, prior to high priority loads wherever possible. In the transmission reliability option, loads can be partially or fully shed, whereas in the distribution option, loads can only be fully shed.

### Example

Figure 47.1.1 shows a simple network containing four loads, several circuit breakers (CB) and disconnectors (DS) and a back-feed switch (BF).

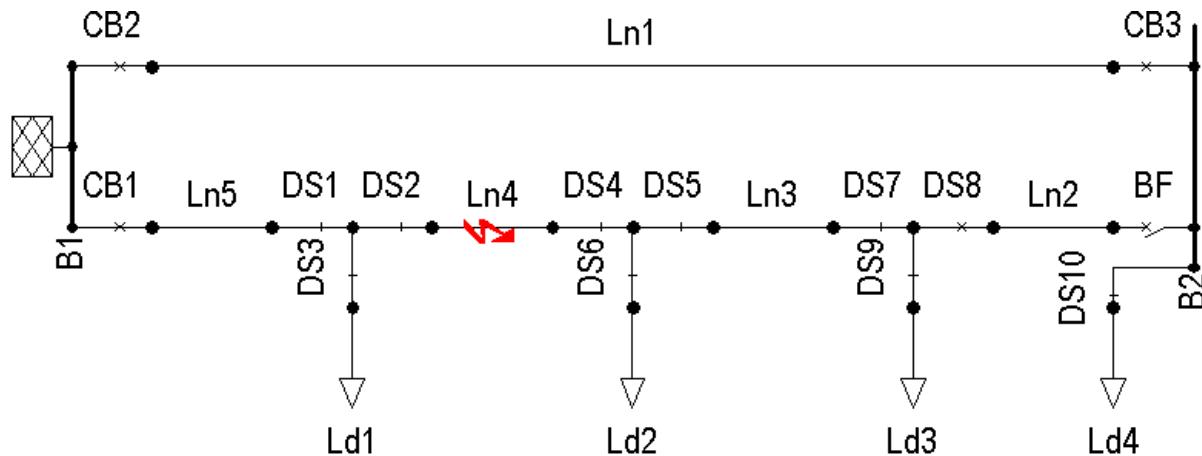


Figure 47.1.1: Short-Circuit on Ln4

### Fault clearance

The area isolated by the fault clearance procedure is called the 'protected area'. Figure 47.1.2 shows the example network after the fault clearance functions have opened the protection breaker 'CB1'. The protected area is the area containing all switches, lines and loads between 'CB1' and the back-feed switch, 'BF'. Therefore, during the clearance of this fault, loads 1, 2, and 3 are interrupted.

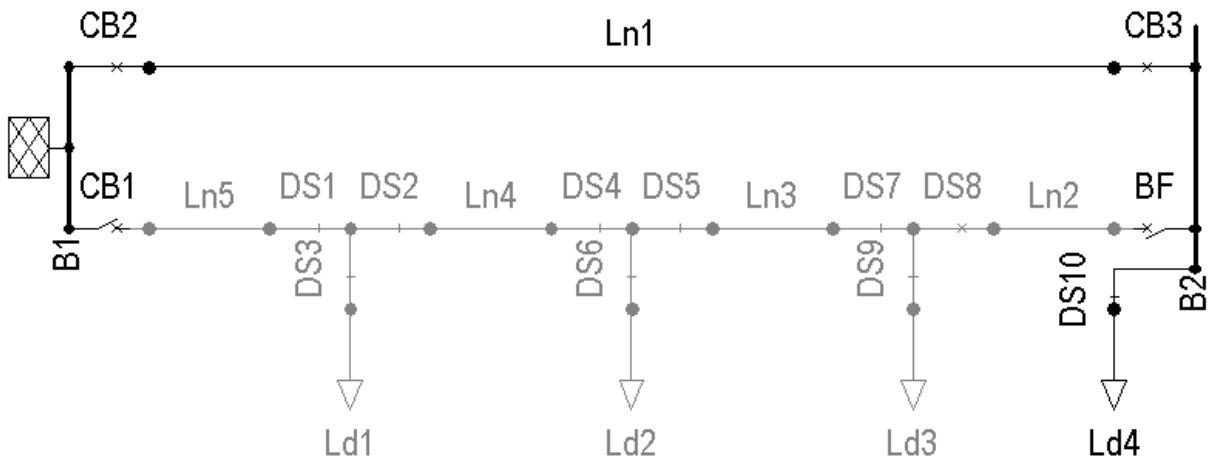


Figure 47.1.2: Protected Area

### Fault Isolation

Figure 47.1.3 shows the example network with the separation switches, 'DS2' and 'DS4' open. The separated area now only contains the faulted line, Ln4. There are now two restorable areas following the fault separation; the area which contains load 1, and the area which contains loads 2 and 3.

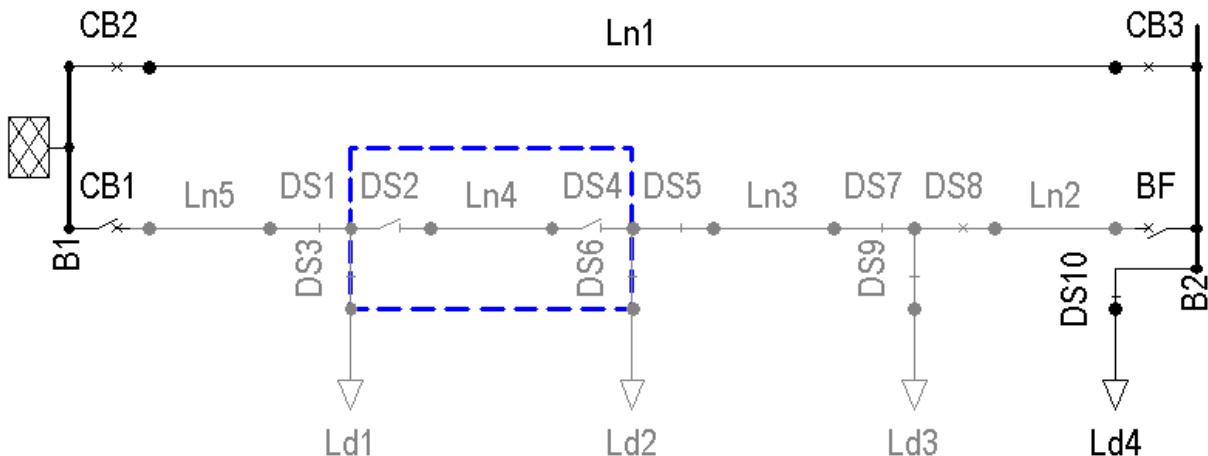


Figure 47.1.3: Separated Area Highlighted

### Power Restoration

After the fault separation phase is complete, the following switch actions are required to restore power to the two separate 'restorable' areas:

- Separation switch 'DS2' is 'remote-controlled' and has a switching time of 3 minutes.
- Power to load 1 is restored by (re)closing the protection breaker, 'CB1' which is also remote controlled.
- Load 1 is therefore restored in 3 minutes (=0.05 hours).
- Power to load 2 and 3 is restored by closing the back-feed switch, 'BF'.
- Because the back-feed switch has a actuation time of 30 minutes, loads 2 and 3 are restored in 0.5 hours.

The network is now in the post-fault condition as illustrated in Figure 47.1.4.

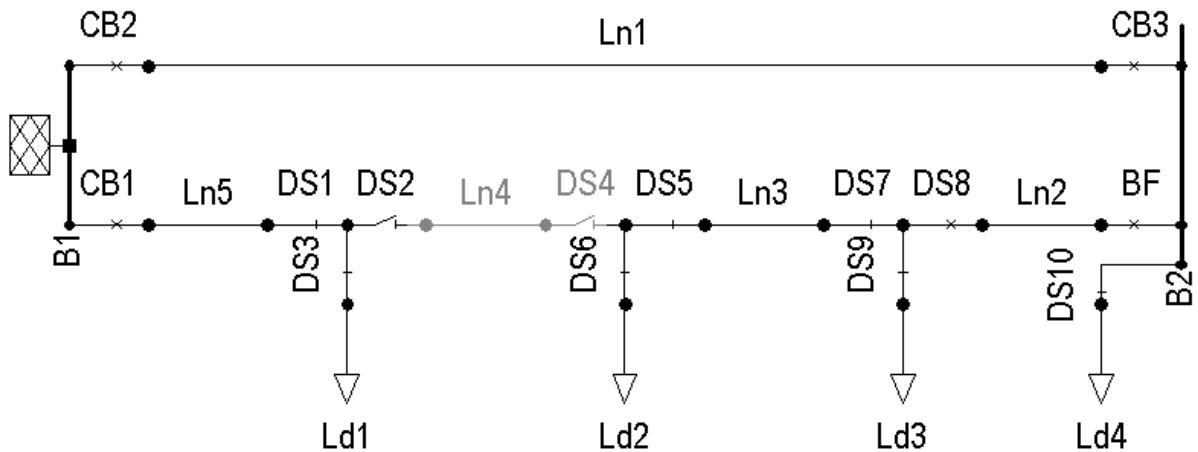


Figure 47.1.4: Power Restoration by Back-Feed Switch BF1 and CB1

#### **Overload Alleviation and Load Shedding**

Figure 47.1.5 shows a line overload in the post-fault condition in the example network: line 'Ln1' is loaded to 113%.

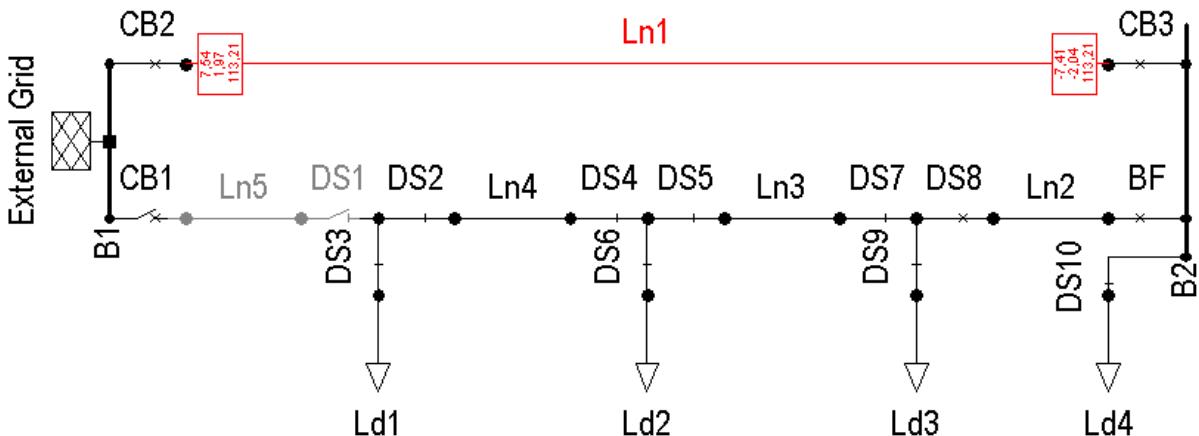


Figure 47.1.5: Overloaded Post-Fault Condition

In this example, loads 1, 2, 3 and 4 all contribute to the line overload on LN1, and consequently load would be shed based on load shedding options and priorities set by the user to alleviate the constraint.

## 47.2 Animated Tracing of Individual Cases

After the Reliability Analysis has completed, it is possible to view the fault clearance, fault separation, power restoration and load shedding actions completed by the algorithm for each contingency. To do this:

1. Click the *Fault Trace* button on the Optimal Power Restoration toolbar. A list of available contingencies will appear in a new window.
2. Select the contingency to consider and click OK. The network will be initialised to the state before the inception of the fault.
3. Click the *Next Step* button to advance to the next system state. This will usually show the system state immediately after the protection has operated and cleared the fault.

4. Click the *Next Step*  button to advance through more steps, each click advances one time step.
5. To stop the fault trace, click the *Stop Trace*  button.

## 47.3 Optimal RCS Placement

Following a Backbone Calculation (see Section 42.3), an Optimal Remote Controlled Switch (RCS) Placement can be performed to optimise placement of remote controlled switches within a feeder/s. The calculation optimises placement of a fixed number or optimal number of switches per feeder or backbone, with an objective function that minimises Energy Not Supplied (ENS), balances ENS, or minimises Expected Interruption Costs (EIC). The Optimal RCS Placement command is a heuristic planning tool, and may precede a detailed reliability analysis.

To conduct an Optimal RCS Placement, reliability data should be specified on the Reliability page of line elements (outages of other elements are not considered). See Chapter 46: Reliability Assessment, Section 46.3 for details.

If the cost of interrupted load is to be considered, a global Energy Tariff must be defined, see Chapter 18, Section 18.5.2: Defining Energy Tariffs for details.

The Optimal RCS command can be selected under Optimal Power Restoration toolbar, as shown on Figure 47.0.1 This section describes the Optimal RCS Placement objective function and command dialogs, and provides an example calculation.

---

**Note:** The Optimal RCS calculation requires that *feeder is supposed to be operated radially* be selected on the Feeder Basic Options page.

---

### 47.3.1 Basic Options Page

#### Calculate optimal RCS

Specify all Feeders or a user-defined set of Feeder/s for the Optimal RCS calculation. To show the Backbones to be considered by the calculation, select **Active Backbones**.

#### Objective Function:

The objective function of the Optimal RCS Placement command can be set to either:

- *Minimise ENS* by installing a specified number of RCS per feeder / backbone to minimise the Energy Not Supplied.
- *Balance ENS* by installing an optimal or fixed number of RCS per feeder / backbone to balance the Energy Not Supplied. This option may be used in some circumstances to plan the network in a way that considers connections with many (or large) customers and connections with few (or small) customers equitably.
- *Minimise EIC* by installing an optimal or fixed number of RCS per feeder / backbone to minimise the Expected Interruption Cost.
  - If this option is selected, a global *Energy Tariff* must be defined (see Chapter 18, Section 18.5.2: Defining Energy Tariffs).

#### Number of RCS:

- With an objective function to *Minimise ENS*, specify:
  - *Number of new RCS per feeder / backbone*.

- With an objective function to *Balance ENS* or *Minimise EIC*, select to either *Optimise number of RCS* or *Fix number of new RCS*.
  - Specify the Number or Maximum number of new RCS per feeder / backbone.
  - If the objective function is set to Minimise EIC, enter the Yearly costs per RCS in \$ per annum.

### Recording of results

- Select *calculate results only* to perform a calculation without making any modifications to the network.
- Select *save results in variations* to save the results to a Variation. Note that by default the variation will be inactive after running the Optimal RCS Placement.
- Select to *change existing network* to change the existing network. Note that this changes object data in the base network.

## 47.3.2 Output Page

### Results

A reference (pointer) to the results object.

### Report

(Optionally) select the format of results printed to the output window. The report provides details of the recommended remote controlled switches and their costs, and depending on the selected objective function, energy not supplied or interruption costs results.

## 47.3.3 Advanced Options Page

### RCS placement

Select to either determine *on selected backbones simultaneously* or *for selected backbones separately*.

- On selected backbones simultaneously (default):  
Positions for optimal RCS are existing switches in all the backbones of a feeder. This results in one set of optimal switches for the whole feeder.
- For selected backbones separately:  
Positions for optimal RCS are existing switches in one of the backbones of a feeder. This results in a set of optimal switches for each backbone of the feeder individually.

### Backbones for RCS placement

Select to either optimise RCS for all backbones, or only for backbones up to a specified order (in which case, define the maximum order). Note that if more than one backbone has been created for a feeder, the main backbone will have order “1”, the second “best” candidate has order “2”, and so on.

### Detailed output of results:

Optionally select detailed output mode to output additional details by “Section”, such as ENS, FOR, and EIC (depending on the optimisation option selected).

### Switching Time:

Set the *Time to actuate RCS* and *Time to actuate manual switches* (applied for all switches). These parameters are used by the calculation to determine ENS and EIC.

### Load flow calculation

Pointer to load-flow command (note for balanced calculations only).

### 47.3.4 Example Optimal RCS Calculation

Consider the simple example shown in Figure 47.3.1 where two feeders with three loads each are separated via three open points. Line outage rates and load parameters have been defined. To illustrate line Forced Outage Rates, from the main menu select *View → Diagram Colouring* (or select the *Diagram Colouring* icon). Under *3. Other* select *Primary Equipment → Forced Outage Rate*. In the example, there is a requirement to install a single Remote Controlled Switch (RCS) on each feeder to minimise the ENS.

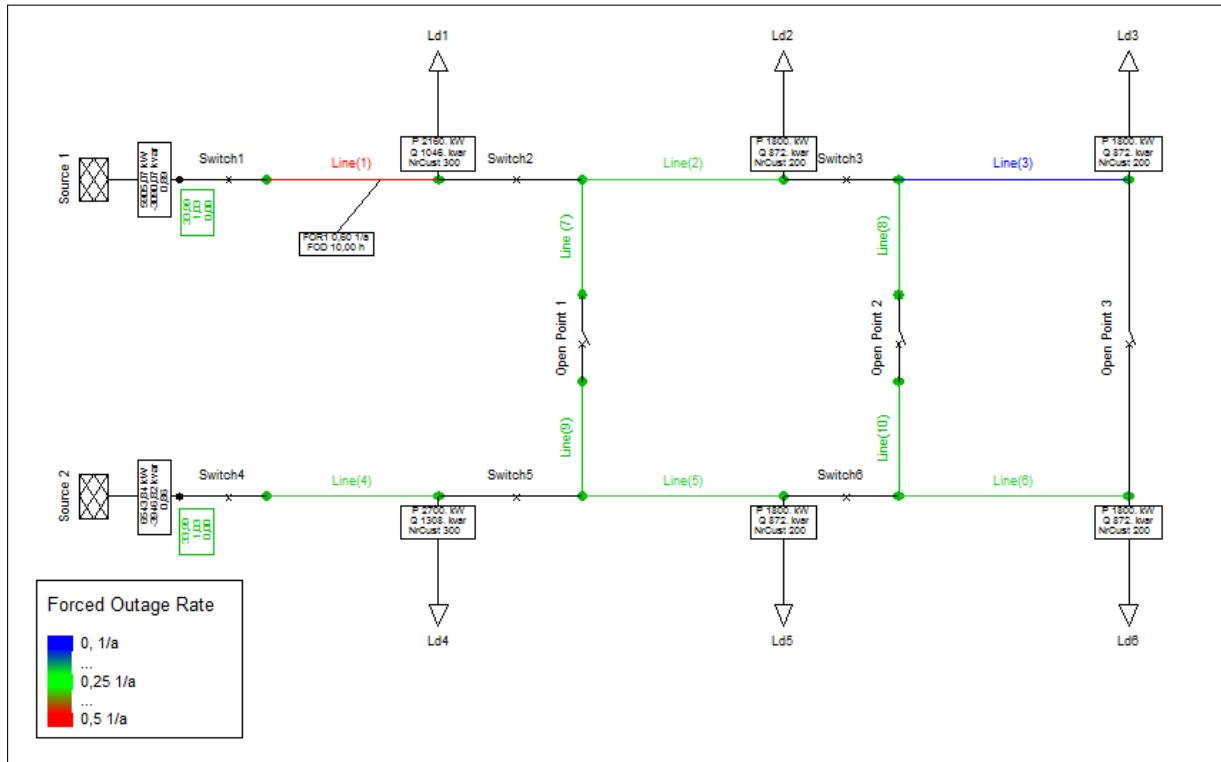


Figure 47.3.1: Example Optimal RCS Model

To calculate the optimal location(s) for remote controlled switches, a Backbone Calculation *for all feeders* based on *network structure* is first executed (see Section 42.3 for details of how to run the Backbone Calculation).

Next, an Optimal RCS calculation is executed *for all feeders*, with an objective function to *Minimise ENS*, limited to 1 RCS *per backbone*. Note that the calculation will run twice in this example (once for each feeder), and so two RCS's will be recommended.

The calculation simulates outages of each line, and calculates the ENS for placement of RCS's at each location. In order to mitigate the impact of outages (in particular, from the “problem line” Line(1)) the calculation recommends installation of remote controlled switches at locations “Switch2” and “Switch5” to minimise the ENS.

## 47.4 Optimal Manual Restoration

The Optimal Manual Restoration (OMR) command (*ComOmr*) can be found under the Optimal Power Restoration toolbar (click on the *Change Toolbox* icon (▼) of the main toolbar). The OMR command dialog is shown by clicking on the *Optimal Manual Restoration* icon (↙). The OMR calculation determines the optimal sequence for operating manual switches when searching for location of a fault

in a distribution network. This tool is intended for distribution networks with a radial feeder topology which may contain remote controlled switches (RCS). The Optimal Manual Restoration tool defines the locations of manual switches which are to be opened/closed and the corresponding sequential order that a service team should open/close these switches in order to restore power safely to the greatest number of consumers in the shortest possible time. The sequential order is defined by OMR levels: level 1 corresponds to the first step in the OMR process, level 2 corresponds to the second step and finally level 3 to the last one.

In this section the term switch refers to a coupler element *ElmCoup* or a switch element *StaSwitch*. The concept of feeder pockets is used in the calculation. A pocket represents an enclosed area of the radial network delimited by a remote controlled switch, open manual switches or a calculated OMR terminal. The OMR calculation determines one OMR terminal per level for each pocket. All manually closed switches connected to the OMR terminal are considered to have the same OMR level equivalent to the level for which the OMR terminal has been assigned. Up to three OMR levels can be calculated i.e. Level 1, Level 2 and Level 3. Level 1 pockets are areas enclosed by remote controlled switches and open manual switches. Level II pockets are areas enclosed by remote controlled switches, open manual switches and OMR level I switches. Similarly, Level 3 pockets are areas enclosed by remote controlled switches, open manual switches and OMR switches of level 1 and 2.

#### 47.4.1 OMR Calculation Prerequisites

The following network configuration conditions are required by the Optimal Manual Restoration calculation:

- A balanced *Load Flow* calculation must be available.
- The network must contain at least one defined feeder element *ElmFeeder*.
- Only radial networks will be processed. The option “Feeder is supposed to be operated radially” available in the feeder’s Basic Data page must be selected for the relevant feeders.
- It is recommended that a Backbone calculation is first performed (see Section 42.3).
- There must be at least one remote controlled switch in the network.
- It is recommended to build the network using terminals or secondary substation layouts (*ElmTrfstat*).

#### 47.4.2 Basic Options Page

##### Determine ‘OMR’ for

In this field the user must specify either *All Feeders* or *Selected Feeders*. If *Selected Feeders* option is chosen then a user-defined set (*SetSelect*) of feeders can be defined for the OMR calculation.

##### Max. Number of ‘OMR’ Levels

The maximum number of *OMR* levels can be set in this field with values between 1 and 3. All OMR levels higher than this setting will not be calculated.

##### Min. Power in Pocket

The minimum consumption (sum of all load elements within a pocket) below which a delimited area will not be considered as a pocket for the purposes of the calculation. This value applies to all OMR levels.

##### Backbone Order (Max.)

If a number of network backbones exist (e.g. following a *Backbone* calculation), the *Backbone Order (Max.)* option defines the number of backbones to be considered for calculation (ordered according to parameter *e:cBbOrder* of the backbone element *ElmBbone*). The elements contained within a backbones of an order higher than this value will be considered as part of a non-backbone branch.

**Show Backbones button**

The button **Show Backbones** provides access to the calculation relevant backbones. The *Backbone Order (Max.)* option must be higher than or equal to 1 in order to for at least one calculation relevant backbone to be shown.

**Show Output**

The Show Output checkbox enables the display of a calculation report in the output window.

### 47.4.3 Advanced Options Page

**Penalty Factor**

Penalty factors for switches depend on branch type and the level for which the OMR is being calculated. Two settings are available for introducing penalty factors: *Branches end at Manual Switch* (default value: 20%) and *Non-Backbone Branches (Level 1)* (default value: 25%). The default values are referred to below to illustrate their practical usage. Penalty factors are used differently depending on the OMR level being calculated:

- OMR level 1:
  - Switches located in backbone branches which end only with an RCS - no penalty factor is applied, weighting factor is 1.0.
  - Switches located in backbone branches which end only with a manual switch - 20% penalty factor is applied, weighting factor is 0.8.
  - Switches located in non-backbone branches which end only with an RCS - 25% penalty factor is applied, weighting factor is 0.75.
  - Switches located in non-backbone branches which end only with an open manual switch - 20% and 25% penalty factors are applied resulting to a weighting factor of 0.6.
  - Switches located in non-backbone branches which end with an open RCS and an open manual switch - 25% penalty factor is applied, weighting factor is 0.75.
- OMR level 2 and 3:
  - Switches located in backbone branches which end with an open RCS - No penalty is applied, weighting factor is 1.0.
  - Switches located in backbone branches which end with an open manual switch - 20% penalty factor is applied, weighting factor is 0.8.
  - Switches located in non-backbone branches which end with an open RCS - no penalty is applied, weighting factor is 1.0.
  - Switches located in non-backbone branches which end with an open RCS and an open manual switch - no penalty is applied, weighting factor is 1.0.
  - Switches located in non-backbone branches which end with an open manual switch - 20% penalty factor is applied, weighting factor is 0.8.

The term “network branches” is used for applying penalty factors. Branches are network paths starting from the feeder’s starting terminal and ending at a final downstream element (a radial topology is always assumed). For this purpose, branches are categorised according to the following criteria:

- Branches that end with an open manual switch that cannot be activated (parameter e:iResDir of the switch element is set to “Do not use for power restoration”): Inaccessible (geographical limitation, old technology etc...). These branches are not used in the OMR calculation.
- Branches that end with an open manual switch that can be activated. For these branches the manual restoration from the same feeder applies.
- Branches that end with a load element (does not lead to an open switch). These branches are not used in the OMR calculation.
- Branches that end with an open remote controlled switch that cannot be activated. These types of branches are not considered to lead to an open manual switch.

- Branches that end with an open remote controlled switch that can be activated. For these branches the remote control restoration from same feeder applies.
- Branches that end (within selected backbones) with an open remote controlled switch that can be activated. These branches are considered as a tie open point restoration from another feeder.

### Load Flow

A link to the *Load Flow* calculation settings is available by clicking on the blue arrow pointing to the right of the *Load Flow* field. The balanced *Load Flow* calculation type is automatically chosen (Unbalanced and DC *Load Flow* options are not supported).

#### 47.4.4 Definition of the objective function

The aim of the OMR calculation is to minimise the following objective function:

$$\Delta^x = |P_{upReg}^x \cdot F_{upReg}^x - P_{downReg}^x \cdot F_{downReg}^x| \quad (47.1)$$

The members of the above objective function are defined based on the following equalities:

$$P_{upReg}^x = P_{up}^x - P_{up}^{startReg} - \sum P_{down}^{upNStart} \quad (47.2)$$

$$F_{upReg}^x = F_{up}^x - F_{up}^{startReg} - \sum F_{down}^{upNStart} \quad (47.3)$$

$$P_{downReg}^x = P_{down}^x - \sum P_{down}^{downNStart} \quad (47.4)$$

$$F_{downReg}^x = F_{down}^x - \sum F_{down}^{downNStart} \quad (47.5)$$

$$(47.6)$$

where:

- $x$  is the terminal of the calculated pocket,
- $P_{upReg}^x$  is the upstream active power at terminal  $x$  with reference to the corresponding pocket,
- $P_{downReg}^x$  is the downstream active power at terminal  $x$  with reference to the corresponding pocket,
- $F_{upReg}^x$  is the upstream forced outage rate (FOR) at terminal  $x$  with reference to the corresponding pocket,
- $F_{downReg}^x$  is the downstream forced outage rate (FOR) at terminal  $x$  with reference to the corresponding pocket,
- $P_{up}^x$  is the upstream active power at terminal  $x$  with reference to corresponding feeder,
- $P_{up}^{startReg}$  is the upstream active power at the corresponding pocket starting element with reference to feeder,
- $P_{down}^{upNStart}$  is the downstream active power of neighbouring pocket's (upstream with respect to terminal  $x$ ) starting element with reference to feeder,
- $F_{up}^x$  is the upstream FOR at terminal  $x$  with reference to corresponding feeder,
- $F_{up}^{startReg}$  is the upstream FOR at corresponding pocket's starting element with reference to feeder,
- $F_{down}^{upNStart}$  is the downstream FOR of neighbour pocket's (upstream with respect to terminal  $x$ ) starting element with reference to feeder,
- $P_{down}^x$  is the downstream active power at terminal  $x$  with reference to corresponding feeder,
- $P_{down}^{downNStart}$  is the downstream active power of neighbour pocket's (downstream with respect to terminal  $x$ ) starting element with reference to feeder,
- $F_{down}^x$  is the downstream FOR at terminal  $x$  with reference to corresponding feeder and

- $F_{down}^{NStart}$  is the downstream FOR of neighbour pockets (downstream with respect to terminal  $x$ ) starting element with reference to feeder.

A manual switch is considered as being an OMR switch of a certain level if its associated terminal  $\Delta^x$  objective function is minimum compared with the objective functions of the other terminals within the calculated pocket.

#### 47.4.5 Example of an Optimal Manual Restoration Calculation

An example of the use of the Optimal Manual Restoration tool is shown here. Consider the MV distribution network (20 kV) as displayed in Figure 47.4.1. Five feeders are defined, one main feeder (*Feeder A*) supplies power in normal operation to the displayed network. *Feeder A* is radially operated and containing a number of normally open switches. Several remotely controlled switches are also defined and their associated substation is marked with a green circle.

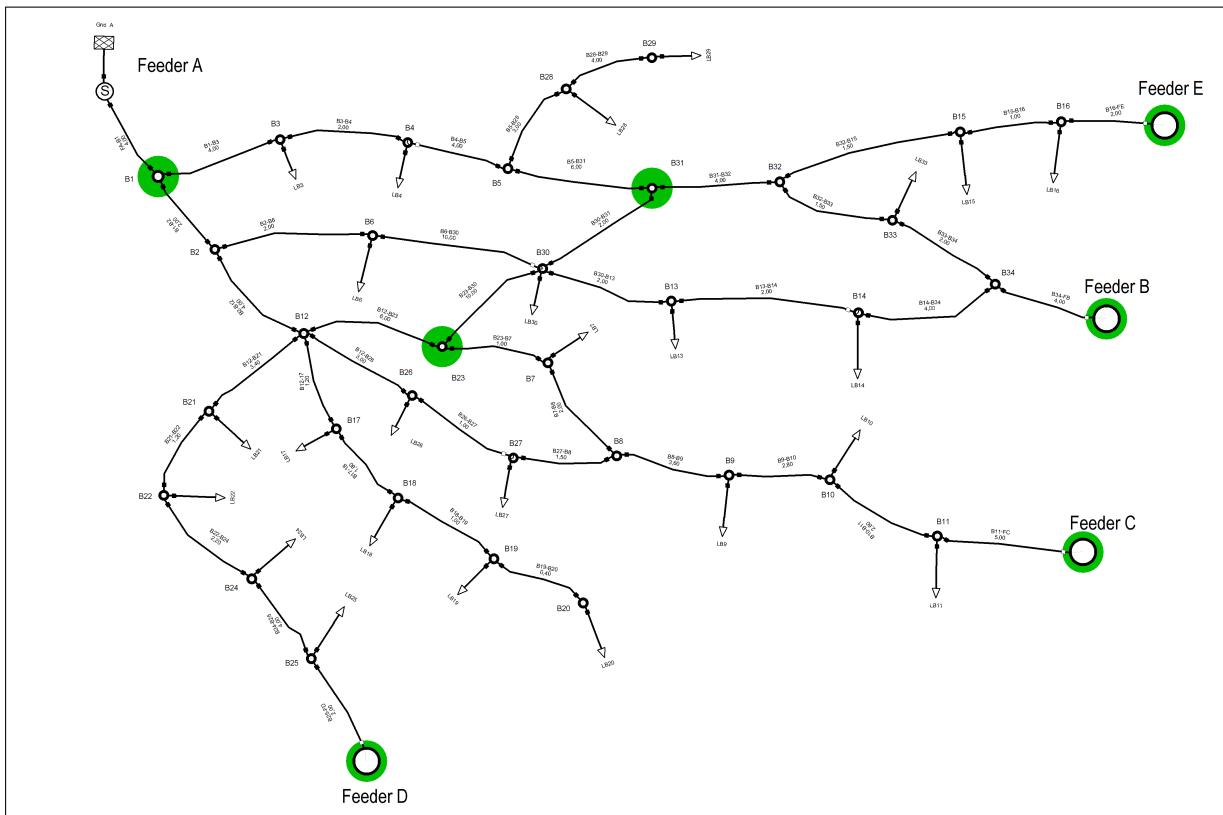


Figure 47.4.1: Generic MV Distribution Network

A substation layout similar to the one shown in Figure 47.4.2 is used for all substations.

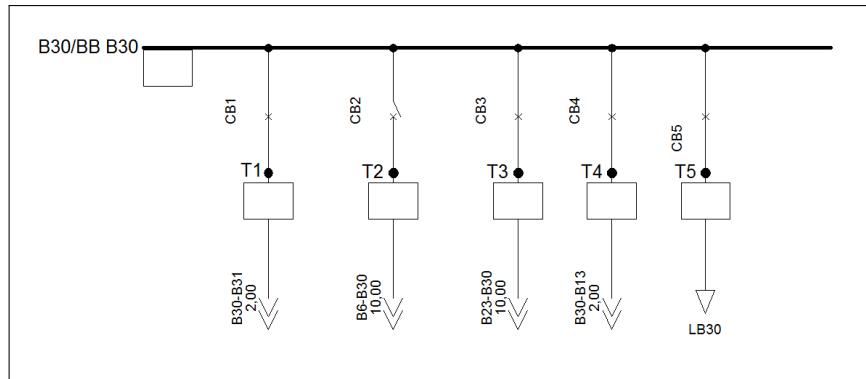


Figure 47.4.2: Generic Substation Single Line Diagram

A backbone calculation (*ComBbone*) for *Feeder A* is performed on this network based on *path load* (see Section 42.3 for details of how to run the Backbone Calculation), thus obtaining four backbones (from main *Feeder A* to the other four).

Using the backbone information an OMR calculation may be performed with reference to main *Feeder A*. The OMR calculation automatically updates the single line diagram with specific colours for the different OMR levels for each switch and associated substation as in Figure 47.4.3.

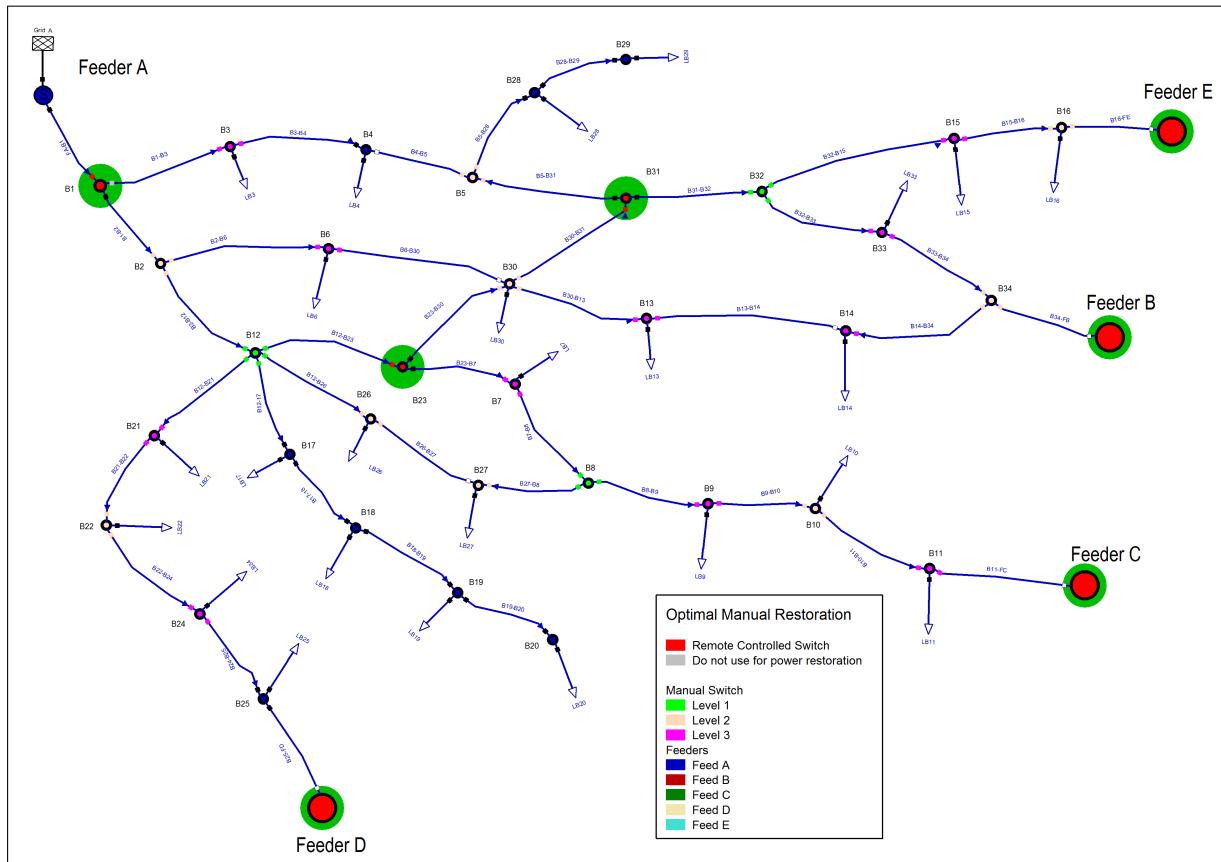


Figure 47.4.3: OMR Calculation Results Shown in the Single Line Diagram using Different Colours

If the **Show Output** checkbox is enabled in the Basic Data page of the OMR command dialog then a list of all the switches and their associated OMR level will be printed to the output window.

## 47.5 Optimal Recloser Placement

Circuit breakers equipped with protection relays including automatic reclosure (ARC) functionality are designed to handle fast and repetitive open-close duty cycles. These switching devices, which are referred to as Reclosers, allow minimisation of the duration of outages - especially in the case of faults of a transient nature -, leading to an increased availability of power supply to customers (reliability). The “Optimal Recloser Placement”  function, located in the “Optimal Power Restoration” toolbox, determines the best candidate locations for reclosers, leading to the highest improvement in the target reliability index.

Optimal Recloser Placement is based on the Reliability Assessment calculation function, evaluating the contributions of circuit breakers to the target reliability index for faults within their protecting area. The proposed recloser locations are highlighted in the network diagrams with red circles, reported in the output window and may be directly configured in the model.

The following sections provide a short technical background followed by explanations of the command dialog pages and its parameters.

### 47.5.1 Technical Background

An electrical network is subject to different types of faults that reduce the reliability of the power supply to customers. Depending on the duration of an interruption, the reliability indices evaluated in the reliability assessment account for either a sustained or momentary loss of supply. Even if the reasons for the faults are manifold, the faults themselves can be classified into sustained and transient faults. Independent from the fault classification, the fault clearance separates the faulted equipment by disconnecting the protected area from the supply. Circuit breakers with reclosing functionality are able to restore the supply by rapidly reclosing their contacts after having cleared a fault. For transient faults, where the loss of supply leads to the extinction of an electric arc caused for example by a lightning strike on an overhead line (one of the main causes for transient faults), the insulation medium “heals” during the disconnection period and allows an immediate restoration of supply for the reclosing attempt. The recloser logic attempts the reclosing a few times in case the fault is still present on the protected area. Each failed attempt will lead to a momentary interruption and is considered in the MAIFI calculation. This protection mechanism decreases the duration of customer interruptions as transient faults only cause momentary instead of sustained interruptions to the power supply. For sustained faults on the other hand, the reclosing attempts increase the number of momentary interruptions, based on the number of failed attempts.

Due to the high share of transient faults in the overall fault rate for overhead lines, and the high success rate of immediate power restoration with the reclosing attempts, reclosers are very popular and used from distribution up to transmission networks for reliability improvement. In *PowerFactory* reclosers can be considered for protection-related functions as well as for the reliability assessment.

#### Configuration of the breaker reclosing function

The Reliability Assessment calculation function supports two approaches regarding the modelling of reclosers:

- simplified: ticked checkbox “Consider as switch with automatic reclosing device” on the Reliability page of the breaker (*ElmCoup* and *StaSwitch*)
- detailed: relay including a reclosing block assigned to the breaker

## 47.5.2 Basic Options

### 47.5.2.1 Candidate locations

The settings for candidate locations define which breakers will be considered as potential reclosers in the optimal placement. Since the reclosing actions are preceded by a fault clearance, only circuit breakers will be considered as candidates for the recloser placement. Two options are available:

- **Circuit breakers:** all breakers configured with the switch type “Circuit-Breaker” or “Disconnecting Circuit-Breaker” on the Basic Data page are addressed by this selection
- **Switches with protection device:** all circuit breakers with an explicitly configured or modelled protection device are addressed by this selection. The configuration of the protection device can be as follows:
  - protection relay (*ElmRelay*) explicitly modelled and assigned to the breaker
  - ticked checkbox “Consider as switch with protection device” on the Reliability page of the breaker dialog

In addition to the general differentiation of breakers, two further options are available to restrict the number of potential locations to realistic ones.

**Exclude candidates that protect cables:** since the reclosing actions address transient faults, which occur only on overhead lines, breakers protecting cables or cable sections can be excluded by ticking the checkbox. For protected areas with a mixture of overhead lines and cables, special attention is needed for the thermal rating of the cables and the recloser configuration, since each failed reclosing attempt stresses the cable sections thermally with almost no time to cool down in between.

**Exclude specific candidates:** When ticked, an option to select either single breakers or a selection object (*SetSelect*) containing references to multiple breakers, is available. The corresponding breakers will be ignored as potential reclosers in the optimisation.

### 47.5.2.2 Objective function

As objective function for the optimisation algorithm, the minimisation of the following reliability indices is available:

- ENS: Energy Not Supplied
- SAIDI: System Average Interruption Duration Index
- EIC: Energy Interruption Costs

For the first two indices, the stochastic model including the failure rates and repair times are sufficient for an optimisation. For the minimisation of the EIC, interruption costs have to be configured (globally or locally) in the Reliability Assessment command.

### 47.5.2.3 Reliability assessment

Optimal Recloser Placement is based on the Reliability Assessment command, which is linked in the dialog. The flag for “Enhanced consideration of automatic reclosing devices” on the Advanced Options page is a mandatory requirement for Optimal Recloser Placement and is set therefore without the possibility to untick it. In addition, the selection of candidate locations, being replicated in the Optimal Recloser Placement command, is greyed out in the referenced Reliability command.

When Optimal Recloser Placement is called for the first time in the study case, an existing Reliability command including all its configurations is copied as subobject to the Optimal Recloser Placement (and if none is available, a new Reliability command object is created with default settings). Any change

done after that to the original Reliability command directly within the study case will not be replicated and considered by the command linked in the Optimal Recloser Placement command.

#### 47.5.2.4 Settings for placement

**Maximum number of reclosers to place:** Defines how many candidate locations will be selected at most as potential reclosers.

**Maximum number of reclosers per feeder:** If the reliability assessment covers a network with multiple feeders, the Optimal Recloser Placement calculation will limit the reclosers per feeder to the entered value. The value should be less than or equal to the Maximum number of reclosers to place.

**Maximum number of reclosing attempts:** The number of attempts is used for the evaluation of the MAIFI considering the proposed reclosers. If the network is changed, the number will be replicated into the breakers' corresponding settings.

#### 47.5.2.5 Recording of results

If the optimisation finds one or more locations which lead to an improved target reliability index, the result is treated according to one of the following options:

- **Calculate results only:** The optimisation reports the proposed reclosers, but does not change the relevant breaker settings.
- **Change existing network:** The optimisation changes the breaker settings of proposed recloser locations by ticking the settings “Consider as switch with automatic reclosing device” as well as “Consider as switch with protection device”. The number of attempts from the Optimal Recloser Placement command is set for the breakers, too. The change will be performed on the base network or on the recording expansion stage of an active variation.
- **Save results in variation:** The changes to the proposed recloser locations will be recorded by an automatically created expansion stage and corresponding variation, for which the name can be entered in the input field “Variation name”.

### 47.5.3 Results

On the Results page of the Optimal Recloser Placement command, the result object (*ElmRes*) is linked. It contains the values for the contributions from each candidate location on the main reliability indices with and without reclosing ability. The results are used by the Optimal Recloser Placement Reports (see Section 47.5.4).

**Show reports after calculation:** If ticked, the linked Optimal Recloser Placement Reports command is executed automatically, taking into account the configured settings (see Section 47.5.4).

**Show output of Reliability Assessment:** If ticked, the relevant output generated by the linked Reliability command from the Basic Options page is reported in the output window. Otherwise it is suppressed.

### 47.5.4 Optimal Recloser Placement Reports

The Optimal Recloser Placement Reports can be accessed using the icon  to the right of the Optimal Recloser Placement icon.

On the General tab, the result object (*ElmRes*) is linked, which has been filled by a preceding execution of the Optimal Recloser Placement command. As reports, at least one of the following has to be

selected:

**Reliability indices:** The report lists for the main reliability indices (ENS, EIC, SAIDI and MAIFI) the values before and after the placement of the reclosers as well as the differences between the two cases. If not selected as target index for the optimisation, the EIC may be zero if no costs were configured for the reliability assessment.

**Contributions of switches:** The report lists for all candidate locations the impact onto the main reliability indices (ENS, EIC, SAIDI and MAIFI). A negative value corresponds to an improvement, a positive value an increase of the corresponding index. The latter is only the case for the MAIFI, which is increased because of the increase of momentary interruptions due to failing attempts on sustained faults and due to successful reclosing actions (a transient fault results in only a momentary interruption instead of a sustained one, as it would be the case without the reclosing function).

In addition, the report provides some further information about the breakers, such as the Substation or Site and Feeder (if relevant).

# Chapter 48

# Contingency Restoration Analysis

## 48.1 Introduction

Generally, the term “contingency analysis” is essentially referring to the analysis of abnormal system conditions. Here, a contingency analysis is considered as the process of evaluating the network states like loading and voltages resulting from faults in the network. For fully meshed grids with circuit breakers at each station and station-specific remedial action schemes, the *Contingency Analysis*, explained in Chapter 27, can be used to estimate these values. For example in distribution networks, however, the network is often not or not fully meshed and circuit breakers are only available at certain locations, so that faults usually lead to unsupplied customers. Depending on the fault location, additional switching operations are therefore required in the network in order to resupply the largest possible number of customers without violating network constraints.

With the *Contingency Restoration Analysis*, explained in this chapter, it is possible to consider these cases as this calculation combines the handling of contingencies from the *Contingency Analysis* with the power restoration processes of the *Optimal Power Restoration* (see Chapter 47). The structure of this chapter is as follows:

- Section 48.2 gives a short overview of the technical background.
- Section 48.3 explains the *Contingency Restoration Analysis* toolbar and describes briefly what each button does.
- Section 48.4 goes into detail about the various options of the *Contingency Restoration Analysis* command and explains how switches can be configured.
- Section 48.5 explains how to manage the variables to be recorded.
- Section 48.6 shows the trace functionality.
- Section 48.7 looks at the standard colouring and reporting options available once the analysis has been executed.

## 48.2 Technical Background

A contingency analysis involves simulating and evaluating the effects of predefined contingencies, such as faults, on the network’s operational state. In fully meshed grids, such as transmission grids, a single contingency should not lead to the interruption of the power supply as the n-1 criterion is typically applied. However, in a grid that is not fully meshed, and where circuit breakers are only available at certain locations, the tripping of circuit breakers during the fault clearance can lead to the interruption

of customers. An additional restoration process is therefore necessary to re-establish power supply to as many customers as possible after a fault, while maintaining operational limits.

The power restoration process begins by isolating the fault. The feasibility of resupplying customers then depends on the network topology (meshed, open-ring, radial, ...) and additional constraints, such as thermal loading and voltages, that can be considered. The aim is to find the best set of switching operations to restore the power of all customers. A simple example for this process is shown in Figure 48.2.1.

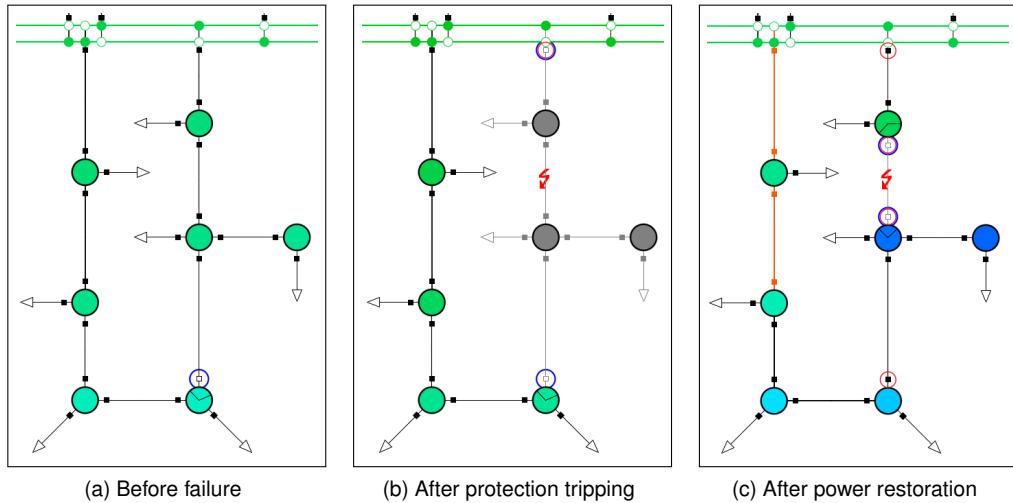


Figure 48.2.1: Complete power restoration in an open-ring structure after a fault

If constraints are taken into account, one result of the investigation can also be to identify the loads that cannot be resupplied, e.g. to avoid the overloading of elements as can be seen in Figure 48.2.2.

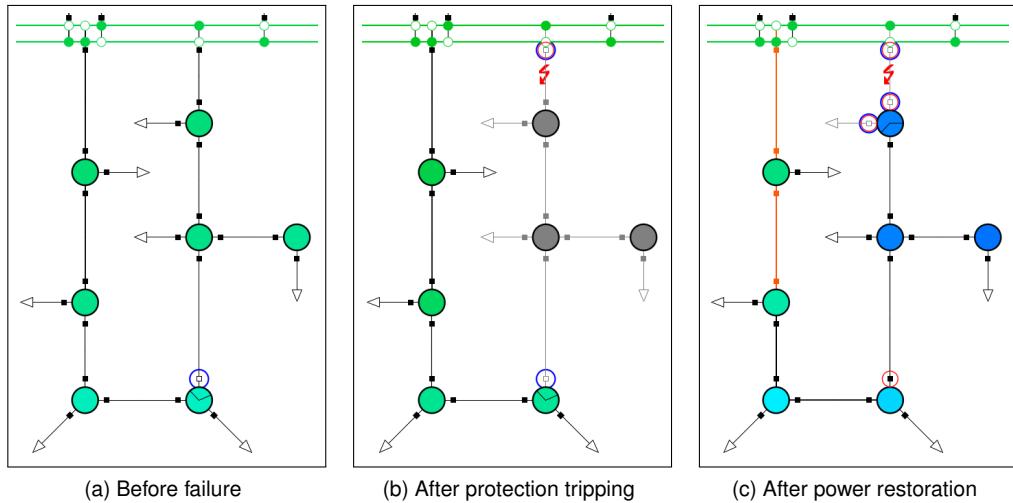


Figure 48.2.2: Power restoration with incomplete resupply to comply with constraints

## 48.3 Contingency Restoration Analysis Toolbar

To access the *Contingency Restoration Analysis* related functions within *PowerFactory*, click on the icon *Change Toolbox* (▼) and select *Contingency Restoration Analysis*. The figure below shows the

functions available on the *Contingency Restoration Analysis* Toolbar.

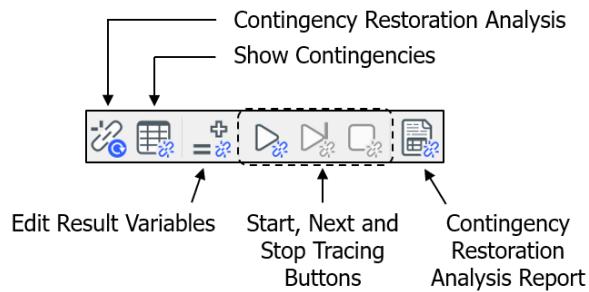


Figure 48.3.1: Contingency Restoration Analysis Toolbar Functions

### 48.3.1 Contingency Restoration Analysis Command

The *ComCntrst* command ( ) can be configured as required and executed. All the options are described in Section 48.4.

### 48.3.2 Show Contingencies

The *Show Contingencies* ( ) button can be used to look at the current selection of contingencies. Contingencies can either be automatically created with the *Contingency Restoration Analysis* command or manually before the execution of the command.

### 48.3.3 Edit Results Variables

For each element class relevant to the *Contingency Restoration Analysis*, there is a default set of result variables which are recorded in the results file (see Section 48.5). Using the *Edit Results Variables* ( ) button, the user can specify additional variables and get easy access to the variable definitions.

### 48.3.4 Tracing Buttons

To visualise the changing system state during the fault and the restoration process, a trace function is available. See Section 48.6 for details.

### 48.3.5 Contingency Restoration Analysis Reports

A set of built-in reporting scripts is available using the *Contingency Restoration Analysis Reports* ( ) button. These are described in Section 48.7.

## 48.4 Command Dialog, Options and Switch Configuration

The *Contingency Restoration Analysis* uses feeders (*ElmFeeder*) to calculate the best restoration strategy. It can therefore only be used in grids where feeders are defined. More information about feeders can be found in Section 15.6.

### 48.4.1 Basic Options page

#### Calculation

This section contains a link to the *Load Flow* command that will be used to calculate the *Contingency Restoration Analysis*.

#### Contingencies

The command will execute individual contingency objects which are called *ComContingency* and are stored in the *Contingency Restoration Analysis* command. They define a network element (or several elements) that is faulted and leads to the potential customer outage. It is possible to either create the contingency objects automatically or predefine them by manually creating *ComContingency* objects in the *Contingency Restoration Analysis* command before the execution.

- *Automatically at calculation start*: The contingency elements are automatically created and stored in the command according to the selection in *Fault Locations* on the same page. The created contingencies will be considered for the analysis. Other contingencies already existing in the command are set to “Not analysed”.
- *Manually before the calculation*: All contingencies that are already existing in the command and that are not ignored with their “Not analysed” flag will be considered for the analysis. The **Show** button can be used to display a list of all contingencies that are stored in the command. The contingencies that are *Not analysed* are also displayed in this list and can be activated if required. With the **Reconfigure** button, it is possible to automatically configure the contingencies as with the selection *Automatically at calculation start*, but before the command is executed.

#### Fault Locations

- *Whole System*: Contingencies are generated for all relevant elements in the whole network.
- *User Defined*: Contingencies are only generated for all relevant elements that are located within the selected element (e.g. a grid or a feeder) or that are part of a chosen selection.

Contingencies will only be created for the elements classes that are selected:

- *Busbars / terminals*: All *ElmTerm* objects with the *Usage* “Busbar” or “Junction Node”.
- *Lines / cables*: All lines and cables (*ElmLne* objects).
- *Transformers*: All transformers (*ElmTr2*, *ElmTr3*, *ElmTr4*, *ElmTrmult*, *ElmVoltreg*).
- *Shunts/Filters/Ser. Impedances*: All parallel and series elements.

#### Additional Fault Cases

It is possible to select additional, predefined fault cases (*IntEvt*) and fault groups (*IntFaultgrp*) to be considered in the analysis. These have to be stored in the *Fault Cases* (*IntFltcases*) or *Fault Groups* folder (*IntFltgroups*) within the project’s operational library. A *ComContingency* object is automatically created for each selected fault case. If a fault group is selected, all contained fault cases will be considered as separate contingencies.

Using additional fault cases, it is also possible to consider multiple faults (e.g. n-2 cases).

## 48.4.2 Protection page

### Fault Clearance Breakers

- *Circuit breakers*: All switches in the system whose *aUsage* is set to “Circuit Breaker” can be used for fault clearance.
- *Switches with protection device*: All circuit breakers in the system which are controlled by a protection device (fuse or relay) can be used for fault clearance. Circuit breakers which are set to have a protection device are also considered.

## 48.4.3 Restoration page

### 48.4.3.1 Restoration tab

If *Automatic Power Restoration* is selected, the *Contingency Restoration Analysis* automatically determines switching operations to resupply as many customers as possible. Section 48.4.6 explains how to configure switches that are to be used for this purpose. To fine-tune the resupply the following options are available.

- **Main target**: It can be selected whether the constraints shall be kept during the power restoration or only serve as a limit value to indicate violations that occur:
  - *Maximise restored power*: The power of all customers who can be resupplied will be restored. Constraints are complied with wherever possible. However, if some customers cannot be resupplied while complying with the constraints, the resupply is carried out in violation of constraints. The reports and the colouring of results then highlight which faults lead to constraint violations.
  - *Comply with constraints*: Only the power of customers who can be resupplied in compliance with the constraints is restored. As many customers as possible are resupplied.
- **Tie Open Point Optimisation**: After the isolation of failures, parts of the network may be unsupplied. However, the network can be reconfigured by moving the tie open point in order to restore as much power as possible (partial power restoration). This reconfiguration might lead to violations of constraints (e.g. overloading), which should be avoided. The following power restoration modes are available:
  - *Disabled*: No movement of tie open points.
  - *Enabled without load transfer*: Tie open points can only be moved between the feeder affected by the fault and a directly-bordering feeder. Only unsupplied loads are then transferred to and restored by the neighbouring feeder. An exception applies in the event that constraints are violated after the fault isolation, e.g. due to the loss of generation. In this case, loads would have to be shed in order to comply with constraints. A load transfer can still be carried out for these loads, as they would otherwise remain unsupplied.
  - *Enabled with load transfer*: Tie open points can be moved between a bordering feeder and a second-level bordering feeder. In this case supplied loads not affected by the fault can also be transferred to other feeders to increase the flexibility of the restoration algorithm.
- **Busbar transfer**: Whenever a busbar in a multi-busbar substation with multiple transformers is unsupplied as a consequence of a fault, closing coupling breakers within the substation may be an option for restoration. This transfer of a busbar to the supply of another transformer can be executed in different ways:
  - *Simple reconnection*: An unsupplied busbar will be connected to the adjacent busbar. If constraint violations remain, further measures will be considered, e.g. load shedding within feeders.
  - *Optimised without additional meshes*: The algorithm will optimise the coupling breaker positions in the substations to supply as much as possible of the unsupplied feeders, taking the

constraints into account. The transfer of busbars not directly affected by the fault to other transformers is possible. The switching actions will not lead to additional meshes, e.g. the parallel operation of transformers due to the busbar transfer.

- *Optimised with and without meshes*: Same as “Optimised without additional meshes” but with the further freedom for the algorithm of creating new meshes, e.g. leading to parallel operation of transformers.
  - **Backward recovery**: Whenever a busbar, being the starting point of one or more feeders, is de-energised after a fault (e.g. of an HV/MV-transformer), closing tie open points in one of the feeders can restore the supply of the busbar and the other feeders from a neighbouring substation. This so-called backward recovery is available for networks with explicit substation modelling (*ElmSubstat*, *ElmTrfstat*). The algorithm finds the best feeder for resupplying the substation and the interrupted feeders. For a better understanding, see the example for a backward recovery in Figure 46.4.1 of Section 46 in the chapter *Reliability Assessment*.
- The *Backward recovery* offers the following options:
- *Do not allow*: Backward recovery will not be considered for restoration.
  - *Allow but prefer standard recovery*: Backward recovery will only be used if the other restoration options fail.
  - *Allow with user-defined preference*: Substations where backward recovery is allowed can be selected here.
  - *Allow and prefer*: Whenever possible, backward recovery will be preferred over the other restoration options.
- **Limit number of switch actions for restoration**: By activating this selection, a *Maximum number* of switching operations can be entered. The power restoration is stopped once the *Maximum number* has been reached, even if not all customers have been resupplied yet.

#### 48.4.3.2 Advanced tab

This selection is only shown if “Comply with constraints” is selected as *Main target* on the Restoration page.

##### Load Priorities

This setting is used to evaluate the element's priority value, entered by the user in the load elements.

- **Lowest priority number refers to most critical load**: Loads with highest priorities are resupplied last.
- **Highest priority number refers to most critical load**: Loads with lowest priorities are resupplied last.

#### 48.4.4 Constraints page

##### 48.4.4.1 Constraints tab

##### Constraints

This page allows the user to define the consideration and the limits for various constrained quantities. For each considered constraint, either the “global” definition of the limit on this page or the “individual” definition on the Reliability page of each component can be used. The following constraints can be considered:

- *Thermal loading constraints*

- *Voltage constraints*
- *Feeder voltage drop/rise constraints*

### Global limits

If a constraint is defined in such a ways that it uses “global” limits, these can be entered here:

- *Maximum thermal loading*
- *Lower and Upper limit of allowed voltage*
- *Maximum drop and rise of the feeder voltage*

#### 48.4.4.2 Advanced tab

##### Calculation if base case has constraint violations

It can happen that the load flow for the base case without the application of contingencies already leads to the violation of constraints for some components. In this case, one of the following two options can be selected.

- *Stop Calculation*: The calculation of contingencies won't be executed and an error message is displayed.
- *Relax constraints of initially overloaded elements*: For elements where initially the loading limit is exceeded, the constraints are relaxed. The calculated loading of the base case load flow (e.g. a loading of 85 %) is taken, increased by the value entered here (e.g. a value of 10 % would lead to a value of  $1.1 \cdot 85\% = 93.5\%$ ) and used as new constraint for the analysis.

##### Ignore all constraints for...

Constraints are ignored for all terminals and components below the entered voltage level.

- *Nominal voltage below or equal to*: The voltage level in kV is specified here if *Ignore all constraints for...* is enabled.

##### Consider Boundary Constraints outside feeders

If this option is set, the boundary constraints, applied on the boundaries *Contingency Restoration Analysis* page are considered during restoration.

#### 48.4.5 Results page

##### Results

This field provides a link to the Contingency Restoration Analysis results object (*ElmRes*) that stores the three result files that are used:

- **Load Flow**: Contains recorded Load Flow results at the end state of the restoration process for each recorded network component, selected variable and contingency.
- **Restoration**: Contains information about the restoration process like interrupted customers, restored power and switching actions.
- **Summary**: Contains the extreme values of voltage and loading for each network element and the contingency in which they occur.

A different results object can be selected if required.

## Recording Limits

These parameters set the global threshold used to determine whether a calculated result is recorded in the *Load Flow Results* object. Whenever one of the defined constraints is violated, the calculated result (for the corresponding contingency and network component) is recorded in the *Load Flow Results* file:

- **Thermal loadings above (%)**: Only loadings exceeding this value will be recorded.
- **Absolute voltage below (p.u.)**: Voltages lower than this value will be recorded.
- **Absolute voltage above (p.u.)**: Voltages higher than this value will be recorded.
- **Feeder voltage drop above (%)**: Voltage drops along a feeder higher than this value will be recorded.
- **Feeder voltage rise above (%)**: Voltage rises along a feeder higher than this value will be recorded.

---

**Note:** As the results for an element are only recorded for contingencies where the recording limits are exceeded, the *Load Flow Results* matrix is sparse. This must be considered when displaying results from the *ElmRes* object via scripts or in some of the predefined reports such as *Loading/Voltage violations per case*.

---

### 48.4.6 Configuring Switches for the Contingency Restoration Analysis

On the *Contingency Restoration Analysis* page of switches, it can be configured whether they can be opened or closed during the restoration.

- **Switch can be opened during restoration**: If activated, the switch can be opened during the restoration process (e.g. to isolate the faulted element).
- **Power Restoration**: The following selections are available to specify whether the switch can be closed during the power restoration.
  - *Independent of direction*: If this option is selected, the switch will be used to restore power flow regardless of the direction of the post restoration active power flow.
  - *Direction dependency of the restoration*:
    - \* For switches (StaSwitch) and breakers (ElmCoup) in bays of substations feeding branches:
      - *From Branch to Node*: If the branch side of the switch is supplied, it can be used to restore the unsupplied node side of the switch.
      - *From Node to Branch*: If the node side of the switch is supplied, it can be used to restore the unsupplied branch side of the switch.
    - \* For breakers (ElmCoup) connecting two terminals:
      - *From j to i*: If the connection side *j* of the breaker is supplied, it can be used to restore the unsupplied side *i* of the breaker.
      - *From i to j*: If the connection side *i* of the breaker is supplied, it can be used to restore the unsupplied side *j* of the breaker.
  - *Do not use for power restoration*: If this option is selected, the switch will not be used to restore power.

## 48.5 Managing variables to be recorded

In the Study Cases chapter, Section 13.11, there is a description of how results variables are managed, using the *ElmRes* object. For the *Contingency Restoration Analysis*, a minimum set of variables for

each element class is recorded, which is detailed in each relevant variable selection *IntMon* object, but it is possible to add additional variables to these *IntMon* objects if required.

The *Variable Selection* object is described in Section [19.3: Variable Selection](#).

---

**Note:** For elements included in the variable selection by default, additional variables are only recorded for those contingencies in which the recording limit of the element (e.g. the loading for *ElmLne*) is exceeded.

---

## 48.6 Trace Function

The trace functionality allows the user to visualise the changing system state during the fault and the restoration process.

A trace can be initiated using the *Start Trace* button (▶) on the *Contingency Restoration Analysis* toolbar. When this button is clicked, a dialog opens allowing the user to select a contingency. Following the selection of a contingency by the user and clicking on **OK**, the contingency dialog is closed and the base case load flow is executed. Clicking on the *Next Time Step* icon (▶) on the main toolbar for the first time will then execute the contingency and calculate the load flow after the tripping of the protection. A second click on *Next Time Step* (▶) shows the state at the end of the power restoration and also the load flow results for this state. A click on the *Stop Trace* icon (□) clears the calculation.

## 48.7 Result Analysis

### 48.7.1 Diagram Colouring

There are two different default colouring modes that can be used to display the results of the Contingency Restoration Analysis in a single-line diagram.

- **Contingencies: successful/unsuccessful restoration:** With this colouring it is shown whether a contingency at an element leads to a successful or unsuccessful restoration.
  - *Complete restoration within constraints:* It is possible to fully restore the power of all customers without violating any constraint.
  - *Incomplete restoration:* A complete restoration was not possible due to topology reasons (e.g. remote customers without an alternative supply path are still unsupplied or there are not enough usable switches).

With the *main target* “Comply with constraints” the last possible colouring is:

- *Incomplete restoration, limited by constraints:* It was not possible to resupply all customers as otherwise constraints would have been violated.

With the *main target* “Maximise restored power” two additional colourings may be used:

- *Complete restoration with constraint violations:* It is possible to fully restore the power of all customers, however, constraints are violated in doing so.
- *Incomplete restoration with constraint violations:* The restoration of as many customers as possible leads to constraint violations and is not complete due to topology reasons.

- **Voltages / Loading:** The elements are coloured according to their maximal loading (*maxLoading* of branches) or their minimum/maximum voltages (*min\_v* and *max\_v* of nodes) at the end of the restoration.

## 48.7.2 Predefined Reports

The built-in *Contingency Restoration Analysis* reports are presented in tabular format. They are accessed via the *Contingency Restoration Analysis Reports* button (Report icon) in the Contingency Restoration Analysis toolbar.

### 48.7.2.1 Basic Options page

On this page, the results object (*ElmRes*) of the Contingency Restoration Analysis that is to be used to create the reports can be selected. It also provides an overview of how many reports are selected and will be created when the report command is executed.

### 48.7.2.2 Loading page

#### Loading Reports

- **Worst loading violations:** Reports the largest loading violation for each component (if above the specified *loading threshold*), considering all contingencies. Any such component is reported only once, i.e. it is reported for the contingency causing this violation.
- **All loading violations:** All overloaded components (if above the specified *loading threshold*) for each contingency are displayed in a single list.
- **Loading violations per case:** All overloaded components (if above the specified *loading threshold*) for each contingency are displayed in separate lists (i.e. one list per contingency case).

#### Filters

- **Branches: report highest loading only:** This can reduce the amount of reporting when the network contains many branches consisting of multiple line elements.
- **Suppress contingency violation if base case is violated:** This is typically used to filter out overloads which are inherent in the network and are of little interest.
- **Loading threshold:** Loading value above which the loadings are displayed in the report.

The tabular reports offer the option to use the usual filtering and sorting options. With the *Loading Limit* it is possible to change the threshold value above which the loadings are displayed. The parameter *Overloading Limit*, on the other hand, defines the threshold value above which the displayed loading bars are coloured red. Within the table, references to objects can be used to edit the object itself or mark it in graphic.

### 48.7.2.3 Voltage page

#### Voltage Reports

- **Worst voltage violations, Maximum voltage:** Reports the largest voltage violation of a terminal (greater than or equal to the specified voltage threshold) considering all contingencies. Any such terminal is reported only once (i.e. it is reported for the contingency causing this violation).
- **Worst voltage violations, Minimum voltage:** Reports the largest voltage violation of a terminal (less than or equal to the specified voltage threshold) considering all contingencies. Any such terminal is reported only once (i.e. it is reported for the contingency causing this violation).
- **All voltage violations, Maximum voltage:** Reports all voltage violations of a terminal (greater than or equal to the specified upper voltage threshold) considering all contingencies.

- **All voltage violations, Minimum voltage:** Reports all voltage violations of a terminal (less than or equal to the specified lower voltage threshold) considering all contingencies.
- **Voltage violations per case:** All busbars with exceeding voltage (maximum or minimum) are displayed in separate lists.

### Filters

- **Suppress contingency violation if base case is violated:** This is typically used to filter out voltage violations which are inherent in the network and are of little interest.
- **Max. voltage threshold:** Voltage above which the terminal is displayed in the report.
- **Min. voltage threshold:** Voltage below which the terminal is displayed in the report.

With the *Max./Min. voltage threshold* it is possible to change the threshold value above or below which the voltages are displayed. The parameter *Max./Min. Voltage Limit*, on the other hand, defines the threshold value above or below which the displayed voltage bars are coloured red.

#### 48.7.2.4 Feeder page

##### Feeder Reports

- **All voltage drop violations:** Reports all voltage drop violations of a feeder (greater than or equal to the specified minimum voltage drop threshold) considering all contingencies.
- **All voltage rise violations:** Reports all voltage rise violations of a feeder (greater than or equal to the specified maximum voltage rise threshold) considering all contingencies.
- **Voltage drop and rise violations per case:** All feeders that are exceeding the voltage rise or drop thresholds are displayed in separate lists.

### Filters

- **Min. voltage drop:** Voltage drop along a feeder above which it is displayed in the report.
- **Min. voltage rise:** Voltage rise along a feeder above which it is displayed in the report.

With the *Reporting threshold* it is possible to change the threshold value above which the voltage drops or rises are displayed. The parameter *Critical voltage drop/rise*, on the other hand, defines the threshold value above which the displayed voltage drop/rise bars are coloured red.

#### 48.7.2.5 Summary page

- **Contingency Summary:** The summary shows an overview for each contingency. For contingencies which cause interrupted customers, the report also includes the number of customers interrupted, together with the total MW interruption. If there are customers that could not be resupplied due to constraints (only relevant with the main target *Comply with constraints*) or due to the topology, which makes a resupply impossible (e.g. remote customers without an alternative supply path), their number and active power is shown as well. The number of switching actions for the restoration is also shown. The tripping of the protection is not counted, but the fault isolation is counted towards this number.
- **Restoration:** The restoration report displays which switching operations are carried out for each contingency. The tripping of circuit breakers for the fault clearing is shown in the step “Protection”. All switching actions that are carried out to isolate the fault and restore the power supply are shown in the “Restoration” step.

**Note:** If you right-click on a contingency in the *Contingency Summary* report, you can quickly open the corresponding restoration report by selecting *Open Restoration Report*.

---

### 48.7.3 Customised reports

Although the tabular reports are already predefined, the user can modify them if required. The report formats are accessed via the Format tab on the relevant page of the *Contingency Restoration Analysis Reports* dialog and using the right-arrow icon.

Alternatively, users can write their own reporting scripts, which directly access the results in the *ElmRes* result files.

# Chapter 49

# Generation Adequacy Analysis

## 49.1 Introduction

The ability of the power system to be able to supply system load under all possible load conditions is known as *System Adequacy*. Specifically this relates to the ability of the generation to meet the system demand while also considering typical system constraints such as:

- Generation unavailability due to fault or maintenance requirements;
- Variation in system load on an monthly, hourly and minute by minute basis;
- Variations in renewable output (notably wind generation output), which in turn affects the available generation capacity.

The *PowerFactory Generation Adequacy* tool is designed specifically for testing of *System Adequacy*. Using this tool, it is possible to determine the contribution of wind generation to overall system capacity and to determine the probability of *Loss of Load* (LOLP) and the *Expected Demand Not Supplied* (EDNS).

---

**Note:** The Generation Adequacy Assessment is completed using the *Monte Carlo Method* (probabilistic)

---

## 49.2 Technical Background

The analytical assessment of Generation Adequacy requires that each generator in the system is assigned a number of *probabilistic states* which determine the likelihood of a generator operating at various output levels. Likewise, each of the system loads can be assigned a time based characteristic that determines the actual system load level for any point of time. A simplified general illustration of the Generation Adequacy assessment is shown in Figure 49.2.1.

In such a small example, it is possible to determine the generation adequacy analytically in a relatively short time. However, as the number of generators, generator states, loads and load states increases, the degrees of freedom for the analysis rapidly expands so that it becomes impossible to solve in a reasonable amount of time. Such a problem is ideally suited to Monte Carlo simulation.

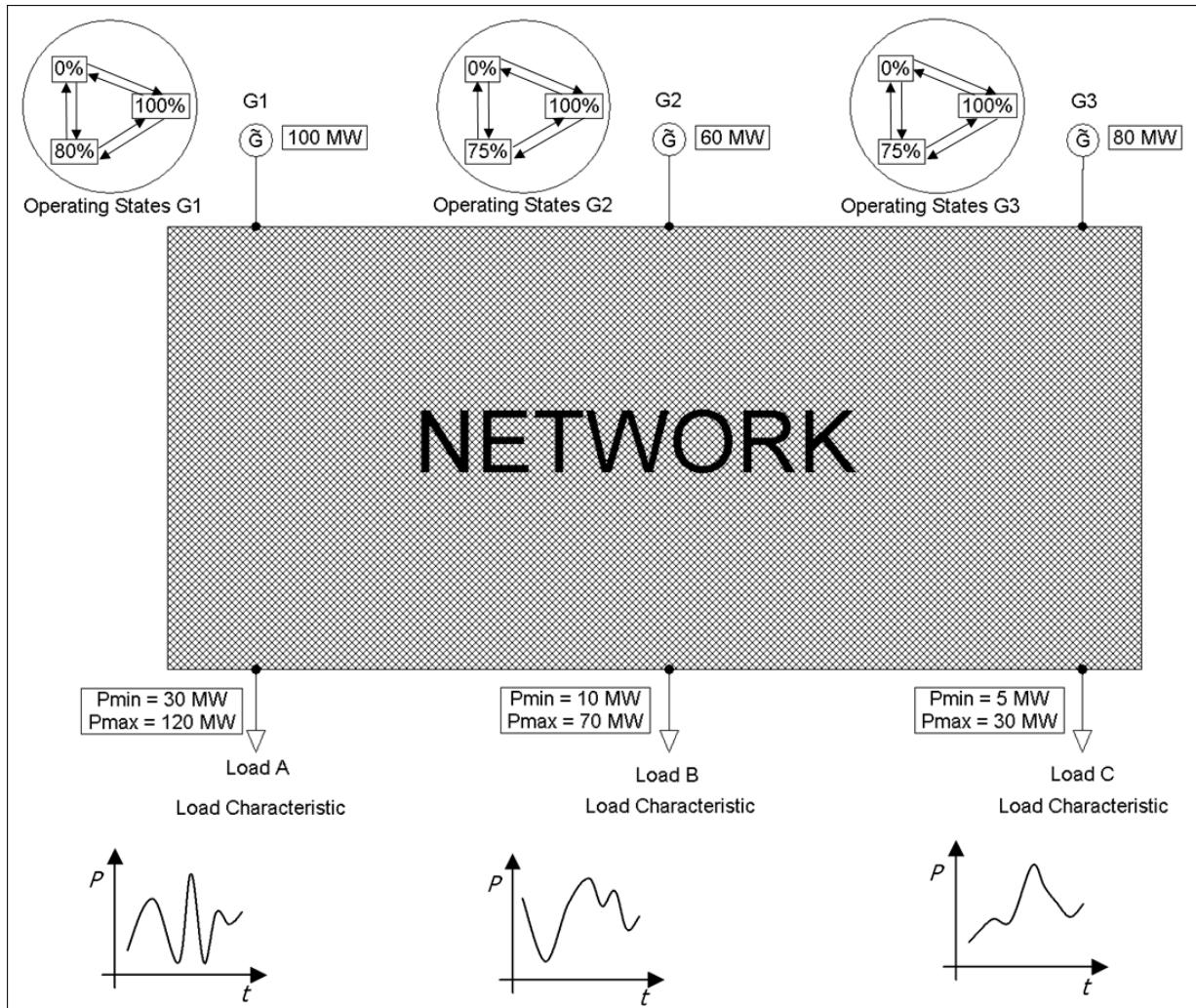


Figure 49.2.1: Generation Adequacy Assessment Illustration

### Monte Carlo Method

In the Monte Carlo method, a sampling simulation is performed. Using uniform random number sequences, a random system state is generated. This system state consists of random generating operating states and of random time points. The generating operating states will have a corresponding generation power output, whereas the time points will have a corresponding power demand. The value of Demand Not Supplied (DNS) is then calculated for such state. This process is done for a specific number of draws (iterations). At the end of the simulation, the values of the Loss of Load Probability (LOLP), Loss of Load Expectancy (LOLE), Expected Demand Not Supplied (EDNS), and Loss of Energy Expectancy (LOEE) indices are calculated as average values from all the iterations performed.

### Pseudo Random Number Generator

A Monte Carlo simulation relies on the generation of random numbers of “high” quality. As all computers run deterministic code to generate random numbers, a software random number generator is known as a pseudo random number generator (PRNG). There are various PRNGs available, some of which do not display appropriate statistical qualities for use in Monte Carlo simulations, where very long sequences of independent random numbers are required.

*PowerFactory* uses an implementation of the 'RANROT' PRNG. This generator displays excellent statistical qualities suitable for Monte Carlo simulations and is also relatively fast.

**Example**

To illustrate the process of a Monte Carlo simulation, an example is now presented using Figure 49.2.1 as the example network.

For each iteration, the operating state for each generator is randomly selected by generating a uniform random number. For each of these states, the corresponding power output of the generator is calculated. The total generation power of the system is calculated by summing all the generator outputs.

For the same iteration, a time point in the system is randomly selected. For this time point, the power demand of each load is obtained. The total demand of the system is calculated by summing all the load demands. It is then possible to obtain the 'Demand Not Supplied' (DNS) value for this iteration, where DNS is defined as shown in equation 49.1.

$$DNS = \sum Demand - \sum Generation \quad (49.1)$$

For example, in the first iteration, the generator states might be G1: 100%, G2: 100%, and G3: 75%. The corresponding outputs would be then G1: 100 MW, G2: 60 MW, and G3: 60 MW. The total generation output is the sum of all the three generator outputs; 220 MW. Also, a random time point yields Load A: 85 MW, Load B: 60 MW and Load C: 30 MW. The total system demand is the sum of all the load demands; 175 MW. Since the generation is greater than the demand, all the demand is supplied and the value of DNS is zero.

In a second iteration, the generator states might be G1: 0%, G2: 75%, and G3: 75%. The corresponding outputs would be then G1: 0 MW, G2: 45 MW, and G3: 60 MW. The total generation output is now 105 MW. A second random time point yields say Load A: 60 MW, Load B: 50 MW, and Load C: 20 MW. The total system demand is now 130 MW. In this case, the generation is smaller than the demand, so there is demand that cannot be supplied. The demand not supplied is defined as the difference between demand and generation - 25 MW in this iteration.

Continuing the analysis for a few subsequent iterations yields the results shown in Table 49.2.1:

Draw	G1 MW	G2 MW	G3 MW	$\Sigma G$ MW	Load A MW	Load B MW	Load C MW	$\Sigma D$ MW	DNS max(0, $\Sigma D - \Sigma G$ )	DNS > 0
1	100	60	60	220	85	60	30	175	0	No
2	0	45	60	105	60	50	20	130	25	Yes
3	80	0	90	170	110	35	10	155	0	No
4	100	60	60	220	40	50	15	105	0	No
5	80	45	90	215	60	40	20	120	0	No
6	80	60	0	140	90	50	5	145	5	Yes
<b>Total</b>									30	2

Table 49.2.1: Example Monte Carlo Analysis

Iteration six yields a second case where demand is not supplied.

Once the analysis has continued in this way (usually for several tens of thousands of iterations) various indices of system adequacy can be calculated. The indices Loss of Load Probability (LOLP) and Expected Demand Not Supplied (EDNS) are the critical measures. They are calculated as follows:

$$LOLP = \frac{N_{DNS}}{N} \cdot 100\% \quad (49.2)$$

$$EDNS = \frac{\sum DNS}{N} \quad (49.3)$$

where  $N_{DNS}$  is the number of iterations where  $DNS > 0$  and  $N$  is the total number of iterations.

Therefore, for the above example the indices are calculated as follows:

$$LOLP = \frac{2}{6} \cdot 100 = 33,33\% \quad (49.4)$$

$$EDNS = \frac{30}{6} = 5MW \quad (49.5)$$

## 49.3 Database Objects and Models

There are several database objects in *PowerFactory* specifically related to the *Generation Adequacy Analysis*, such as:

- Stochastic Model for Generation object (*StoGen*);
- Power Curve Type (*TypPowercurve*); and
- Meteorological Station (*ElmMeteostat*).

This section provides information about each of these objects.

### 49.3.1 Stochastic Model for Generation

The Stochastic Model for Generation object (*StoGen*) is used for defining the availability states of a generator, an example of which is shown in Figure 49.3.1. An unlimited number of states is possible with each state divided into:

**State** : name of the state

**Availability [% ]**: percentage of the rated power available

**Probability [% ]**: probability that this state is valid (the sum of all probabilities must always be 100 %)

**Duration [h ]**: time needed to solve the given failure

**Frequency [1/a ]**: number of incidents that cause the given state per year

**Total Duration [h/a ]**: total duration of the given state per year

While only the parameters *Availability* and *Probability* are used for the *Generation Adequacy*, all parameters are used for the Reliability Assessment, see Section 46.3.1.5.

This means that for each state, the total available generation capacity in % of maximum output must be specified along with the probability that this availability occurs. Note that probability column is automatically constrained, so that the sum of the probability of all states must equal 100 %.

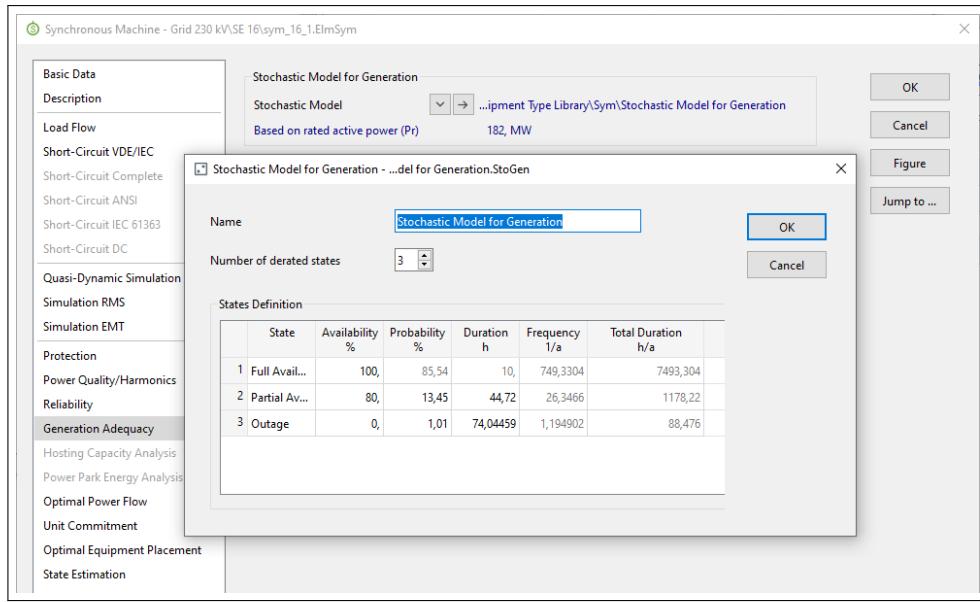


Figure 49.3.1: Stochastic Model for Generation dialog

The Stochastic model for generation object should reside within the project library, *Equipment Type Library*.

Note that the generator maximum output is calculated as  $S_{nom} \cdot \cos \theta$  where  $S_{nom}$  is the rated apparent power and  $\cos \theta$  is the rated power factor.

### 49.3.2 Power Curve Type

The Power Curve Type (*TypPowercurve*) object is used to specify the wind speed (in m/s) vs rated power output (p.u or MW) for wind turbine generators.

For wind-speed values between specified curve values, *PowerFactory* interpolates using the method specified in the *Approximation* drop down menu. Interpolation options include:

- constant
- linear
- polynomial
- spline and
- hermite.

To change the Power unit, go to the configuration tab and choose either p.u or MW by selecting the appropriate radio button.

### 49.3.3 Meteorological station

If a group of wind generators have a wind speed characteristic that is correlated, it can be represented through the *Meteorological Station* object (described in Section 15.7).

Note that when two wind generators are correlated as members of the same *Meteorological Station*, they may still have different average wind speeds defined within their *Generation Adequacy* page. During the Monte Carlo Analysis, a random wind speed is drawn for each *Meteorological Station*. This wind

speed is then applied to every wind generator in that *Meteorological Station* using the Weibull Stochastic Model. Thus, the power is calculated according to the individual power curve of the generator.

When the generator is using time characteristics as a wind model, then the correlation is given by the Monte Carlo drawn time, which is the same for all the generators of the system.

## 49.4 Assignment of Stochastic Model for Generation

For the Generation Adequacy Analysis, there is a distinction between *Dispatchable (Conventional) Generation* and *Non-dispatchable Generation*. Dispatchable generation refers to generation that can be controlled at a fixed output automatically, typically by varying the rate of fuel consumption. This includes generation technologies such as gas thermal, coal thermal, nuclear thermal and hydro.

Non-dispatchable generation refers to generation that cannot be automatically controlled because the output depends on some non controllable environmental condition such as solar radiation or the wind speed. Wind turbine and solar photovoltaic generators are examples of such *environmentally dependent* generation technologies.

### 49.4.1 Definition of a Stochastic Multi-State Model

For both Dispatchable and Non-dispatchable generation it is possible to assign a Stochastic Multi-State model to define the availability of each unit. The availability is defined in a number of 'States' each with a certain probability as described in Section [49.3.1](#).

- Synchronous machine (*ElmSym*);
- Static generator (*ElmGenstat*) set as *Fuel Cell*, *HVDC Terminal*, *Reactive Power Compensation*, *Storage*, or other *Static Generator*;
- Asynchronous machine (*ElmAsm*); and
- Doubly-fed asynchronous machine (*ElmAsmsc*)

In all cases, the stochastic model object is assigned on the element's *Generation Adequacy* page, under *Stochastic Multi-State Model*.

Also, to consider the generation as *dispatchable*, the *Wind Generation* option in the *Basic Data* page of the synchronous, asynchronous, and doubly fed machine should be disabled.

### Definition of a Stochastic Model for Non-Dispatchable (Wind and Renewable) Generation

As for the dispatchable generation, the following 3-phase models are capable of utilising the stochastic model for generation object, provided they are defined as generators and not as motors:

- Synchronous machine (*ElmSym*) set as *Wind Generator*;
- Static generator (*ElmGenstat*) set as *Wind Generator*, *Photovoltaic* or *Other Renewable*
- Asynchronous machine (*ElmAsm*) set as *Wind Generator*; and
- Doubly-fed asynchronous machine (*ElmAsmsc*) set as *Wind Generator*

**Objects not considered in Generation Adequacy Analysis** External Grids (*ElmXnet*), voltage and current sources (*ElmVac*, *Elmlac*) are ignored in the Generation Adequacy analysis.

#### 49.4.2 Stochastic Wind Model

In addition to the stochastic multi-state model for generation described above, a stochastic wind model may be defined on the element's *Generation Adequacy* page (provided that the type of generation is a wind generator). To enable this, navigate to the *Generation Adequacy* page and check the option *Wind Model*.

When the Stochastic Wind Model is selected, the wind generation characteristic is described using the Weibull Distribution. The mean wind speed, and shape factor (Beta) of the distribution can be adjusted to achieve the desired wind characteristic for each wind generator.

In addition to describing the Weibull distribution using Mean Wind Speed and Beta, the following alternate methods of data input can be used:

- Mean Wind Speed and Variance;
- Lambda and Variance;
- Lambda and Beta.

The input method can be changed by using the input selection arrow and choosing the desired method from the input window that appears.

#### 49.4.3 Time Series Characteristic for Wind Generation

If detailed data of wind generation output over time or wind speed over time is available, then this can be used instead of a Stochastic Model. The data can be read by *PowerFactory* as either a *ChaVec* characteristic or from an external file using the *ChaVecfile* characteristic. In both cases the information required is one year of data in hourly intervals - although non integer values can also be specified in the referenced data.

If the option *Time Series Characteristics of Wind Speed* is selected, then the actual wind generator power output for each iteration is calculated automatically from the Wind Power Curve. If the option, *Time Series Characteristic of Active Power Contribution* is selected then no power curve is required.

Data for multiple years can also be used by referencing an additional characteristic for each year. The *Generation Adequacy* algorithm then selects a random wind speed or power value from one of the input data years - essentially there is more data for the random Monte Carlo iteration to select from.

A screen-shot showing a wind generator model with three years of data is shown in Figure 49.4.1.

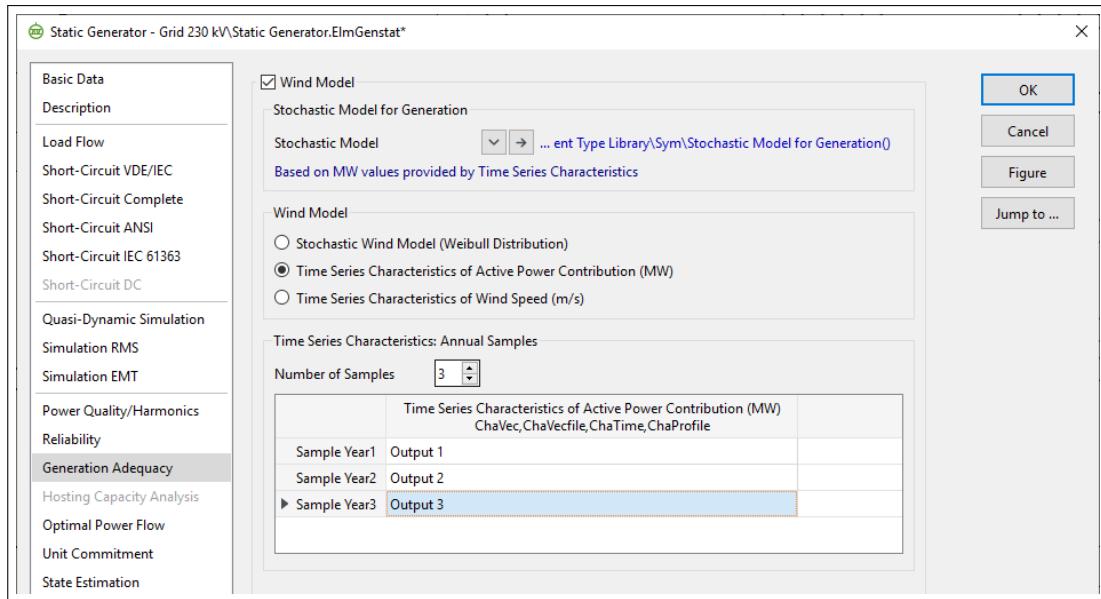


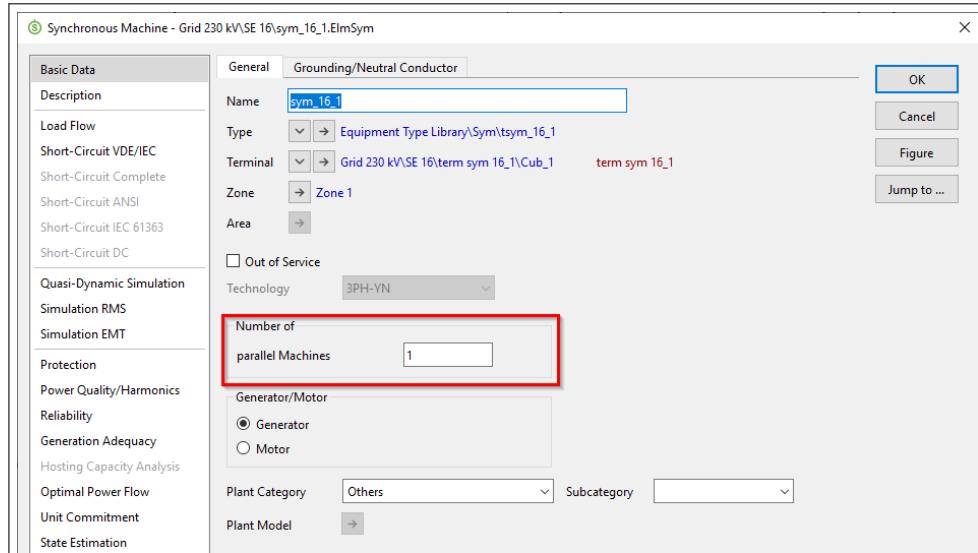
Figure 49.4.1: Wind Model using Wind Output Data

## Other Renewable Generation

Static Generators (*ElmGenstat*) of category *Photovoltaic* or *Other Renewable* cannot have a Stochastic wind model definition. However, they may still have a *Stochastic Multi-State model*. Their output is added to the aggregated non-dispatchable generation as described later in this chapter.

## Consideration of Parallel Machines

The Generation Adequacy analysis automatically considers parallel machines defined in the basic data of the generator object using the variable (*nghnum*), as shown in Figure 49.4.2. Each of the parallel machines is treated independently. For example, a random operational state is generated for each of the parallel machines. Effectively this is the same as if *n* machines were modelled separately.

Figure 49.4.2: Synchronous machine element with the parameter *nghnum* (number of parallel machines highlighted).

#### 49.4.4 Demand definition

Unless a time characteristic is assigned to either the Active Power (*plini*) or Scaling Factor (*scale0*) variables of the load element, then the load is treated as fixed demand. This means that the demand value does not change during the entire analysis. Both General Loads (*ElmLod*) and LV Loads (*ElmLodlv*) are considered for the analysis.

More information about assigning time based characteristics to object variables can be found in Chapter 18: Parameter Characteristics, Load States, and Tariffs.

### 49.5 Generation Adequacy Analysis toolbar

Once the Generation Adequacy toolbar is selected, the available buttons are shown in Figure 49.5.1.

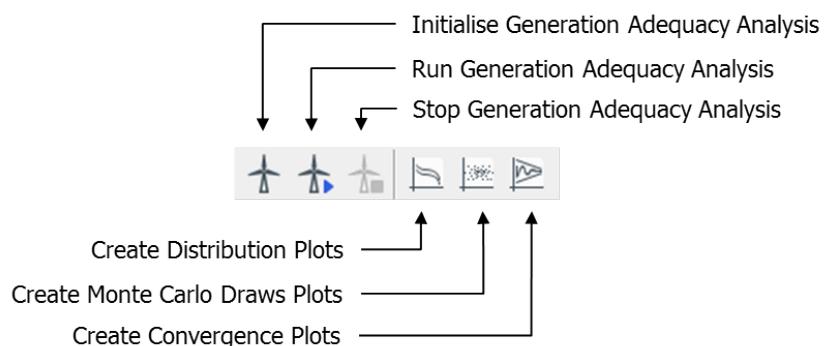


Figure 49.5.1: Generation Adequacy Analysis toolbar buttons

#### 49.5.1 Generation Adequacy Initialisation command

Before a Generation Adequacy Analysis can be completed, the simulation must be initialised. The available options of the *Initialise Generation Adequacy Analysis* command (*ComGenrelinic*) are explained in this section.

##### 49.5.1.1 Basic Options page

###### Network

- System Losses: here a fixed percentage of losses can be entered. This value is subtracted from the total generation at each iteration.
- Load Flow Command: this is a reference to the *Load Flow Calculation* command that will be used to obtain the network topology for the analysis. It must be set to *AC load-flow balanced, positive sequence* or *DC load-flow*. A converging load flow is a requirement for the generation adequacy analysis.

###### Demand Consideration

- Fixed Demand Level: if this option is selected, all load time characteristics are ignored and the total demand is calculated at the initial iteration and used for all subsequent iterations.

- Consider Time Characteristics: if this option is selected, any time characteristics assigned to loads will be automatically considered in the calculation. Therefore, the total demand can vary at each iteration.

### Consider Maintenance Plans

If this option is enabled then any maintenance plans (out of service or derating) in the project will be automatically considered. Consequently, when an iteration draws a time that falls within a planned outage or derating, the outage (or derating) is applied to the target element resulting in a reduction in available generation capacity.

To define a maintenance plan, right-click the target object from the single line graphic or from the Data Manager and select the option *Operational Library* → *Planned Outage* → *New...*. For more information on Planned Outages refer to Chapter 14: Libraries, Section 14.6.7 (Planned Outages).

### Time Dependent Data

- Year of Study: the period considered for the Generation Adequacy Analysis is always one year. However, it is possible for load characteristics to contain information for many years. Therefore, the year considered by the calculation must be selected. Note that this variable does not influence the wind speed or wind power data if the wind model for the generator references time series data as described in Section 49.4.3 (Time Series Characteristic for Wind Generation). If more than one year's data is available, this simply increases the *pool* of available data for the analysis.
- Months, Days: these checkboxes allow the user to select the time period that will be considered for the analysis. For instance, if only *January* is selected then the iteration time will be constrained to within this month.

### Time Intervals

The user can specify up to three time intervals for the time window in which the analysis will be completed. The time interval starts at the *From* hour (0 minutes, 0 seconds), and ends at the *To* hour (0 minutes, 0 seconds) inclusive.

#### 49.5.1.2 Output page

- MC Draws: if this option is checked, then *PowerFactory* will automatically create Monte-Carlo Draw plots after the simulation finishes. See Section 49.6 for details of the plots that are automatically created. Note this will generate a new set of plots for each run of the analysis. So, if you wish for an existing set of plots to be updated, then leave this option unchecked.
- Distribution: here the user can select the storage location for the distribution probabilities for the entire analysis. This information is always retained in the database.
- Report: if this option is checked, then the user can specify a location for the results of the simulation to be permanently stored within the database. This is the result of each iteration. If this option is unchecked, then the results are deleted after each simulation run.

#### 49.5.1.3 Advanced Options page

In the *Advanced Options* page, the user can change the option for the generation of random numbers from *auto* to *renew*. If the *renew* option is selected, then the simulation can use one of a number of pre-defined random seeds (A-K). As the software 'pseudo-random' number generator is deterministic, this allows for the exact sequence of random numbers to be repeated.

### 49.5.2 Run Generation Adequacy command

The *Run Generation Adequacy Analysis* command (*ComGenrel*) appears in two styles depending on the status of the calculation. If the calculation is being run for the first time, then it appears as shown in Figure 49.5.2. On the other hand, if some iterations are already complete, then the calculation can be continued and the dialog appears as shown in Figure 49.5.3.

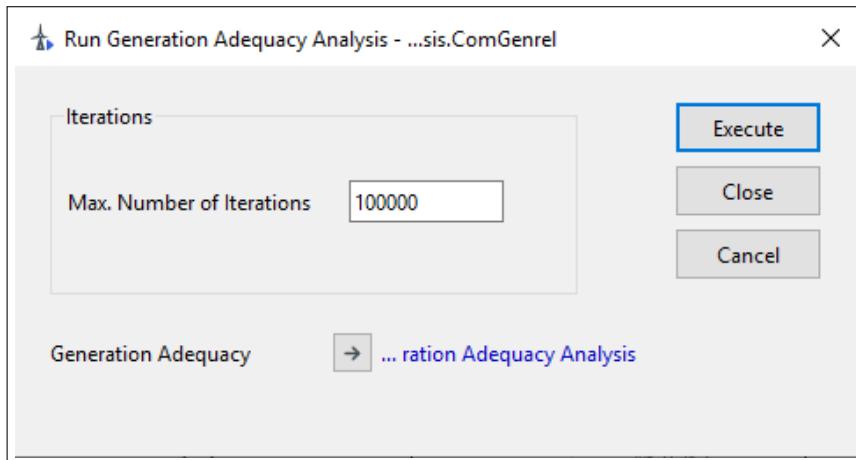


Figure 49.5.2: Run Generation Adequacy command dialog (new simulation)

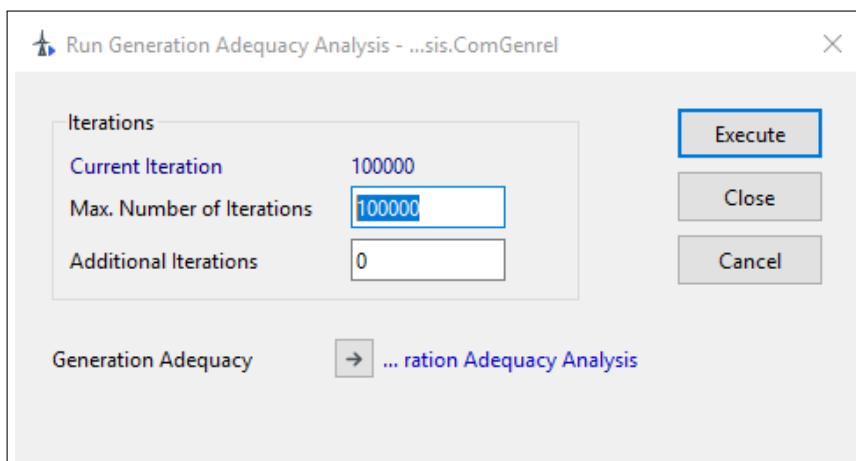


Figure 49.5.3: Run Generation Adequacy command dialog (post simulation)

Pressing **Execute** will run the Generation Adequacy Analysis. The button can be used to interrupt the analysis before the set number of iterations is complete, if desired. Later, the simulation can be resumed from the *stop point* using the *Run Generation Adequacy Analysis* command.

#### Max Number of Iterations

This specifies the number of iterations to be completed by the Monte Carlo Analysis. The default setting is 100000.

#### Additional Iterations

After one analysis is completed, the Generation Adequacy Analysis can be extended for a number of *Additional Iterations*. Especially in very large systems, it may be useful to run the first simulation with a smaller number of initial iterations, say 20000 and then run additional iterations as necessary using this

option.

### Generation Adequacy

This reference provides a link to the generation adequacy initialisation command, so that the calculation settings can be easily inspected.

## 49.6 Generation Adequacy results

Result plots for the Generation Adequacy Analysis can be manually created using the toolbar plot icons. The different types of plots are explained in the following sections.

### 49.6.1 Distribution (Cumulative Probability) Plots

This button  draws a distribution plot which is essentially the data from 'Monte-Carlo Draws' plots sorted in descending order. The data then becomes a cumulative probability distribution. An example is shown in Figure 49.6.1.

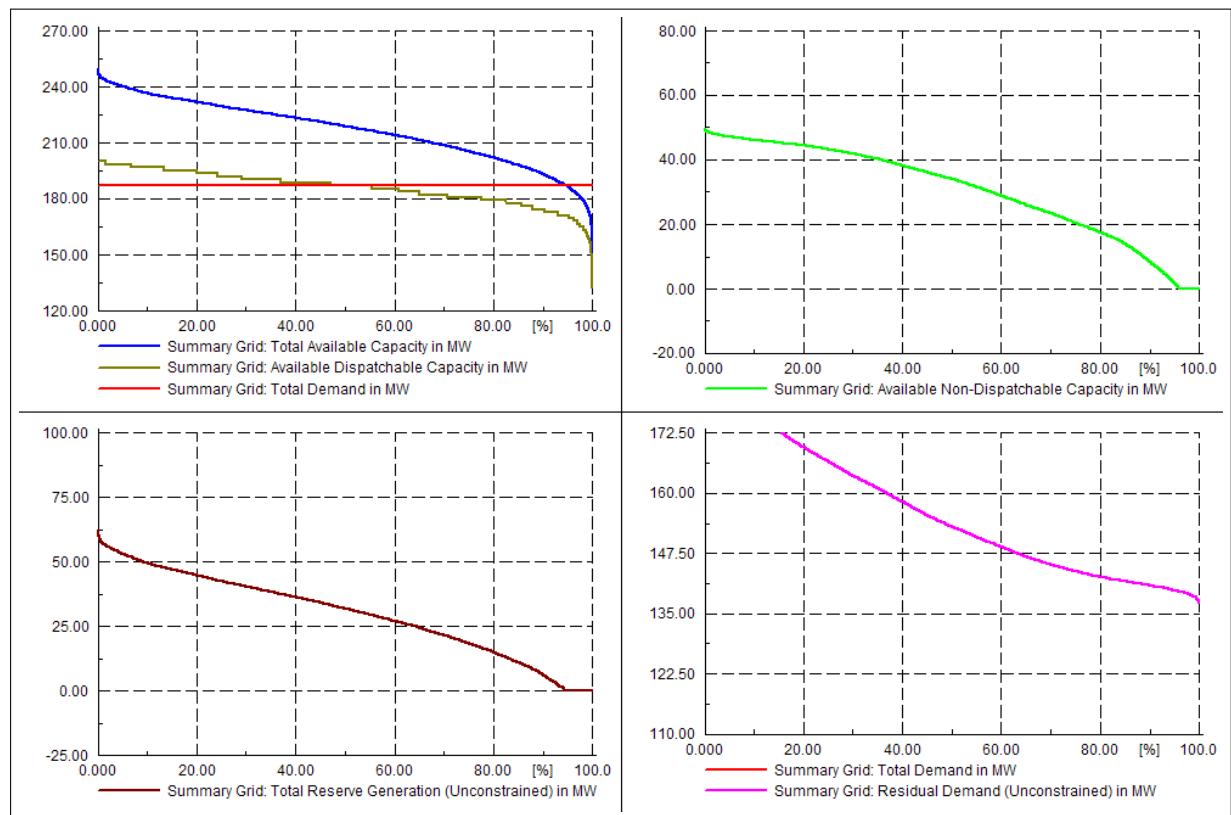


Figure 49.6.1: Distribution (Cumulative Probability) Plots

### Obtaining the LOLP from the Distribution Plots

The LOLP index can be obtained by inspection directly from the Distribution Plots if the demand is constant. The LOLP can be read directly from the intersection of the Total Generation curve and the Total Demand curve as demonstrated in Figure 49.6.2.

When the demand is variable, then the LOLP index cannot be inferred from the above diagram. Figure 49.6.3 shows such a case. There is no intersection point even though the calculated LOLP index in this case is 20 %. In such cases, the LOLP index must be inferred from the distribution plot of the Total Reserve Generation. As shown in Figure 49.6.4, the intersection of this curve with the x-axis gives the LOLP index.

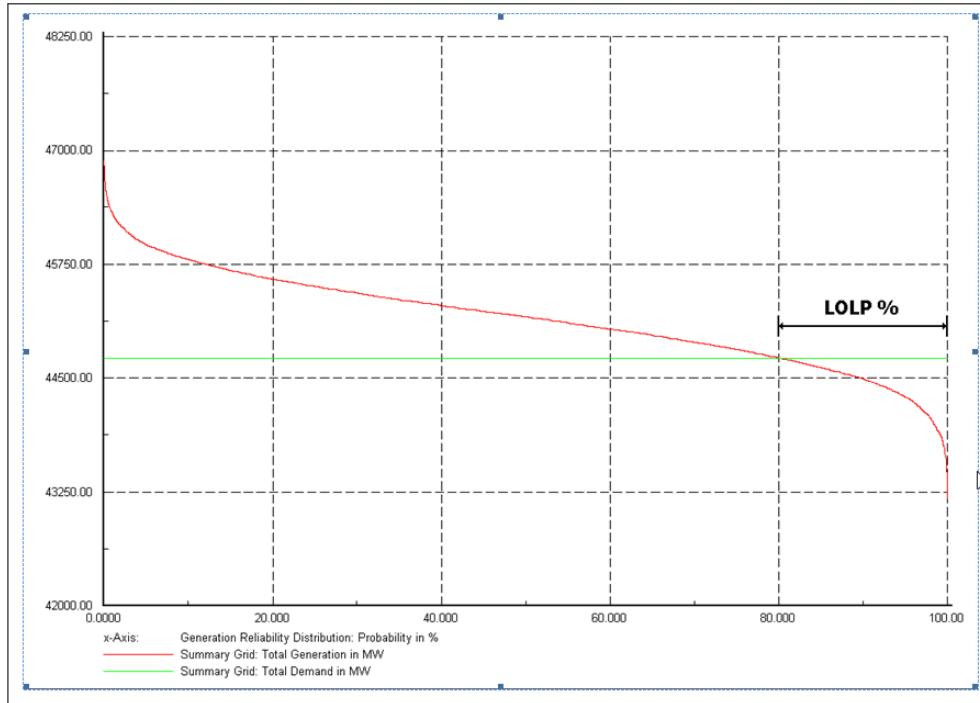


Figure 49.6.2: Inferring the LOLP index directly from the intersection of the Total Generation and Total Demand

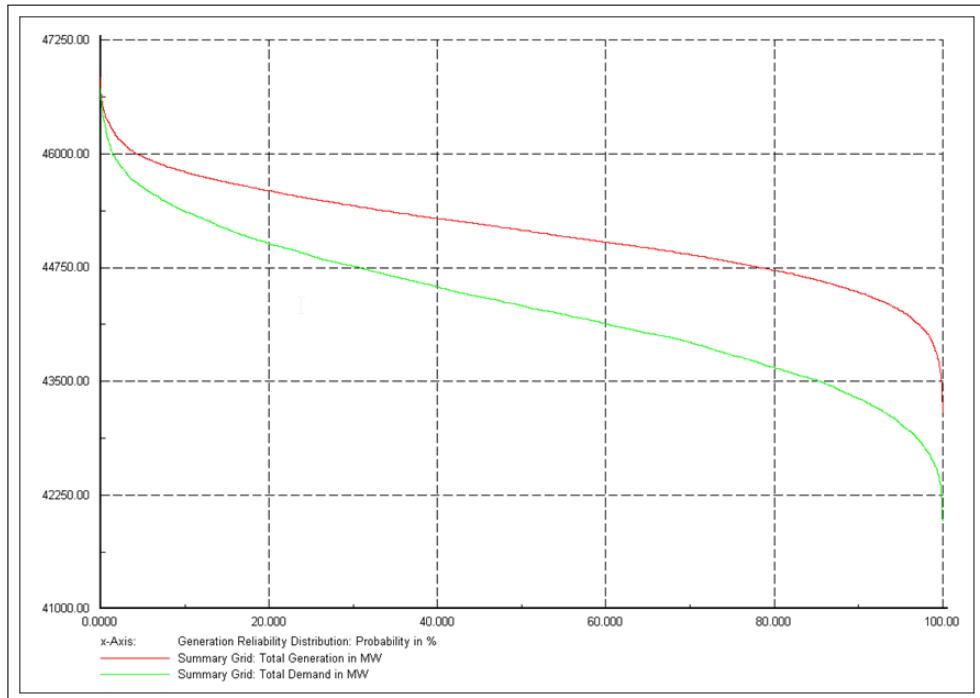


Figure 49.6.3: Variable Demand - distribution of Total Generation and Total Demand

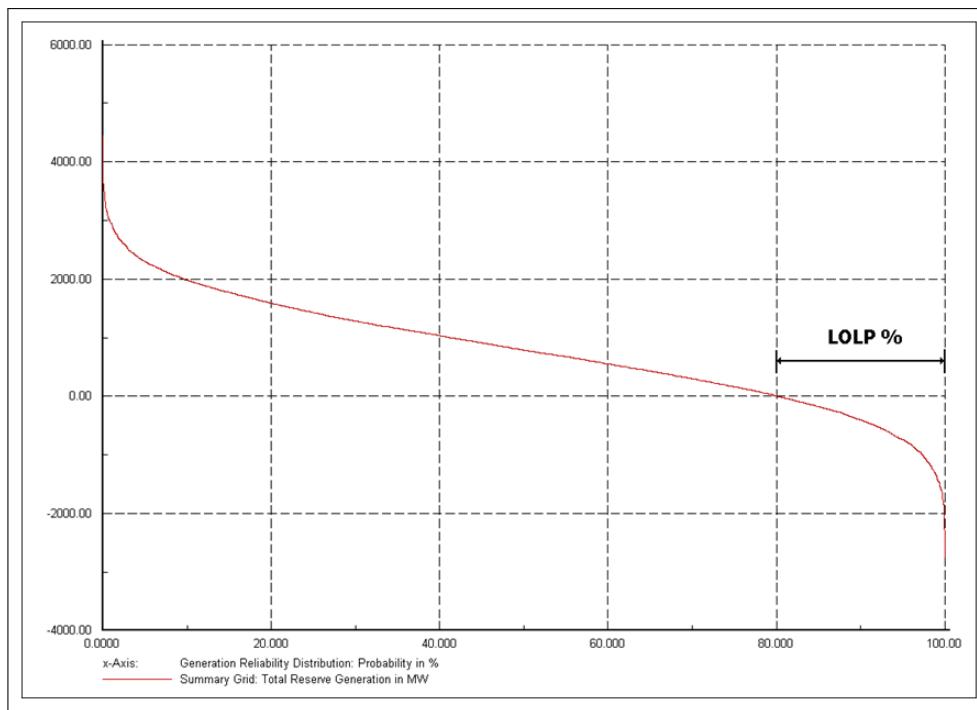


Figure 49.6.4: Total Reserve Generation

## 49.6.2 Monte-Carlo Draws (Iterations) Plots

These plots are automatically generated if the *MC Draws* option is enabled in the *Output* page of the initialisation command, alternatively, the button ( ) can be used. This button draws by default four figures as shown in Figure 49.6.5. Each of the data points on the plots represents a single Monte Carlo simulation.

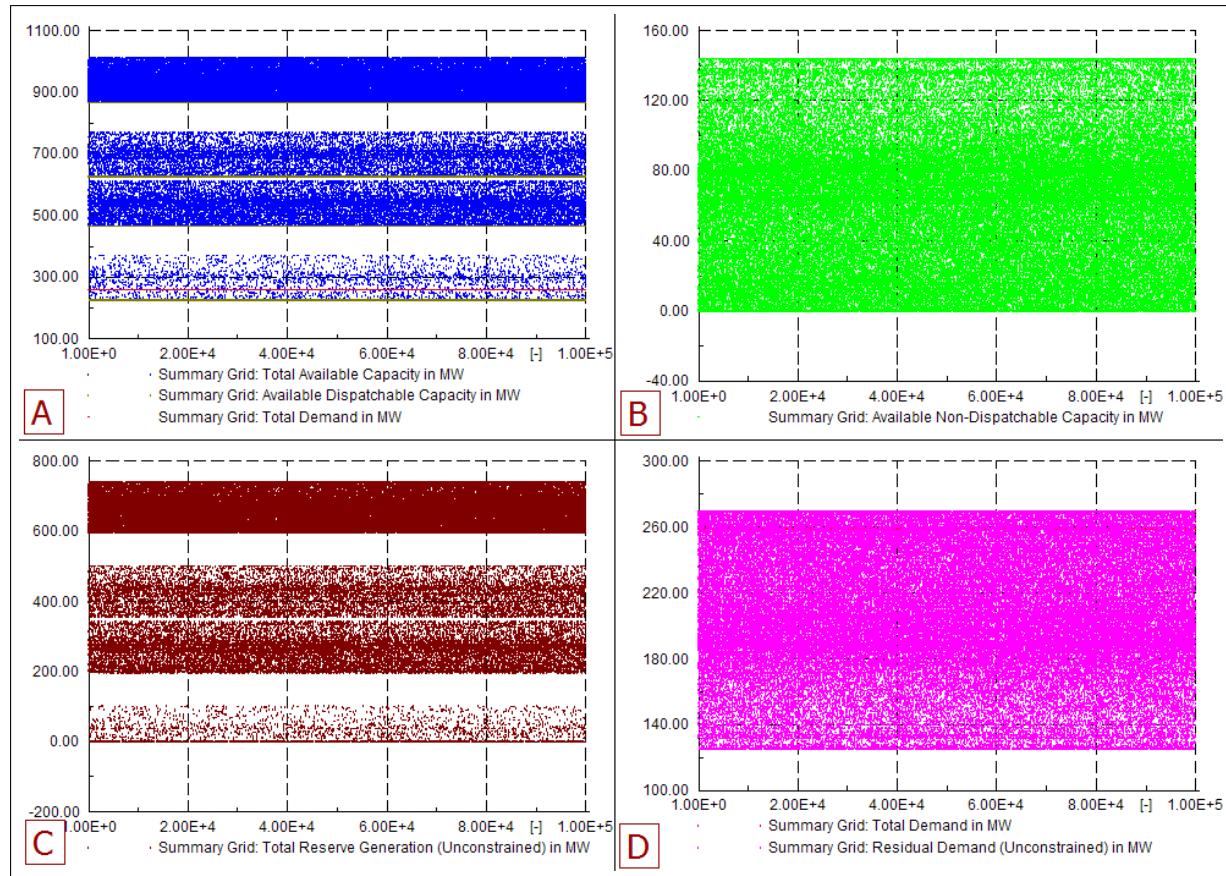


Figure 49.6.5: Monte-Carlo Draws (Iterations) Plots

Figure A displays the following:

- Total Available Capacity in MW;
- Available Dispatchable Generation in MW;
- Total Demand in MW;

Figure B displays the following:

- Available Non-dispatchable capacity in MW;

Figure C displays the following::

- Total Reserve Generation Capacity in MW;

Figure D displays the following::

- Total Demand in MW;
- Residual Demand in MW;

### 49.6.3 Convergence Plots

This button ( ) creates the so-called convergence plots for the LOLP and EDNS. As the number of iterations becomes large the LOLP index will converge towards its final value, likewise for the EDNS. The convergence plots are a way of visualising this process. An example convergence plot is shown in Figure 49.6.6.

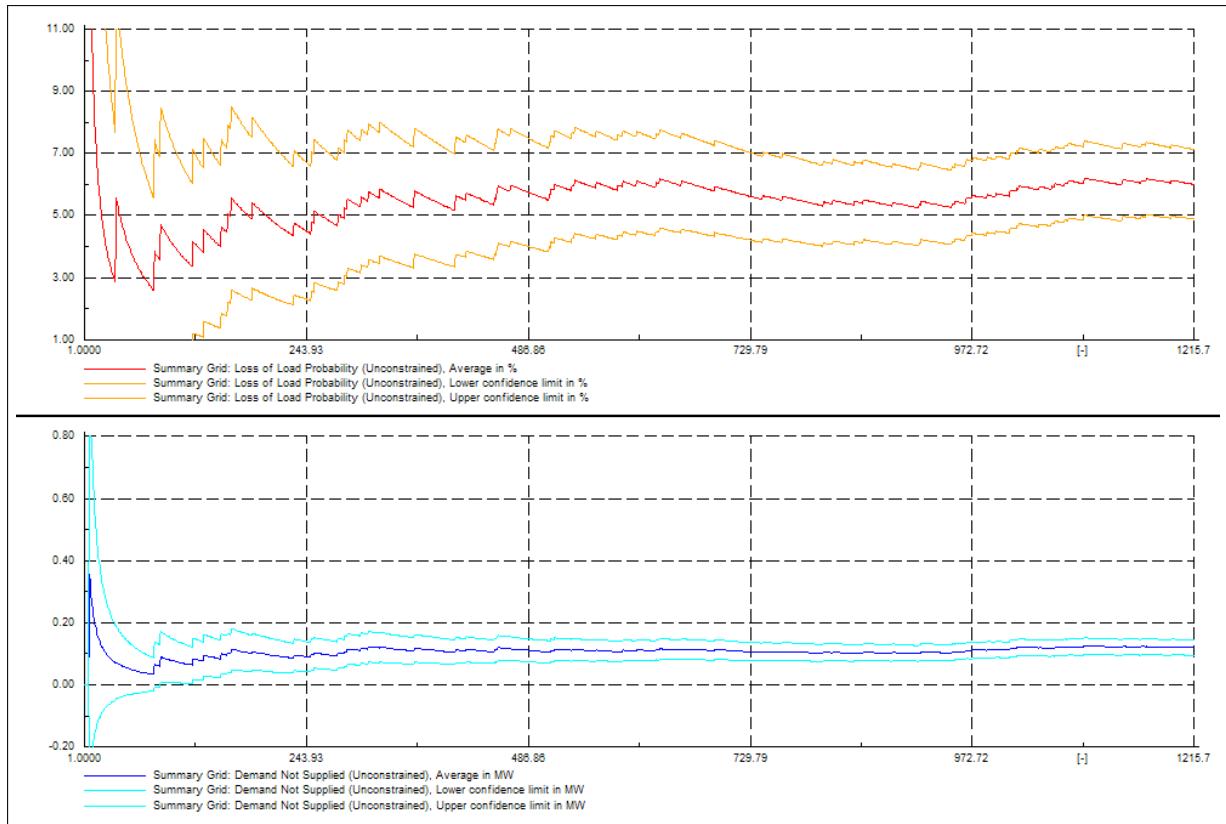


Figure 49.6.6: Example Convergence Plot

**Note:** By default, the convergence plot is zoomed to the plot extent and due to the number of iterations it may be difficult to observe the upper and lower confidence limits. It is suggested that the 'Zoom Y-axis' and 'Zoom X-axis' buttons are used to observe the confidence limits in greater detail.

On both plots, the upper and lower confidence intervals are also drawn.

The sample variance is calculated as follows:

$$\sigma^2 = \frac{1}{n-1} \cdot \sum_{i=1}^n (y_i - \bar{y})^2 \quad (49.6)$$

where  $n$  is the number of samples,  $y_i$  is the sample and  $\bar{y}$  is the sample mean. The 90 % confidence interval is calculated according to the following formula:

$$CL = \bar{y} \pm \frac{\sigma}{\sqrt{n}} \cdot z \quad (49.7)$$

where  $z$  is the standard inverse probability for the *Student's t distribution* with a confidence interval of 90 %. Note  $z$  tends to 1.645 (inverse normal) as the number of iterations becomes large.

#### 49.6.4 Summary of variables calculated during the Generation Adequacy Analysis

Name	Internal Name	Description
Available Dispatchable Capacity	c:AvailDCap	The sum of dispatchable capacity at each iteration after the consideration of the availability states
Available Non-Dispatchable Capacity	c:AvailNDCap	The sum of non-dispatchable capacity at each iteration after the consideration of the availability states and also the stochastic/time models for wind generation
Total Available Capacity	c:AvailTotcap	c:AvailNDCap + c:AvailDCap
Total Demand	c:DemTot	Total Demand considering any time based characteristics
Demand Supplied	c:DemS	$\min(C:DemTot, c:AvailTotcap * (1 - Losses\% / 100))$
Demand Not Supplied	c:DNS	$c:DemTot - DemS$
Total reserve Generation	c:ResvTotGen	$c:AvailTotCap - c:DemTot * (1 + Losses\% / 100)$
Reserve Dispatchable generation	c:ResDG	$c:AvailDCap - c:DemTot * (1 + Losses\% / 100)$
Used Non-Dispatchable Generation	c:NDDG	$\min(C:AvailNDCap, DemTot * (1 + Losses\% / 100))$
Used Dispatchable Generation	c:DDG	$\min(C:AvailDCap, DemTot * (1 + Losses\% / 100) - c:NDDG)$
Total Used generation	c:TotGen	$c:Dgen + c:NDDG$
Residual Demand	c:ResidDem	$c:DemTot * (1 + Losses\% / 100) - c:DDG$

Table 49.6.1: Generation Adequacy Calculated Variables

# Chapter 50

## Sensitivities / Distribution Factors

*PowerFactory's Sensitivities / Distribution Factors* calculation (*ComVstab*) offers a range of sensitivity analyses based on the linearisation of the system around the operational point resulting from a load flow calculation. Formerly referred to as *Load Flow Sensitivities*, in version 2020 of *PowerFactory*, the functionality was extended and renamed *Sensitivities / Distribution Factors*.

The *ComVstab* command dialog can be accessed by:

- using the *Change Toolbox* icon (▼) to select *Additional Functions* and then clicking on the *Sensitivities / Distribution Factors* icon (δ); or
- right-clicking on a busbar/terminal or transformer and selecting *Calculation → Sensitivities / Distribution factors...*. In this case the command will be automatically set to calculate the sensitivity to power injections / tap changes on the selected busbar or terminal.

### 50.1 Overview of Sensitivity / Distribution Factors Calculations

There is a range of sensitivity calculations available to the user, including single sensitivity calculations (for example to power injections at a single busbar or multiple busbars concurrently) and multiple sensitivities (for example to power injections at many busbars in turn). In addition to the base load flow sensitivities, there are also options to consider contingencies, which lends another dimension to the results.

If a single sensitivity calculation is being executed, the results can be viewed directly in a Network Model Manager. For multiple sensitivities only the results of the last-executed calculation are shown in the Network Model Manager; reporting scripts can be used to access the complete set of results in the result file.

#### 50.1.1 Terminology

The Load Flow Sensitivities calculation is referred to as “Sensitivity / Distribution Factors” because within the field of electricity transmission in particular such sensitivity calculations are referred to as distribution factors.

Reference will be made to some of these; the acronyms used are explained in general terms here:

- **PTDF (Power Transfer Distribution Factor):** The change in power flow on a circuit for a given power injection at one or more nodes.

- **LODF (Line Outage Distribution Factor):** The percentage of the power flow of a circuit that can be observed on another branch if there is an outage of the circuit.
- **OTDF (Outage Transfer Distribution Factor):** The change in power flow on a circuit for a given pre-fault power injection at one or more nodes, under fault outage conditions. This quantity is obtained by running a PTDF calculation for each contingency case.
- **TCDF (Tap Change Distribution Factor):** The change in power flow on a circuit as a result of tapping on a transformer.
- **PSDF (Phase Shift Distribution Factor):** The change in power flow on a circuit as a result of tapping a phase-shift transformer. In essence, PSDF=TCDF.

---

**Note:** Users of the Transmission Network Tools module (Section 41.4) may already be familiar with the PTDF functionality provided there. Although the underlying calculation is the same, the approach is somewhat different and the choice between the two will depend on the individual requirements of the user. The PTDF calculation in the Transmission Network Tools module offers some flexibility in terms of customising the generator and load changes, and allows the reporting of sensitivities across flowgates, but is restricted to region-to-region factors; the PTDF calculation here has the benefit of offering sequential busbar injections, as well as incorporating contingency analysis.

---

### 50.1.2 Result Quantities

Depending on the calculations executed, the following results will be available:

- **On branch elements for a power-injection sensitivity:**  $dP/dP$ ,  $dP/dQ$ ,  $dQ/dP$  and  $dQ/dQ$ .  
Also  $dP_{loss}/dP$ ,  $dP_{loss}/dQ$ ,  $dQ_{loss}/dP$  and  $dQ_{loss}/dQ$ .
- **On branch elements for a transformer-tap sensitivity:**  $dP/dtap$  and  $dQ/dtap$ .  
Also  $dP_{loss}/dtap$  and  $dQ_{loss}/dtap$ .
- **At terminals for a power-injection sensitivity:**  $dv/dP$ ,  $dv/dQ$ ,  $dphi/dP$  and  $dphi/dQ$ .
- **At terminals for a transformer-tap sensitivity:**  $dv/dtap$ , and  $dphi/dtap$ .

## 50.2 Sensitivities / Distribution Factors Options

### 50.2.1 Basic Options

#### 50.2.1.1 Calculation method

Here the load flow calculation method is selected and the Load Flow dialog can be accessed.

---

**Note:** If an unbalanced AC load flow is used, only busbar sensitivities can be calculated and the reporting options are similarly limited.

---

#### 50.2.1.2 Consider contingencies

As well as executing the requested sensitivity analyses on the base load flow, it is possible to provide a set of contingencies so that the requested sensitivities are calculated for these too. Note that an option

“Use linearised AC calculation” is provided. See Section [27.4](#) for more information about this option; it can speed up the calculation but the results may not be so accurate.

### 50.2.1.3 Sensitivities / Distribution factors

Several options are available; it is possible to select more than one to be run during the sensitivity analysis calculation, but they are separate calculations.

#### Busbar

This busbar injection option corresponds to a PTDF calculation. The user can select a single terminal or a set of terminals for analysis. Unless only one terminal is selected, the user should also decide whether the power injections should be concurrent or sequential. This option is found on the Advanced Options page: see Section [50.2.3](#). On the other hand, the user can select regions such as Zones, Areas and Grids etc. In this case the option “Calculate regional sensitivities” can be selected in the Advanced Options page: refer to Section [50.2.3](#).

Along with busbars, several elements such as *ElmGenstat*, *ElmLod*, *ElmPvsys*, *ElmSym* and *ElmVac*, are shown for the selection. For convenience the user can select the busbars by directly choosing the corresponding generators and loads.

If the *Consider contingencies* option is selected, the PTDF calculations for the contingency cases correspond to an OTDF calculation.

#### Phase Shift/Tap Change

This option corresponds to a PSDF calculation. The user can select a single transformer or tap controller, or a set of transformer / tap controllers for analysis. Unless only one transformer / tap controller is selected, the user should also decide whether the tapping should be concurrent or sequential. This option is found on the Advanced Options page: see Section [50.2.3](#).

#### Sensitivity to HVDC

This option corresponds to a PTDF calculation. The user can select a HVDC element, or a set of HVDC elements (*ElmVsc*, *ElmVsmono*, *ElmRec* and *ElmRecmono*) for analysis. If only this option is selected, the options in the Advanced Options page (see Section [50.2.3](#)) will be greyed out. In case of multiple HVDC elements, the calculation is always sequential.

#### Line Outage Distribution Factors

This option corresponds to a LODF calculation. To enable this option, the *Consider contingencies* option is selected and a set of contingencies must be made available.

Line Outage Distributions Factors are by default calculated for line elements. However, it is also possible to calculate LODF for other branch elements, i.e. 2- or 3-winding transformers, series capacitors and series reactors. To obtain these results, it is necessary to use the **Variable Selection** feature described in the next section to select the relevant variables to be recorded. The Variable Selection (*Intmon*) dialog should be opened for the relevant element class, and the “LODF” variables selected.

## 50.2.2 Results

### 50.2.2.1 General Tab

#### Elements for results

Here the user has the choice to take the default option of calculating results for the whole system or applying a user-defined element filter.

The User-defined filter option allows the user to set up or modify filters in order to specify for which elements results should be made available. Using the filter to restrict results to elements of interest can significantly reduce calculation times. The filter applies whether a result file is used or not.

### Calculate sensitivities for reducible elements

Reducible elements are those such as switches, which have no resistance and can therefore effectively be eliminated during calculations. Unless the user requires results for these elements, it is better to leave this option unselected because it will improve the speed of the calculation.

### Result File

In this panel, there is a link to the “results object” (\*.ElmRes), which is held inside the Study Case. Note that if contingency analysis has been included, there will be individual results files for each contingency case. A button **Sub-Results** in the results object gives access to these.

Below the link to the results object, there are two further buttons relating to variable selection (the default set of variables recorded in the result file depends on the calculation method selected on the Basic Options page):

- The **Variable Selection** button allows the user to specify variable selections for the recording of results. If no variables are selected for recording, the full set of default variables will automatically be selected.
- The **Add default variables** button allows the user to include all default variables to the selection used for the recording of results.

---

**Note:** When executing sensitivity analysis on a large network it is not recommended that the full set of default variables is used. Instead, it is better to define specific variable definitions to restrict the recording to what is required. This will help speed up the calculation. The Variable Selection object is described in Section 19.3: [Variable Selection](#).

---

### Consider recording thresholds for branches

Two main thresholds can be set. One is a minimum reporting level for the relative quantities  $dP/dP$ ,  $dP/dQ$ ,  $dQ/dP$  or  $dQ/dQ$ . The other is a minimum reporting level for “Changes per tap”, i.e.  $dP/dtap$  or  $dQ/dtap$ .

In addition, a third threshold called *Min. LODF* will be available, if the Line Outage Distribution Factors are being calculated.

### Consider recording thresholds for terminals

Two thresholds can be set. One is for voltage changes, namely  $dv/dP$ ,  $dv/dQ$  or  $dv/dtap$ . The other is for changes in voltage angle, namely  $dphi/dP$ ,  $dphi/dQ$  or  $dphi/dtap$ .

## 50.2.2.2 Advanced Tab

### Calculate all sensitivity variables

This option, which is provided for consistency with earlier versions of *PowerFactory*, is only available when calculations are for single elements (and without contingency analysis). It will be then selected by default and means that all sensitivity values will be available to show in the flexible data page regardless of the selection of variables to be recorded in the result file.

### 50.2.3 Advanced Options

The user will not see all the options described below at any one time: the options presented are dependent on the options selected on the Basic Options page.

#### 50.2.3.1 Busbar Tab

##### **Diagonal elements only**

This is a rather specific option which is only visible when just *Busbar* is selected on the *Basic Options* page.

It is used to calculate the sensitivities of the busbar by an injection on the busbar itself.

Sensitivities for loadings and losses on branches cannot be calculated with this option active.

##### **Power change**

This setting relates to Busbar sensitivity calculations. If *Each bus in turn* is selected, a separate sensitivity calculation for each specified terminal will be carried out. If *All buses simultaneously* is selected, the calculation assumes a power injection at all the specified terminals at the same time.

##### **Calculate regional sensitivities**

By checking the box of this option it is possible to determine the sensitivities of regions (Zones, Areas and Grids). This option is only available if *All buses simultaneously* is selected and also a region type element has been chosen as busbar input on the Basic Options page. An injection of 1 MW will be distributed over the buses in each region. The calculation of the regional sensitivities can be executed also considering contingencies.

##### **Calculate boundary sensitivity between adjacent regions**

The sensitivities of the boundary between regions will be determined. The boundary will be automatically internally created for the purpose of calculations. However, the boundary will not be stored in the database. The results can be obtained in the busbar sensitivity (PTDF/OTDF) report.

##### **Use fictitious border network**

This option is by default not selected and for most networks it is not needed. It is provided for use in networks where adjacent regions, normally representing adjacent countries, are not connected directly but via a so-called “fictitious border grid”. The graphical representation of the fictitious border network is shown in Figure 50.2.1. The fictitious border network has to be selected in the grid model. If there is no such network, then the box of this option can be left unchecked.

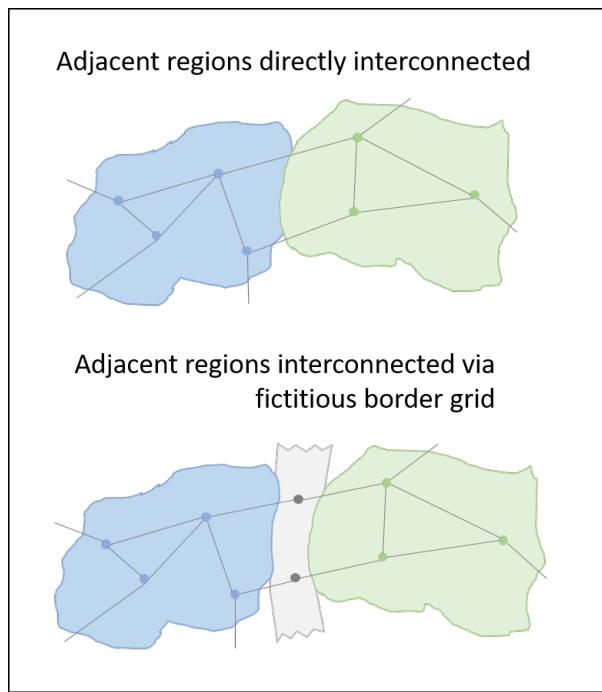


Figure 50.2.1: Network regions without and with fictitious border network

### **Injection based on generation shift key (GSK)**

In this case all generators will be taken into consideration for GSK. The GSK of generation units is given in percentage value, which can be entered on the Advanced tab of the Load Flow page of generation units: *ElmSym*, *ElmStat*, *ElmPvsys*, *ElmXnet* and *ElmAsm*.

#### **Consider generators with the setting 'Out of service when active power is zero'**

This option is only relevant (and visible) if the previous option (*Injection based on generation shift key (GSK)*) is selected. Its purpose is to allow different approaches as to which generators are to be regarded as out of service, therefore not considered for GSK. For normal load flow calculations, generators which have the *Out of service when active power is zero* setting selected and have an active power of zero are regarded as out of service. If the option is selected, which is the default, these particular generators will be considered for GSK alongside the in-service generators.

#### **50.2.3.2 Phase Shift / Tap Change**

##### **User-defined 3-/4-winding transformer control side**

This option becomes visible once one or more three- or four-winding transformers have been selected for sensitivity analysis. It allows the user to override the "side for tapping" setting on the individual elements with a global setting.

##### **Tap change**

This setting relates to transformer sensitivity calculations, i.e. the Phase Shift/Tap Change option. If *Each transformer in turn* is selected, a separate sensitivity calculation for each specified transformer or tap changer will be carried out. If *All transformers simultaneously* is selected, the calculation assumes that all tap at same time.

##### **Calculation method for transformer tap sensitivities**

There are two options for calculating transformer tap sensitivities:

**Linearisation of transformer tap changes** uses linearised load flow equations around the operating point to derive sensitivities to transformer tap positions.

**Discrete transformer tap assessment** actually solves the load flow twice, once at the current operating point and once when the tap position is changed by one tap up or down; the user specifies the direction. Then, the flows and voltages of the two load flow solutions are compared to deduce the sensitivity.

This method provides a more accurate assessment in cases when a strong dependence of the impedance on the current tap position is present, which, e.g., may result from a user-defined measurement report for the transformer.

For a DC calculation, the algorithm additionally checks whether the degree of dependence between the impedance and the current tap position is significant. If this is not the case the (faster) linearisation algorithm is used.

## 50.2.4 Output

On this page the user can configure the amount of information written to the output window, with “Short output” being the default.

- *Off* - Only the start and completion messages are seen.
- *Short output* - Information and warning messages from the sensitivities command will be seen, but no load flow details. If contingency analysis is included, the “short” contingency reporting will be output.
- *Detailed output* - This option displays more detailed information, including load flow and contingency analysis reporting as specified in the respective command dialogs.

## 50.2.5 Modal/Eigenvalue Analysis

On this page, the user can select the option for **Modal/Eigenvalue Analysis**, in order to execute an eigenvalue calculation on the sensitivity matrix. The number of eigenvalues to be calculated is defined in the *Number of Eigenvalues* field. The eigenvalues are always calculated in order of their largest magnitude, so selecting n eigenvalues will display the n eigenvalues in descending order according to magnitude (note that the calculation time increases with n).

In the *Display Results for Mode* field, the user can specify the number of a specific eigenvalue, for which the stability behaviour (i.e. the eigenvectors and participation factors) is to be analysed. The algorithm then additionally calculates the  $(\partial P / \partial Q)$ ,  $(\partial Q / \partial Q)$  (branch sensitivities) and the  $(\partial v / \partial Q)$ ,  $(\partial \varphi / \partial Q)$  (bus sensitivities) which correspond to the mode specified.

# 50.3 Reporting

The results of the *Sensitivities / Distribution Factors* calculation can be reported using the  icon. Below, the reporting command options are described, and Section 50.3.4 gives some information about the expected output.

## 50.3.1 General

On the General tab, the Result File can be selected. This is only necessary if more than one result file has been retained in the study case.

The **Use filter** setting allows the user to filter the elements to be shown in the report. If not selected, results for all relevant elements will be reported.

The remaining options on this page allow the user to select the individual reports to be generated. For standard quantities such as PTDF or PSDF, specific reports are provided. For other sensitivity values, the information will be found in the Busbar or Transformer reports.

These are the reports which are available:

- Busbar sensitivities
  - PTDF/OTDF ( $dP/dP$ )
  - $dP/dP$ ,  $dQ/dP$ ,  $dv/dP$ ,  $dphi/dP$ ,  $dPloss/dP$ ,  $dQloss/dP$
  - $dP/dQ$ ,  $dQ/dQ$ ,  $dv/dQ$ ,  $dphi/dQ$ ,  $dPloss/dQ$ ,  $dQloss/dQ$
- Transformer sensitivities
  - PSDF/TCDF ( $dP/dtap$ )
  - $dP/dtap$ ,  $dQ/dtap$ ,  $dphi/dtap$ ,  $dv/dtap$ ,  $dPloss/dtap$ ,  $dQloss/dtap$
- Sensitivities to HVDC(s)
- LODF values

The results of the variables  $dLoading/dP$  ( $c:dLoadingdP$ ) and  $dLoading/dtap$  ( $c:dLoadingdtap$ ) can be seen by accessing the Flexible Data Page for lines and transformers.

### 50.3.2 Thresholds

This tab allows the user to set the thresholds for the branch and terminals sensitivities to be included in the reports.

### 50.3.3 Used Format

This tab gives read access to the reporting formats, held in the System folder of the database.

### 50.3.4 Report output

A number of options are available within the reports themselves to customise what is shown, for example for which end (bus1 or bus2) of the lines the results should be shown.

The column titled with *Branch, Substation or Site* shows the parent object of the injection busbar (or phase-shift transformer).

If contingency analysis has been included in the calculation, it is possible to step through the results of the base case and each contingency case within the reports.

On the right-hand side of each report, filters are available to allow the user to filter by element class or specific elements so as to more easily view the results of interest.

## 50.4 Troubleshooting

If results are not obtained when running the calculation, or the results are not as expected, here is a short list of things to check:

- Consider the recording limits. Are they set appropriately?
- There are limits for filtering results within the reports. However, the user should bear in mind that only recorded results can be reported.
- If the result variables being recorded have been customised, some information may be lost; consider resetting them to the default set of variables.
- If the results are not as expected, check the options selected on the Advanced Options page, for example power change on each busbar in turn versus a simultaneous power change.

# Chapter 51

## Network Reduction

### 51.1 Introduction

This chapter explains how to use the *PowerFactory* Network Reduction tool. A typical application of Network Reduction is when a network that is part of or adjacent to a much larger network must be analysed, but cannot be studied independently of the larger network. In such cases, one option is to model both networks in detail for calculation purposes. However, there might be situations when it is not desirable to do studies with the complete model. For example, when the calculation times would increase significantly or when the data of the neighbouring network is confidential and cannot be published.

In these cases, it is common practice to provide a simplified representation of the neighbouring network that contains only the interface nodes (connection points). These can then be connected by equivalent impedances and voltage sources, so that the short circuit and load-flow response within the kept (non reduced) system is the same as when the detailed model is used.

*PowerFactory* offers two static methods for producing an equivalent representation of the reduced part of the network and calculating its parameters, valid for both load flow and short-circuit calculations, including asymmetrical faults such as single-phase faults. The first method is based on a Ward Equivalent representation and the second method is based on an REI (Radial-Equivalent-Independent) representation, which enables generators and/or loads to be retained and makes it possible to create power injections according to fuel type.

In addition, a reduction of groupings into regional equivalents for AC load flow is available.

The above methods result in networks suitable for the “static” load flow and short circuit calculations, but if (balanced) RMS simulations are to be carried out after reduction, a *Dynamic equivalent* option can be selected. This uses the REI reduction method based on the aggregation of coherent synchronous generation.

This chapter is separated into five parts. Firstly, the technical background of the *PowerFactory* Network Reduction algorithms are explained. Section 51.3 then discusses the steps needed to run a Network Reduction and Section 51.4 explains in detail each of the options of the *PowerFactory* Network Reduction tool. The penultimate part, Section 51.5, presents a simple example and the final section provides some *tips and tricks* to consider when working with the Network Reduction tool.

## 51.2 Technical Background

Some additional technical background on the Network Reduction tool is provided in the following sections.

### 51.2.1 Network Reduction using Ward method

#### 51.2.1.1 Network Reduction for Load Flow

*Network reduction for load flow* is an algorithm based on sensitivity matrices. The basic idea is that the sensitivities of the equivalent grid, measured at the connection points in the kept grid, must be equal to the sensitivities of the grid that has been reduced. This means that for a given (virtual) set of  $\Delta P$  and  $\Delta Q$  injections in the branches, from the kept grid to the grid to be reduced, the resulting  $\Delta u$  and  $\Delta \varphi$  (voltage magnitude and voltage phase angle variations) in the boundary nodes must be the same for the equivalent grid as those that would have been obtained for the original grid (within a user defined tolerance).

#### 51.2.1.2 Network Reduction for Short-Circuit

*Network reduction for short-circuit* is an algorithm based on nodal impedance / nodal admittance matrices. The basic idea is that the impedance matrix of the equivalent grid, measured at the connection points in the kept grid, must be equal to the impedance matrix of the grid to be reduced (for the rows and columns that correspond to the boundary nodes). This means that for a given (virtual) additional  $\Delta I$  injection (variation of current phasor) in the boundary branches, from the kept grid to the grid to be reduced, the resulting  $\Delta u$  (variations of voltage phasor) in the boundary nodes must be the same for the equivalent grid, as those that would have been obtained for the original grid (within a user defined tolerance).

This must be valid for positive sequence, negative sequence, and zero sequence cases, if these are to be considered in the calculation (unbalanced short-circuit equivalent).

### 51.2.2 Network Reduction using REI Method

The REI Equivalent is a methodology for network reduction which allows the flexibility to retain non-linear elements within the reduced area, or represent them with REI equivalent elements. It is possible to aggregate these reduced non-linear elements, with the option of grouping together generators of the same production (fuel) type. The advantages of the REI method are:

- Generators/loads of deleted nodes can be identified.
- Losses are kept at their initial value by using a Zero Power Balance Network.
- Electrical distances between boundary nodes and generation in the deleted network can be kept.
- The reduced networks can potentially be used with other static calculation modules besides load flow, such as contingency analysis and Optimum Power Flow.
- The ability to create equivalent injections per production type assists with system operators' obligations under European Network Codes.

### 51.2.3 Network Reduction using Regional Equivalents

The network reduction based on regional equivalents offers a non-deterministic approach where the load and generation of a region is concentrated to one node. This node is then connected to other nodes from reduced regions and to the retained network with artificial branch elements. The branches have all the same standard value for the impedance in the beginning. The final impedance value for each created connection is then determined by an optimisation with the **Parameter Identification** tool. The target is to minimise the deviations of interchange flows compared to the original case. The impedance optimisation can be executed for a **Load Flow** (one point in time) or a **Quasi-Dynamic Simulation** (time sweep).

The advantages of this method are:

- Reduction for different operating points and time sweep calculations.
- Fewer equivalent elements compared to other reduction methods.
- User definable target function for the optimisation.

### 51.2.4 Network Reduction for Dynamic Equivalent

The dynamic network reduction is based on the aggregation of coherent clusters of synchronous generators. In order to find the coherent clusters the network is excited, either by noise injection or user defined simulation events, and the machines are grouped based on their response. Then, the coherent machines are aggregated to an equivalent machine and obtained in the further reduction. The rest of the to be reduced network is assumed to be passive and therefore reduced with a static REI reduction. In the next step generic or user selectable controllers are added to the aggregated equivalent machines and an optional parameter identification is carried out to adjust the controller parameters to match the dynamic response of the reduced network to the pre-reduction behaviour.

Since this method is based on synchronous generation, dynamic equivalents can only be obtained if the system to be reduced contains synchronous generation. If there are no synchronous generators in the to-be-reduced system, a simplified static REI reduction is executed, where all network element are aggregated to static loads. In general the equivalent network structure obtained by the dynamic network reduction is only valid for balanced RMS-simulations and not for unbalanced RMS or EMT simulations.

## 51.3 How to Carry Out a Network Reduction

This section explains the process for running a Network Reduction. There are several steps that you must complete to successfully reduce a network:

1. Create a boundary or boundaries to define the *interior* and *exterior* regions.
2. Create a backup of the project intended for reduction (optional).
3. Activate the Additional Functions toolbar and configure the Network Reduction Tool options.
4. Run the Network Reduction Tool.

It is necessary to define a boundary or boundaries before proceeding further with the Network Reduction (except for the reduction with regional equivalents).

This process is described in detail in Chapter 15 Grouping Objects, Section 15.4 (Boundaries). However, to summarise, the boundary divides the network into two regions, the area to be reduced which is referred to as the *interior region* and the area to be kept which is referred to as the *exterior region*.

The following section describes the process of backing up the project, running the Network Reduction tool using the default options and describes the expected output of a successful network reduction.

For more information about the options available within the Network Reduction tool, see Section 51.4: Network Reduction Command.

### 51.3.1 How to Backup the Project (optional)

By default, the Network Reduction tool keeps all the original network data and the modifications needed to reduce the network are stored within a new expansion stage that is part of a new variation. It will only destroy the original data if the associated option within the command is configured for this (see Section 51.4.6: Outputs).

However, if you want extra security to guarantee against data loss, in case for instance you accidentally select the option to modify the original network, then you should make a backup copy of the project before completing the Network Reduction. There are three possible ways to do this:

- make a copy of the whole project and paste/store it with a name different to that of the original project; or
- export the project as a \*.pdf file (for information about exporting data refer to Section 9.1.5: Exporting and Importing of Projects); or
- activate the project and create a *Version* of the project. For information about Versions refer to Section 21.2 (Project Versions).

### 51.3.2 How to run the Network Reduction tool

This sub-section describes the procedure you must follow to run the Network Reduction using the default options. Proceed as follows:

1. Activate the base Study Case for the project you wish to reduce.
2. Define a boundary or boundaries that split the grid into the part to be reduced (interior region), and the part to be kept (exterior region). See Section 15.4 (Boundaries) for the procedure.
3. Open the boundary object(s) and use the **Check Split** button in the *ElmBoundary* dialog to check that the boundary correctly splits the network into two regions. See Section 15.4 (Boundaries) for more information about boundaries.
4. Select the *Change Toolbox* button ▾ from the main toolbar.
5. Press the *Network Reduction* icon  from the Additional Functions bar. This opens the dialog for *Network Reduction Command (ComRed)*.
6. Select the boundary/boundaries you previously defined using the button ▾.
7. Optional: If you wish to modify the settings of the command, do so in this dialog. The settings and options are explained in Section 51.4 (Network Reduction Command). However, the default options are recommended, unless you have a specific reason for changing them.
8. Press the **Execute** button to start the reduction procedure.

### 51.3.3 Expected Output of the Network Reduction

The default behaviour of the Network Reduction command is to create a Variation containing a single Expansion Stage called 'Reduction Stage'. For more information see Chapter 17: Network Variations and Expansion Stages. The Variation will be named automatically according to the reduction options selected in the Basic Options page of the Network Reduction command. For example, for the default options using the Ward Equivalent method, the Variation will be called *Equ-LF [EW] - Shc[sym] @*

*Boundary*, whereas if the REI method is used, the Variation will be called *Equ-LF [REI] @ Boundary*. Figure 51.3.1 shows an example of a network data model after a successful Network Reduction.

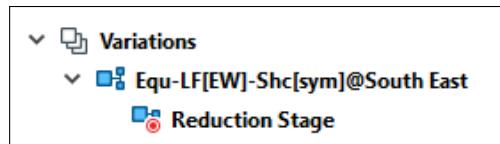


Figure 51.3.1: Variation and Reduction Stage after a successful Network Reduction using the default options.

The Network Reduction tool also creates a new Study Case with a name that matches the new Variation name. To return to your original network, all you need to do is activate the original study case that you used to initiate the Network Reduction.

---

**Note:** The Variation and Study Case created by the Network Reduction tool are automatically activated when the tool is run. To return to your original model you need to reactivate the 'base' Study Case.

---

### New objects added by the Network Reduction command

Depending on the network configuration and the options chosen within the Network Reduction command, during the Network Reduction process some new objects might be created.

In the case of the Ward Equivalent, there are two possible new object types:

- AC Voltage Source (*ElmVac*) ; and
- Common Impedance (*ElmZpu*) 

In the case of the REI method, depending upon the options selected, there may also be created one or more of these objects:

- Equivalent Node (*ElmTerm*)
- REI Load (*ElmLod*)
- REI Generator (*ElmSym*, *ElmGenstat*)
- REI SVS (*ElmSvs*)

For the Regional Equivalent the following elements are added:

- Nodes (*ElmTerm*)
- Loads (*ElmLod*)
- Generators (*ElmSym*, *ElmGenstat*)
- Transformers (*ElmTr2*)
- Common impedances (*ElmZpu*)

In addition to the elements in the REI reduction, the following elements can be added in a dynamic network reduction:

- Composite Models (*ElmComp*)
- DSL Models (*ElmDsl*)

- Measurements (*StaVmea*)

If the Ward Equivalent method is used, there will by default be one voltage source created for every boundary node and one common impedance between every pair of boundary nodes (unless the calculated mutual impedance is greater than the user-defined threshold described in Section 51.4.7). These objects are stored in the database but are not automatically drawn on the single line graphic. To insert graphical representations of the new elements substituting the reduced network, the *Diagram Layout Tool* may be used (see Section 12.6). For the REI method, the number of impedance objects will be greater because of the additional equivalent nodes, but the user-defined threshold still applies.

## 51.4 Network Reduction Command

In this section, the *Network Reduction* command options are explained.

### 51.4.1 Basic Options

#### 51.4.1.1 Reduction type and method

The first option on the Basic Options page is the choice of the **Reduction Type**, between dynamic equivalent and the static equivalent. The available **Methods** of the “Static equivalent” are the “Ward equivalent”, the “REI equivalent” and the “Regional equivalent”. Dependent on this selection the options on the following pages are offered accordingly.

#### 51.4.1.2 Boundary

This option specifies which part of the grid should be reduced. The user may select a single boundary or more than one; if more than one is selected, references to the selected boundaries are stored as a set in the study case. Boundaries used for network reduction must separate the original grid into two parts, the part that shall be reduced (external system) and the part that shall be kept (internal system) and therefore must be splitting boundaries.

The option “To be reduced” defines which side of the boundary or boundaries shall be reduced. Thus the interior region of the boundary doesn’t necessary equal the internal system for the network reduction.

If more than one boundary is used, there is a further requirement that they must not overlap, i.e. there must be no elements that are contained within more than one of the boundaries.

For more information about boundaries, refer to Section 15.4 (Boundaries).

The ability to select multiple boundaries is particularly beneficial when using the REI method and, for example, reducing many low-voltage grids. Depending upon the aggregation options, there is the possibility, with just one boundary, of ending up with a very large generator or load which could result in convergence difficulties. When multiple boundaries are used, the aggregation is done for each boundary, giving a more robust result.

#### 51.4.1.3 Regions

The selection of regions is offered instead of **Boundary** if the regional equivalent is selected. Grids, zones, areas and boundaries are supported for this reduction, but different element classes cannot be mixed for one reduction. Overlapping boundaries are not allowed.

#### 51.4.1.4 Load Flow

The load flow command is linked in the network reduction command. The load flow settings influence the outcome of the reduction and can be adapted by opening the load flow command via the offered pointer.

#### 51.4.1.5 Quasi-Dynamic Simulation

If the **Regional equivalent** option is selected the user has the possibility to select either the *Load Flow* or the *Quasi-Dynamic Simulation* as **Calculation type**. The according **Command** from the study case will be automatically linked. The settings from the linked command will be used for the reduction.

### 51.4.2 Ward Equivalent

**Load flow method** The Ward reduction can be executed based on a AC balanced or a DC load flow.

#### Calculate load flow equivalent

If this option is enabled, as it is by default, the load flow equivalent models will be created by the reduction tool. The AC balanced load flow method and the DC load flow method are supported for the Ward Equivalent reduction and the method can be selected directly in the network reduction dialog. When the option “Calculate load flow equivalent” is enabled, the study case Load Flow command can be accessed and there are further options to define the type of equivalent models to be created.

#### Equivalent Model for Power Injection

The load flow equivalent is composed of mutual impedances between boundary nodes and power injections (and shunt impedances) at boundary nodes. The power injection can be represented by different models. For the load flow equivalent there are three options (models) available:

- **Load Equivalent:** a load demand.
- **Ward Equivalent:** an AC voltage source which is configured as a Ward Equivalent.
- **Extended Ward Equivalent:** an AC voltage source which is configured as an Extended Ward Equivalent.

#### Calculate short-circuit equivalent

If this option is enabled, the short-circuit equivalent model will be created by the Network Reduction tool. Currently, only the *complete* short-circuit calculation method is supported.

#### Asymmetrical Representation

This option is used to specify whether an unbalanced short-circuit equivalent will be created. If this option is disabled, only a balanced short-circuit equivalent will be created, valid for the calculation of 3-phase short-circuits. If this option is enabled, an unbalanced short-circuit equivalent is created, valid for the calculation of single-phase and other unsymmetrical short-circuits. This means the network representation must include zero sequence and negative sequence parameters, otherwise the unbalanced calculation cannot be done.

### 51.4.3 REI Equivalent

#### 51.4.3.1 General

For a REI reduction, there are further options to specify which elements should be reduced:

### Reduction of non-linear elements

Using a set of drop-down menus, the user can select which elements are to be reduced and which are to be retained during the reduction process. In detail:

- Synchronous generators
  - Retain all: All synchronous generators will be retained.
  - Retain all voltage controlled: all PV generators will be retained; PQ generators will be reduced.
  - Reduce all (default): all synchronous generators will be reduced and replaced by REI equivalent elements.
- Static generators
  - Retain all: All static generators will be retained.
  - Retain all voltage controlled: all PV generators will be retained; PQ generators will be reduced.
  - Reduce all (default): all static generators will be reduced and replaced by REI equivalent elements.
- Loads
  - Retain all: All loads will be retained.
  - Reduce all (default): all loads will be reduced and replaced by REI equivalent elements.
- Static Var Systems
  - Retain all: All static Var systems will be retained.
  - Retain all voltage controlled: all voltage controlled SVS will be retained; others will be reduced.
  - Reduce all (default): all SVS will be reduced and replaced by REI equivalent elements.
- Additional Elements: The user can specify an element or a set of elements to be retained, such as important interchange lines between two countries.

### Calculate short-circuit equivalent

If this option is enabled, the short-circuit equivalent model will be created by the Network Reduction tool. Currently, only the *complete* short-circuit calculation method is supported.

### Asymmetrical Representation

This option is used to specify whether an unbalanced short-circuit equivalent will be created. If this option is disabled, only a balanced short-circuit equivalent will be created, valid for the calculation of 3-phase short-circuits. If this option is enabled, an unbalanced short-circuit equivalent is created, valid for the calculation of single-phase and other unsymmetrical short-circuits. This means the network representation must include zero sequence and negative sequence parameters, otherwise the unbalanced calculation cannot be done.

#### 51.4.3.2 Aggregation

##### Aggregation of nonlinear elements

All reduced elements can be simply aggregated together, or the generators and loads can be aggregated separately, although if short-circuit equivalents are to be calculated as well as load flow equivalents then it is necessary to keep the generators and loads separate.

Further sub-options are available:

###### All elements together

- Subgroup generators...
  - According to local controller

### Group loads and generators separately

- Subgroup generators...
    - According to local controller
- And/Or*
- According to model type and plant category
  - According to model type and plant category and subcategory
- Subgroup loads...
    - According to load classification

**Ignore active power flow direction:** This option is disabled by default. In this case, the generation or demand is grouped according to the sign of active power. If this option is activated, the positive and negative active power are aggregated. This option can be helpful if a large part of the surrounding grid is reduced and only a small area is retained. With deactivated option, the reduction could end up with two equivalent generators with large aggregated power, one of which is positive and the other negative. This can lead to non-convergence of the load flow.

---

**Note:** This option can lead to different results compared to the original network, because the impedances between the positive and negative injections are omitted.

---

**Single equivalent bus per substation:** This option is only designed for reductions in low voltage distribution networks and should not be used when reducing large network parts. If the option is used a single equivalent bus is created for each substation (*ElmSubstat* not *ElmTrfstat*) instead of individual busses for every group of aggregated REI elements. This helps to avoid loop flows due to very low impedances.

#### 51.4.3.3 Advanced

**Minimisation of equivalent branches (REI reduction)** This setting is only available in the REI reduction method and minimises the number of common impedances (*ElmZpu*) created in the reduction.

In each separated area of the “to be reduced system”, the equivalent network elements created in a REI reduction are interconnected by common impedances to all boundary nodes. Also every pair of boundary nodes of each separated area will be connected by a common impedance. For large systems with many boundary nodes this leads to a high number of equivalent branches created during a network reduction.

The setting “Minimisation of equivalent branches” on the “Advanced Options” page separates the “to be reduced system” into smaller subsystems by retaining some nodes. These subsystems will then be reduced separately. This leads to a significant decrease in the number of equivalent branches, but the number of equivalent loads and generators will increase.

The minimisation algorithm is configured with different sets of parameters and the user has 4 different selection possibilities

- **Off:** The minimisation is disabled. Output of the *PowerFactory* 2018 and older versions.
- **Fast minimisation:** A time optimised minimisation with large step sizes and a limited number of separated areas.
- **Standard minimisation:** Recommended settings for the minimisation.
- **Optimal minimisation:** Time intensive minimisation with small step sizes.

## 51.4.4 Regional Equivalent

### 51.4.4.1 Aggregation

The elements in the reduced regions are aggregated to one artificial node. The **Aggregation of nonlinear elements**-settings are identical to the ones in the REI-Reduction (51.4.3.2) and the user can select if generators and loads are supposed to be grouped separately and to be further distinguished

### 51.4.4.2 Impedance Identification

Equivalent impedances with a common **Initial value** are created between the nodes from the reduced regions and the retained network parts. The impedance identification is using the **Parameter Identification** in order to adapt the impedances to minimise the **Interchange mismatches**. The target function can be customised in the **Locations and weighting factors** table. The default is the consideration of the interchange of each reduced region with a weighting factor of 1. It is possible to add new locations (regions and all boundaries) and adapt the weighting factors, but it is recommended to add the retained regions and additional boundaries of importance with a higher weighting factor. The **Upper bound of equivalent impedance** is adding limits to the optimisation. The **Optimisation method** used by the parameter identification can be selected and certain **Settings** of the parameter identification can be modified.

### 51.4.4.3 Advanced

If needed, the **fictitious border network** can be selected for the reduction.

Phase shift transformers can be used in combination with the common impedances for each interconnection between the equivalent nodes and the retained network to avoid loop flows and thus achieve a better solution. The **Creation of phase shifters for loop flows** is optional. If the option is enabled, the default **Phase shifter type** can be adapted. The corresponding nominal voltage and the needed phase shift are set automatically.

## 51.4.5 Dynamic Equivalent

### 51.4.5.1 Coherency page

The settings on this page influence the clustering of elements into coherent groups of synchronous generators which will be aggregated in the reduction.

**Disturbance** In order to group the generators a balanced RMS simulation is carried out and the network needs to be excited with a disturbance in order to get a response that can later be used for grouping.

The first option are *Noise injections* at the boundary nodes. Temporary current sources are created to inject a random noise. This would reflect a wide range of disturbances with different excitation frequencies and therefore a more general grouping.

The network can also be exited for specific *Existing simulation events*. This would then result in a better tuned grouping for this exact excitation.

#### Identification method

The coherency between generators and their response can be evaluated in different ways. This has an influence on how many groups are determined. In general more groups/equivalents lead to a higher precision but less reduction.

*Based on correlation coefficients:* The response of a monitored signal can be used to obtain a correlation factor between two generators via a cross correlation. A correlation factor of 1 means that the machines are coherent and a factor of 0 means no correlation between the signals. A *correlation threshold* of 0.8 is usually a good value for the correlation in a dynamic network reduction. The grouping algorithm is starting with the largest machine which is not already in a group and adding all coherent generators (which are not in a group) to this new group, with respect to the correlation factor. This process is repeated until all generators are assigned to a group.

*Hierarchical / Agglomerative clustering:* Grouping the generators up to a specified threshold of the *Maximum distance / average distance*. The distance between two signals is calculated via the ward linkage and the average distance is the average distance of all generator pairs. The maximum distance is the maximum distance between any two generators in the same group.

The **Monitored signal** for the coherency identification can be selected from typical machine signals.

The **According to local controller** setting has the same influence as in the static REI reduction and is differentiating the machines depending on their local load flow controller. This may be important for load flow convergence.

Selective grouping **According to model type and plant category** is an option.

With the setting **Stop after coherency identification** the dynamic reduction can be stopped after this step and the output will only be the displayed coherent groups in the output window.

#### 51.4.5.2 Controllers page

**Create controllers for equivalent generators** The dynamic network reduction will create an equivalent machine for each coherent group of generators. Controller models for these machines can be added.

*Template:* There are several simplified power plant model templates available which can be selected (In the Configuration folder). With the *Copy and Select* option in the drop-down menu a template will be copied to the network reduction command and can be modified there. Also user defined templates can be copied to the network reduction command to be available in the dynamic reduction. A template contains a Plant Model (ElmComp) with a Synchronous machine and the needed DSL models, and the parameters to be optimised (IntIdentctrl) in the parameter optimisation for these machines.

**Parameter identification** The Parameter identification is optional and only available if controllers are added to the equivalent machines.

One of the available *Optimisation methods* from the parameter identification can be selected directly in the network reduction command. The default method for the parameter identification is the Particle Swarm Optimisation.

The *Simulation events* are taken from the study case.

*Settings* is a pointer to the parameter identification command. The allowed iteration number can be specified here. Depending on the settings the parameter identification can be very time consuming.

The *Parameters to be tuned* are taken from the controller templates and can be viewed and modified here.

#### 51.4.5.3 Advanced page

The **Minimisation of equivalent branches** is supported in the dynamic network reduction (see Section 51.4.3.3)

## 51.4.6 Outputs

The section describes the options available on the *Outputs* page of the Network Reduction command. These options define how the Network Reduction command modifies the network model.

---

**Note:** In Scripting there is an additional output possibility. It allows the user to reduce the system only in memory. Therefore the changes are only available during the runtime of the script. This is increasing the performance since the changes are not written to the database.

---

### 51.4.6.1 Calculation of Parameters Only

The equivalent parameters are calculated and reported to the output window. If this option is selected then the Network Reduction command does not modify the network model.

### 51.4.6.2 Create a new Variation for Reduced Network (Default)

The equivalent parameters are calculated and a Variation will be automatically created to store the reduced network model. If the project already includes System Stage(s) (from *PowerFactory* version 13.2 or earlier versions) then System Stage(s) will be created instead of a Variation.

### 51.4.6.3 Reduce Network without Creating a New Variation

The Network Reduction command will directly modify the main network model if this option is selected. Therefore, this option will destroy data by deleting the 'interior' region of the selected boundary, and replacing it with its reduced model, so this option should be used with care. To avoid losing the original grid data, backup the project as described in Section 51.3.1 (How to Backup the Project (optional)).

### 51.4.6.4 Clean up empty substations and bays

If this option is selected the empty substations and bays from the reduced systems will be removed by the network reduction. Also the network elements will be created in the grid and not in the former substations by the network reduction.

### 51.4.6.5 Show detailed output

Select this option in order to see detailed information about the objects that have been created as part of the network reduction.

## 51.4.7 Advanced Options

This section describes the Advanced Options for the Network Reduction command.

### 51.4.7.1 Mutual Impedance (Ignore above)

As part of the Network Reduction process equivalent branches (represented using Common Impedance elements) will be created between the boundary nodes, to maintain the power-flow relationship between

them. If such branches have a calculated impedance larger than this parameter they will be ignored (not added to the network model).

By default, the number of these branches created will be  $N*(N-1)/2$ , where N is the number of boundary nodes. A boundary node is defined for each boundary cubicle. Therefore, the number of created branches can be very high. Normally many of these equivalent branches have a very large impedance value, so their associated power flows are negligible and the branch can be ignored.

The default value for this parameter is 1000 p.u (based on 100 MVA).

#### 51.4.7.2 Calculate Equivalent Parameters at All Frequencies (only for static equivalents)

This option enables the calculation of frequency-related parameters. By default, the short-circuit equivalent parameters are calculated at all frequencies relevant to short-circuit analysis (equivalent frequencies for calculating the d.c. component of the short-circuit current):

- $f = f_n$
- $f/f_n = 0.4$
- $f/f_n = 0.27$
- $f/f_n = 0.15$
- $f/f_n = 0.092$
- $f/f_n = 0.055$

$f_n$  is the nominal frequency of the grid (usually 50 Hz or 60 Hz).

If only transient and sub-transient short-circuit currents are important in the reduced network, the calculation of frequency-related parameters can be skipped by unchecking this option.

#### 51.4.8 Verification

##### 51.4.8.1 Check Equivalent Results

If the option *Check load flow results after reduction* is enabled, the load flow results at the boundary nodes after the network reduction will be checked against the original network results. A warning message will be given if the results do not match (within the user defined *Threshold for check*).

The results of the comparison between the original network and the reduced network are printed to the output window.

The *Check simulation results after reduction* option is only available for the dynamic network reduction.

If the simulation results are checked, the *Generate curve comparison plot for selected signals* functionality is creating a new plot page with plots for the before and after reduction behaviour of selected signals.

For the regional reduction for a Quasi-dynamic simulation the variables for the comparison plots are automatically selected based on the selected regions for reduction and optimisation.

For the dynamic network reduction, on the **Advanced** page the *Signals to be checked* can be specified, either by a user defined variable selection (IntMon) or a auto selection of the first set of variables (IntMon) from the result file of the elements from the retained system.

### 51.4.8.2 Check Deviation of Operating Point

If the option *Save original operating point to results file* on the **Advanced page** is enabled, the base operating point for the Network Reduction will be automatically saved to two results files. These two created files are:

- LdfResultforNR.ElmRes: voltage magnitudes and angles of all boundary nodes; and
- ShcResultforNR.ElmRes: short-circuit level at all boundary nodes, including  $I''_k$  (Ikss),  $I'_k$  (Iks),  $i_p$  (ip),  $i_b$  (ib),  $I_b$  (lb),  $X_b/R_b$  ( $X_{toR_b}$ ), and  $X/R$  ( $X_{toR}$ ).

## 51.5 Network Reduction Example

This section presents a Network Reduction example using a small transmission network feeding a distribution system from “Bus 5” and “Bus 6” as shown in Figure 51.5.1 and represents a reduction with the Ward method.

The distribution system is represented by “Load A” and “Load B” and the corresponding two transformers. As a user you would like to study the distribution system in detail but are not concerned with the detailed power flow within the transmission system. Therefore, the Network Reduction tool can be used to create a equivalent model for the transmission system.

The interior region (the area that shall be reduced) is shown shaded in grey, whereas the non-shaded area is the exterior region that shall be kept. The procedure for completing the Network Reduction according to these parameters is as follows (you can repeat this example yourself using the *Nine-bus System* within the *PowerFactory Examples*):

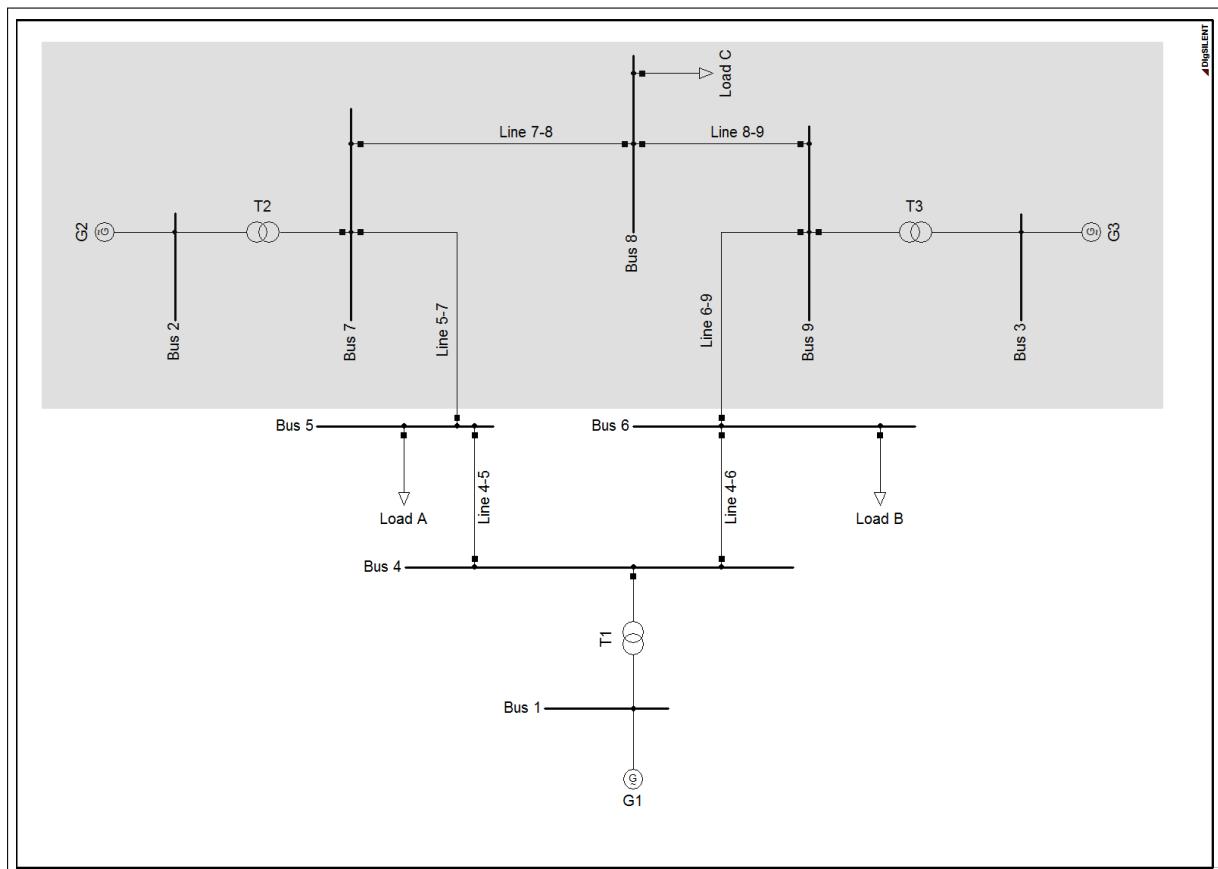


Figure 51.5.1: Example System with Original Network

1. Select the lines “Line 5-7” and “Line 6-9”.
2. Right-click on the selected lines and choose the option *Network Groupings* → *Boundary* → *New...*. The boundary dialog will appear.
3. Click on the **Mark Interior Region** button and verify that the region marked corresponds with the region showed grayout in Figure 51.5.1.
4. Open the Network Reduction command dialog and select newly created boundary using the select button (  ).
5. Press **Execute**. The Network Reduction tool will reduce the system.
6. Now you can draw the new common impedance and equivalent ward voltage source elements using the *Diagram Layout Tool*. The result of the Network Reduction is shown in Figure 51.5.2.

A load flow calculation or a short-circuit calculation in the reduced network gives the same results for the distribution network as for the original (non-reduced) network.

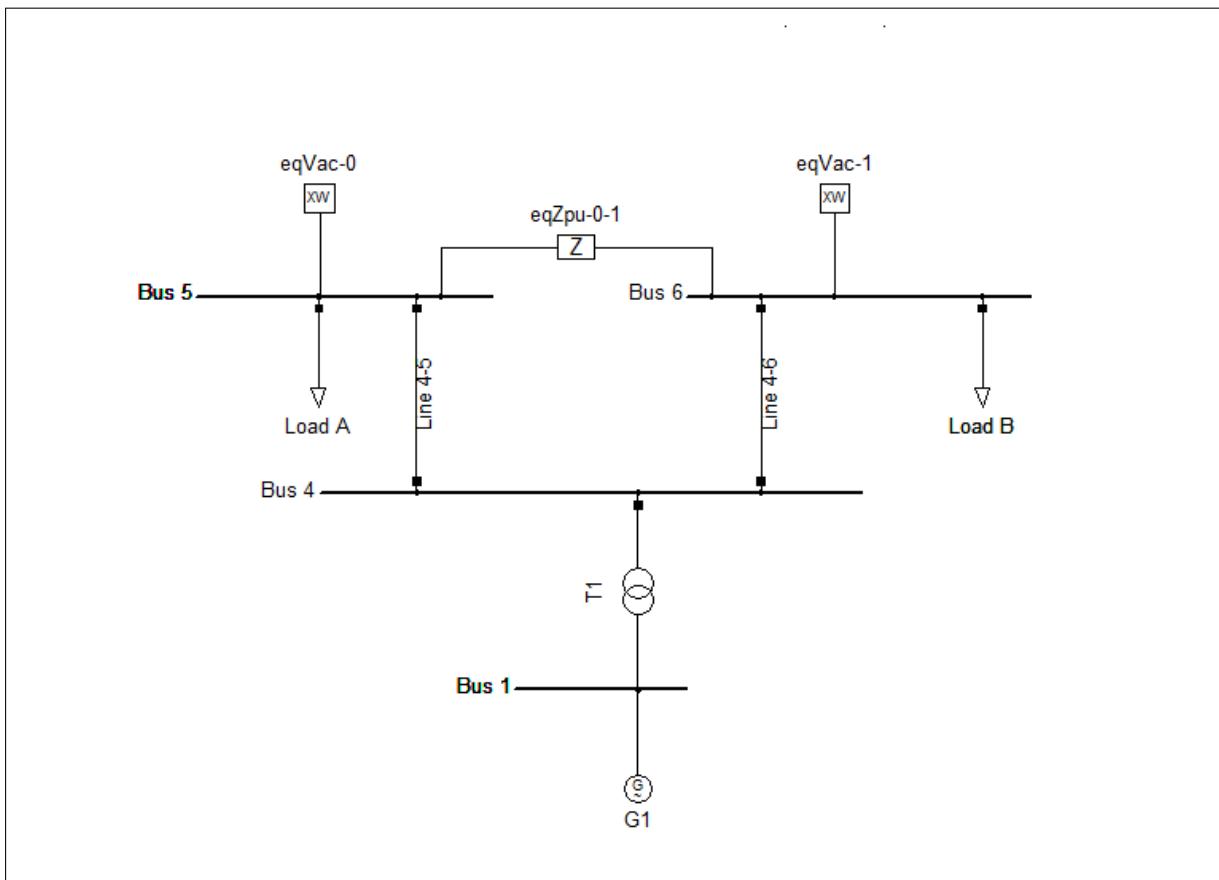


Figure 51.5.2: Example System with Reduced Network

## 51.6 Tips for using the Network Reduction Tool

This section presents some tips for using the Network Reduction tool and some solutions to common problems encountered by users.

### 51.6.1 Network Reduction doesn't Reduce Isolated Areas

By default, the boundary definition search stops when encountering an open breaker. This means that isolated areas can sometimes be excluded from the *interior* region and therefore are not reduced by the Network Reduction tool. The solution to this problem is to disable the boundary flag *Topological search: Stop at open breakers*. This option is enabled by default in all boundary definitions. It is recommended to disable it before attempting a Network Reduction.

A related problem occurs with the project setting (*Edit → Project → Project Settings → Advanced Calculation Parameters*) *Automatic Out of Service Detection*. It is recommended that this option is disabled before attempting a Network Reduction. However, it is disabled by default, so if you have not made changes to the default project settings you should not need to make any changes to this setting.

### 51.6.2 The Reference Machine is not Reduced

The Network Reduction tool will not reduce a reference machine defined within the interior region. It also leaves all network components that are topologically one bus removed from the reference machine (and of non-zero impedance). For example, if the reference machine is a typical synchronous machine connected to the HV system through a step up transformer, then the reduction tool will leave the synchronous machine, the LV bus, the step up transformer and the HV bus within the reduced network.

It is recommended that the reference machine is found within the exterior region before attempting a Network Reduction. The reference machine can be identified by checking the output window after a load-flow calculation.

# Chapter 52

## State Estimation

### 52.1 Introduction

The State Estimation (SE) function of *PowerFactory* provides consistent load flow results for an entire power system, based on measurements, manually entered data and the network model.

State estimation has traditionally been used for transmission systems, where many redundant measurements exist. Bad data from the measurement equipment should be identified and treated in order to determine the most feasible currents and voltages within the grid before running any further analysis, such as contingency analysis, security checks etc. In particular, it has been a task for grid operation and has only required a symmetrical (balanced) calculation.

However, due to the constantly increasing number of distributed generation units and new loads, the challenges at distribution level increase. At the same time, there are more and more measurement devices in distribution systems (e.g. advanced metering systems) available. This makes state estimation at distribution level possible. Due to the unbalanced load conditions in the distribution system, an unbalanced state estimation is therefore necessary. In addition to the classic use case, state estimation also becomes relevant for distribution system planning purposes in order to make realistic power assumptions.

*PowerFactory* supports both calculation methods: balanced within the positive sequence system and unbalanced in the 3-phase system.

The State Estimation requires measurements. The measurement types that are processed by the *PowerFactory* State Estimation are:

- Active Power Branch Flow
- Reactive Power Branch Flow
- Branch Current (Magnitude)
- Bus Bar Voltage (Magnitude)
- Breaker Status
- Transformer Tap Position

Unfortunately, these measurements are usually noisy and some data might even be totally wrong. On the other hand, there may be more data available than absolutely necessary and it is possible to improve the accuracy of the estimated network status through redundant measurements.

The states that can be estimated by the State Estimation on the base of the given measurements vary for different elements in the network:

- General, LV and MV Loads
  - (balanced or unbalanced) Active Power, and/or
  - (balanced or unbalanced) Reactive Power, or
  - (balanced or unbalanced) Scaling Factor, as an alternative
- Synchronous Machines
  - Active Power, and/or
  - Reactive Power
- Asynchronous Machines
  - Active Power
- Static var System
  - Reactive Power
- 2-,3- and 4-winding transformers
  - Tap Positions (for all but one taps)
- Static Generator and PV System
  - Active Power, and/or
  - Reactive Power
- Step-Voltage Regulator
  - (balanced or unbalanced) Tap Positions

## 52.2 Objective Function

The objective of a State Estimation is to assess the generator and load injections, and the tap positions in a way that the resulting load flow result matches as close as possible with the measured branch flows and bus bar voltages. Mathematically, this can be expressed with a weighted square sum of all deviations between calculated (`calVal`) and measured (`meaVal`) branch flows and bus bar voltages:

$$f(\vec{x}) = \sum_{i=1}^n \left( \sigma_i^{-1} \cdot \frac{\text{calVal}_i - \text{meaVal}_i}{\text{rating}} \right)^2 \quad (52.1)$$

where:

$$\sigma_i = (\text{accuracy}_i / 100)$$

The state vector  $\vec{x}$  contains all voltage magnitudes, voltage angles and also all variables to be estimated, such as active and reactive power injections at all bus bars.

Because more accurate measurements should have a higher influence to the final results than less accurate measurements, every measurement error is weighted with the corresponding standard deviation  $\sigma$ . This value should be based on the accuracy of the whole measurement configuration (general measurement device, transformer instruments, transmission channels, etc.).

In this setting, the goal of a State Estimation is to minimise the above given function  $f(\vec{x})$  under the side constraints that all load flow equations are fulfilled.

## 52.3 Components of the *PowerFactory* State Estimation

The State Estimation function in *PowerFactory* consists of several independent components, namely:

1. Preprocessing
2. Plausibility Check
3. Observability Analysis
4. Non-linear optimisation (state estimation)

Figure 52.3.1 illustrates the algorithmic interaction of the different components. The first *Preprocessing* phase adjusts all breaker and tap positions according to their measured signals.

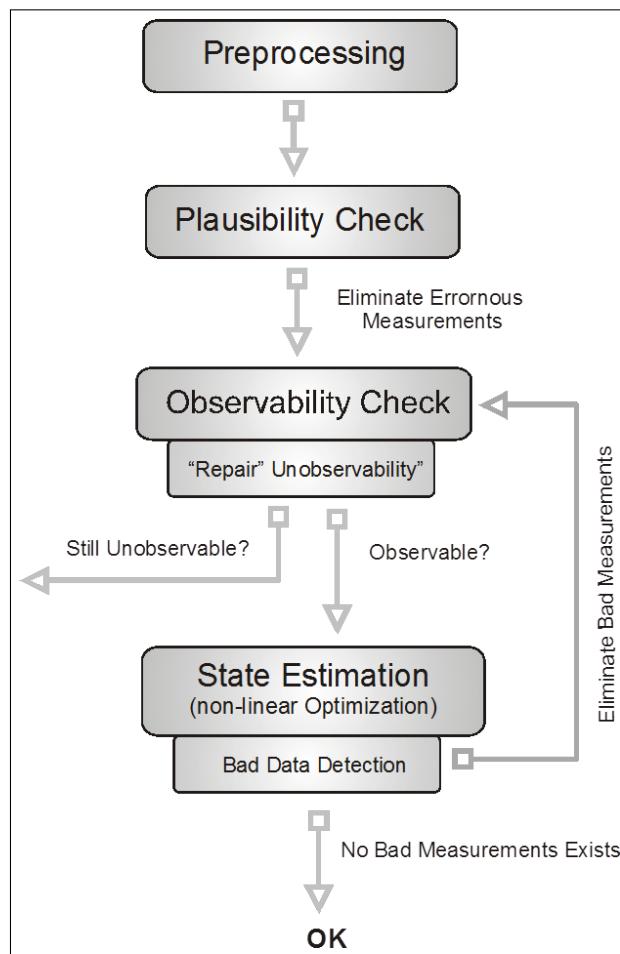


Figure 52.3.1: Variation of the *PowerFactory* State Estimation algorithm

The *Plausibility Check* is sought to detect and separate out, in a second phase, all measurements with some apparent error. *PowerFactory* provides various test criteria for that phase of the algorithm.

In a third phase, the network is checked for its *Observability*. Roughly speaking, a region of the network is called observable, if the measurements in the system provide enough (non-redundant) information to estimate the state of that part of the network.

Finally, the *State Estimation* itself evaluates the state of the entire power system by solving the above mentioned non-linear optimisation problem. *PowerFactory* provides various ways for coping with non-observable areas of the network.

In order to improve the quality of the result, observability analysis and state estimation can be run in a loop. In this mode, at the end of each state estimation, the measurement devices undergo a so-called “Bad Data Detection”: the error of every measurement device can be estimated by evaluating the difference between calculated and measured quantity. Extremely distorted measurements (i.e. the estimated error is much larger than the standard deviation of the measurement device) are not considered in the subsequent iterations. The process is repeated until no bad measurements are detected any more.

In the following, the distinct components of the *PowerFactory* State Estimation are explained in detail.

### 52.3.1 Plausibility Check

In order to avoid any heavy distortion of the estimated network-state due to completely wrong measurements, the following Plausibility Checks can be made before the actual State Estimation is started. Every measurement that fails in any of the listed Plausibility Checks will not be considered.

- Check for consistent active power flow directions at each side of the branch elements.
- Check for extremely large branch losses, which exceed their nominal values.
- Check for negative losses on passive branch elements.
- Check for large branch flows on open ended branch elements.
- Check whether the measured branch loadings exceed the nominal loading value of the branch elements.
- Node sum checks for both, active and reactive power.

Each test is based on a stochastic analysis which takes into account the measurement's individual accuracy. The strictness of the above mentioned checking criteria can be continuously adjusted in the advanced settings.

The result of the Plausibility Check is reported, for each measurement, on a detailed error status page (see Section [52.6](#)).

### 52.3.2 Observability Analysis

A necessary requirement for an observable system is that the number of available measurements is equal or larger than the number of estimated variables. This verification can easily be made at the beginning of every state estimation.

But it can also happen that only parts of the network are observable and some other parts of the system are not observable even if the total number of measurements is sufficient. Hence, it is not only important that there are enough measurements, but also that they are well distributed in the network.

Therefore, additional verifications are made checking for every load or generator injection whether it is observable or not. The entire network is said to be observable if all load or generator injections can be estimated based on the given measurements. *PowerFactory* does not only solve the decision problem whether the given system is observable or not: If a network is not observable, it is still useful to determine the islands in the network that are observable.

The Observability Analysis in *PowerFactory* is not purely based on topological arguments; it heavily takes into account the electrical quantities of the network. Mathematically speaking, the Observability Check is based on an intricate sensitivity analysis, involving fast matrix-rank-calculations, of the whole system.

The result of the Observability Analysis can be viewed using the Data Manager. Besides, *PowerFactory* offers a very flexible colour representation both for observable and unobservable areas, and for redundant and non-redundant measurements (see Section 52.6.4).

### Observability of individual states

The Observability Analysis not only identifies for each state (i.e., load or generator injections) whether it is observable or not. It also subdivides all unobservable states into so-called “equivalence-classes”. Each equivalence-class has the property that it is observable as a group, even though its members (i.e., the single states) cannot be observed. Each group then can be handled individually for the subsequent state estimation.

### Redundancy of measurements

Typically, an observable network is overdetermined in the sense that redundant measurements exist, which for the observability of the system do not provide any further information. During the Observability Analysis, *PowerFactory* determines redundant and non-redundant measurements. Moreover, it subdivides all redundant measurements according to their information content for the system's observability status. In this sense, *PowerFactory* is even able to calculate a redundancy level which then indicates how much reserve the network measurements provide. This helps the system analyst to precisely identify weakly measured areas in the network.

### Repair Unobservability

In case of unobservable states, *PowerFactory* has different treatment options to make a state estimation possible:

- Using internally generated pseudo-measurements.
- Using user-defined pseudo-measurements.
- Using default values from the load flow simulation.

A pseudo-measurement is basically a measurement with very poor accuracy. The treatment should set-up the model to force the non-linear optimisation algorithm to converge. All options are part of the command configuration and are explained in more detail in Section 52.5.1.

### 52.3.3 Non-linear optimisation (state estimation)

The non-linear optimisation is the core part of the State Estimation. As already mentioned in the introduction, the objective is to minimise the weighted square sum of all deviations between calculated and measured branch flows and bus bar voltages whilst fulfilling all load flow equations.

*PowerFactory* uses an extremely fast converging iterative approach to solve the problem based on Lagrange-Newton methods. If the Observability Analysis in the previous step indicates that the entire power system is observable, convergence (in general) is guaranteed.

Recall that the goal of the optimisation is to minimise the objective function  $f$  (i.e., the square sum of the weighted measurements' deviations) under the constraint that all load flow equations are fulfilled. Mathematically speaking, the aim is to find

$$\min f(\vec{x}) \tag{52.2}$$

under the constraint that

$$\vec{g}(\vec{x}) = \vec{0} \quad (52.3)$$

where  $\vec{g}$  is the set of load flow equations that need to be fulfilled. By the Lagrange-Newton method, we thus try to minimise the resulting Lagrange function

$$L(\vec{x}, \vec{\lambda}) = f(\vec{x}) + \vec{\lambda}^T \cdot \vec{g}(\vec{x}) \quad (52.4)$$

with the Lagrange multipliers  $\vec{\lambda}$ .

## 52.4 State Estimation Data Input

The main procedures to introduce and manipulate the State Estimation data are indicated in this section. For applying the *PowerFactory* State Estimation, the following data are required additional to standard load flow data:

- Measurements
  - Active Power Branch Flow
  - Reactive Power Branch Flow
  - Branch Current (Magnitude)
  - Bus Bar Voltage (Magnitude)
  - Breaker Status
  - Transformer Tap Position
- Estimated States
  - Loads: Active Power (P) and/or Reactive Power (Q), or the Scaling Factor, as an alternative.
  - Synchronous Machines: Active Power (P) and/or Reactive Power (Q)
  - Static Generator: Active Power (P) and/or Reactive Power (Q)
  - Asynchronous Machines: Active Power (P)
  - Static var Systems: Reactive Power (Q)
  - Transformers: Tap Positions

For the measurements listed above, *PowerFactory* uses the abbreviated names *P-measurement*, *Q-measurement*, *I-measurement*, *V-measurement*, *Breaker-measurement*, and *Tap position-measurement*. Similarly, as a convention, the four different types of estimated states are shortly called *P-state*, *Q-state*, *Scaling factor-state*, and *Tap position-state*.

### 52.4.1 Measurements

All measurements are defined by placing a so-called “External Measurement Device” inside a cubicle. For this purpose, select the device in the single line graphic and choose from the context menu (right mouse button) “New Devices” and then “External Measurements...” (see Figure 52.4.1). Then, the new object dialog opens with a predefined list of external measurements. Select the desired measurement device among this list.

The following measurement devices are currently supported

- (External) P-Measurement (StaExtpmea)

- (External) Q-Measurement (`StaExtqmea`)
- (External) I-Measurement, current magnitude (`StaExtimea`)
- (External) V-Measurement, voltage magnitude (`StaExtvmea`)
- (External) Breaker Signalisation Breaker Status (`StaExtbrkmea`)
- (External) Tap-Position Measurement Tap Position (`StaExttapmea`)

Any number of mutually distinct measurement devices can be defined in the cubicle.

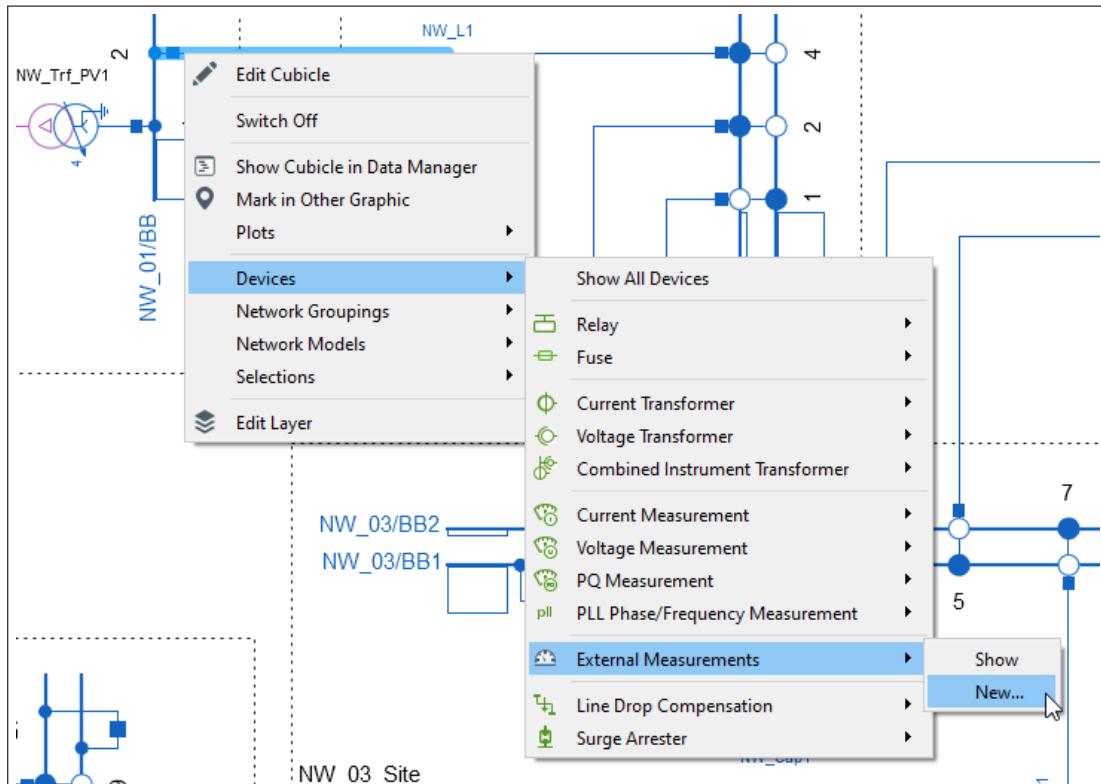


Figure 52.4.1: External Measurements that are located in a cubicle

### 52.4.1.1 Branch Flow Measurements

Any branch flow measurement (`StaExpmea`, `StaExtqmea`) is defined by the following values (see figures 52.4.2 and 52.4.4):

- Balanced/Unbalanced flag (`e:calcType`)
- Measured values (`e:Pmea` for balanced, `e:PmeaDP1`, `e:PmeaDP2` and `e:PmeaDP3` for unbalanced)
- Multiplicator (`e:Multip`)
- Orientation (`e:i_gen`)
- Accuracy class and rating (`e:Snom` and `e:accuracy`)
- Input status (to be found on the second page of the edit object, see Figure 52.4.4):  
E.g., tele-measured, manually entered, read/write protected,...(`e:iStatus`). It is important to note that the State Estimation takes into account only measurements, for which the “read”-Status is explicitly set and for which the “Neglected by SE”-Status is unset.

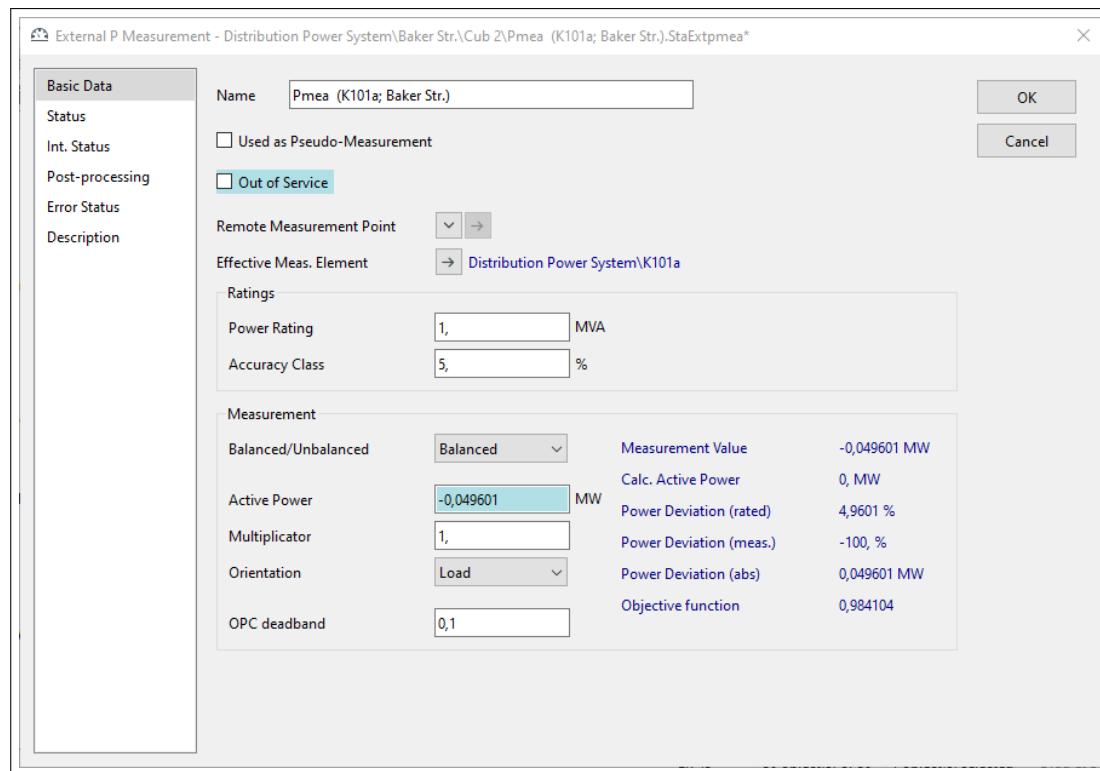


Figure 52.4.2: Dialog for an external P-measurement

The interface of the edit object changes whether the measurement is balanced or unbalanced.

For a balanced measurement, the sum of the power of all three phases is the measured value and the calculation information (e. g. *Power Deviation (rated)*) is displayed right next to them, see Figure 52.4.2.

For an unbalanced measurement, depending on the number of line conductors of the cubical, there are up to three input fields for the measurement values of all phases, see Figure 52.4.3. By ticking the appropriate boxes, it can be decided for each phase whether or not it should be considered as a measured value. The calculation information is shown in the separate tab *Result Overview*.

The accuracy class and the rating are used for weighting the measurement element. In case of redundant measurements, a more accurate measurement will be higher weighted than a less accurate measurement.

Using the flag *Orientation*, it is possible to define the meaning of the active or reactive power sign. Load orientation means that a positively measured P or Q flows into the element, generator orientation defines a positive flow as flowing out of an element. With the *Multiplicator*, a measured quantity can be re-rated. E.g., if a measurement instrument indicates 150kW (instead of 0.15MW), the *Multiplicator* can be set to 0.001 and the measured value is set to 150 resulting in a correct value.

It is important to note, that External P- and Q-measurements have the additional feature to possibly serve as a so-called (externally created) pseudo-measurement. This feature is activated by checking the corresponding box (e:pseudo). Pseudo-measurements are special measurements which are ignored during the regular calculation. They are activated in a selective manner only if the observability check found unobservable states in the network (see Section 52.5.1: Basic Setup Options for details).

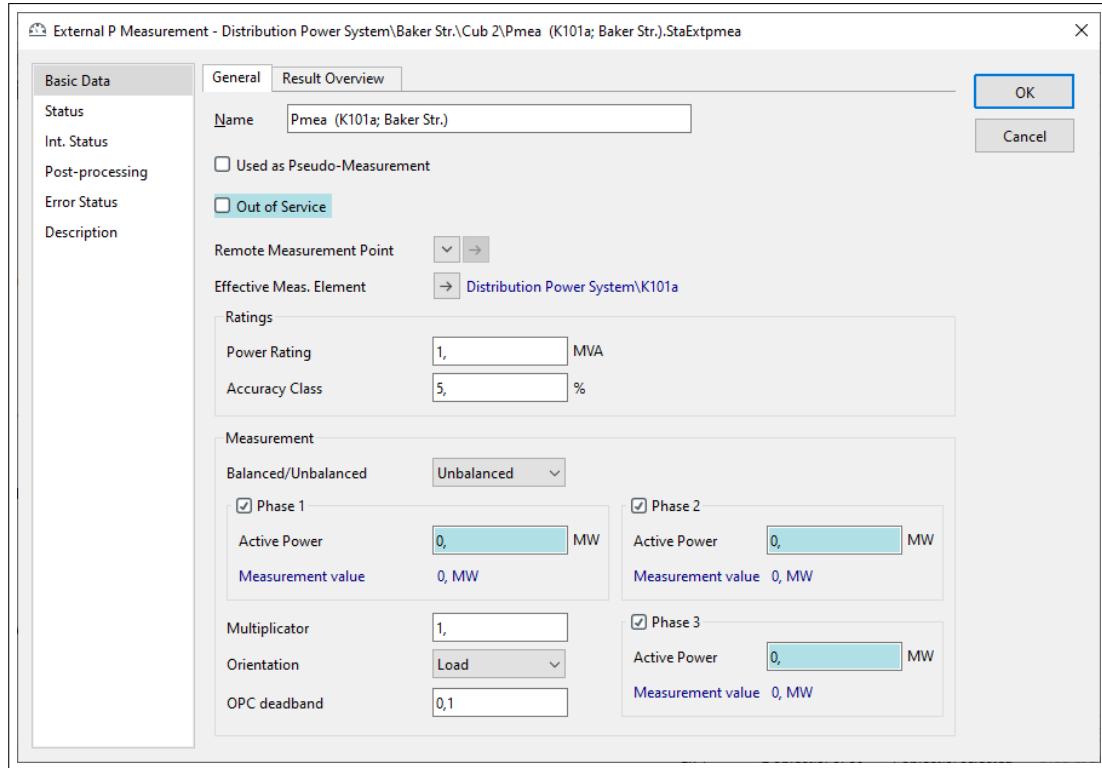


Figure 52.4.3: Dialog for an external P-measurement unbalanced

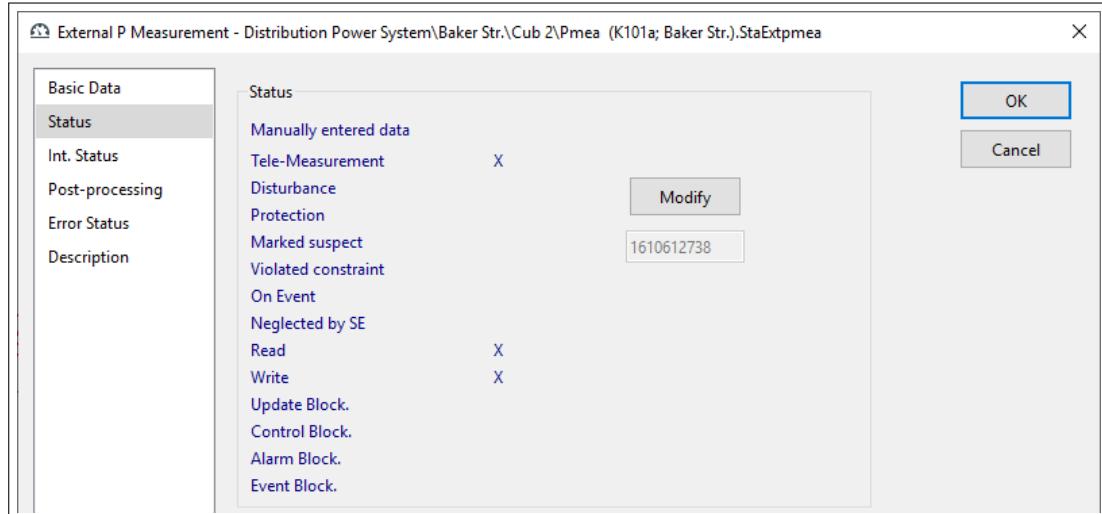


Figure 52.4.4: Status page of the dialog for an external P-measurement

### 52.4.1.2 Current Measurements

The External I-measurement (*StaExtmea*) plays a special role and slightly differs from the External P- and Q-measurements (see Figure 52.4.5): Besides specifying the measured current magnitude (*e:Imea*), the user is asked to enter an assumed (or measured) value for the power factor  $\cos\phi$  (*e:cosphi* and *e:pf\_recapr*).

If it is an unbalanced I-measurement, the current measurement value as well as the power factor need to be defined for each phase.

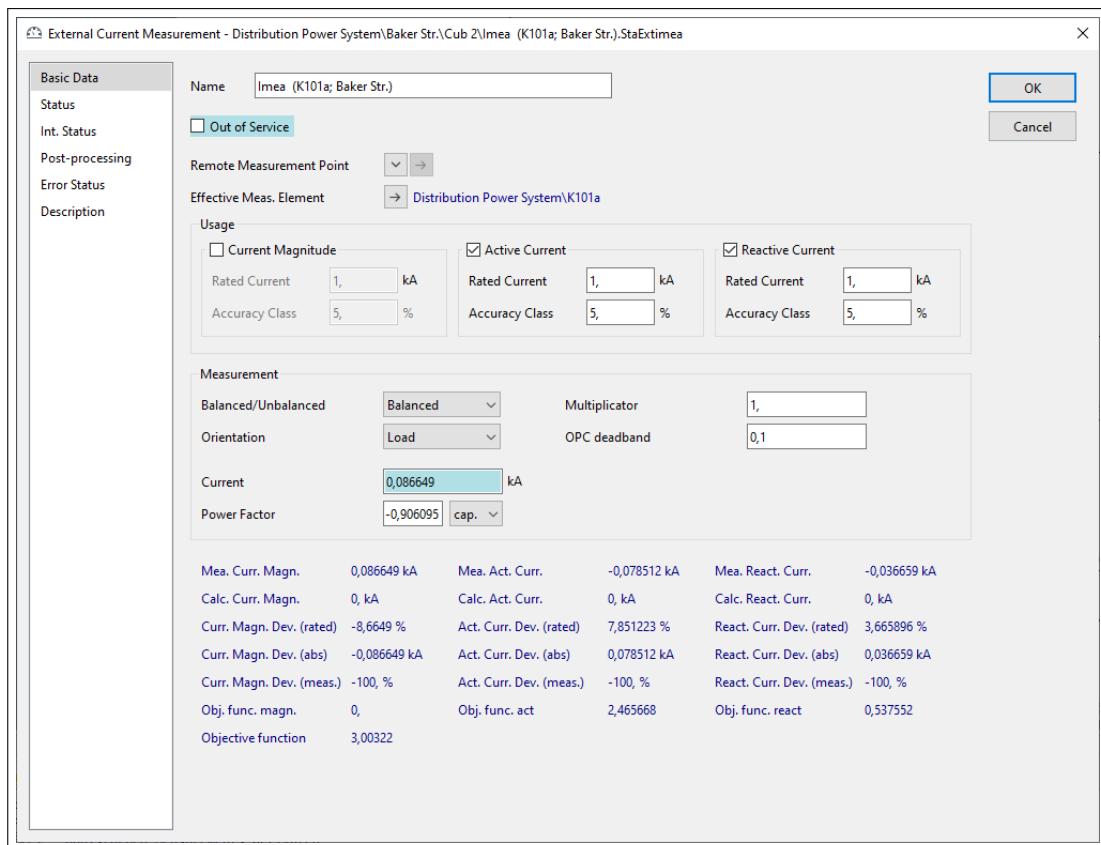


Figure 52.4.5: Dialog for an external I-measurement

Internally, the measured current magnitude is then additionally transformed into two further measurements, namely an active and a reactive current. This is due to the fact that current magnitude does not provide information on the direction of the flow, which on the other hand is essential to avoid ambiguous solutions in the optimisation.

In this sense, an external I-measurement may play the role of up to three measurements:

1. as a current magnitude measurement.
2. as a measurement for active current.
3. as a measurement for reactive current.

The decision which of these measurements shall participate in the State Estimation is left to the user by checking the boxes (`e:iUseMagn`, `e:iUseAct`, and/or `e:iUseReact`). In any case, the corresponding ratings for the used measurement types need to be specified. This is done (accordingly to the flow measurements) by entering the pairs of fields (`e:SnomMagn`, `e:accuracyMagn`), (`e:SnomAct`, `e:accuracyAct`), and (`e:SnomReact`, `e:accuracyReact`), respectively).

### 52.4.1.3 Voltage Measurements

Voltage measurements (`StaExvmea`) need to be placed in cubicles as well. By default the measurement point is located on the side of the edge element of the cubicle. This means if the edge element is disconnected from the terminal (open internal cubicle breaker), the calculated voltage will be at the edge element side of the cubicle. Vice versa if the option for *Measurement Point* is set to *Terminal connected to cubicle*, the calculated voltage will be on the terminal side.

A voltage measurement basically has the same properties as a flow measurement, except, for the

rating, only a single value for the accuracy needs to be specified. The corresponding internal reference is the nominal voltage of the terminal which serves as measurement point.

If the external voltage measurement object is defined as an unbalanced measurement, there will be an additional option for the voltage input format. The voltage measurement values can be entered either as line-to-ground or as line-to-line values.

#### 52.4.1.4 Breaker and Tap Position Measurements

Both breaker and tap position measurements are assumed to measure the corresponding discrete breaker status and tap position signal accurately. Hence, no ratings needs to be specified.

Tap position measurements have a conversion table as extra feature. The conversion table allows any discrete translation mapping between external tap positions (Ext. Tap) and tap positions used by *PowerFactory* (PF Tap).

#### 52.4.2 Editing the Element Data

In addition to the measurement values, the user has to specify which quantities shall be considered as “states to be estimated” by the SE. This must be done in the edit dialog of the corresponding element object on the *State Estimation* page (see Figure 52.4.6 left side). Note, if the corresponding check box(es) are disabled, the values for the conventional load flow calculation (i. e. settings from the *Load Flow* page, see Figure 52.4.6 right side) are used, even if an external measurement is defined.

It makes sense to also enter sensible default values on the load flow page, as these could possibly be used to create pseudo-measurements.

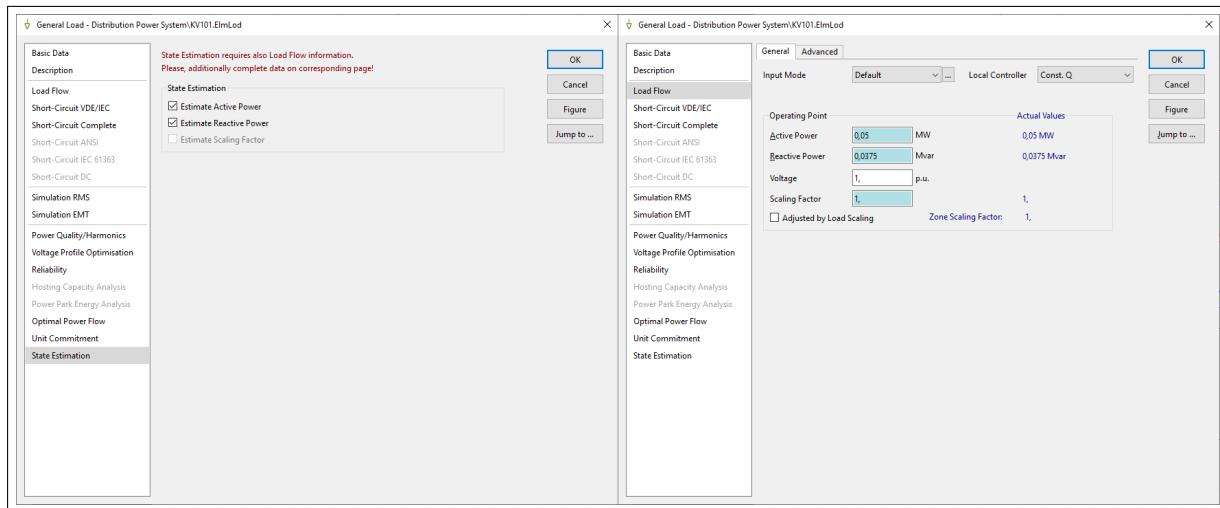


Figure 52.4.6: Element settings for SE set-up: left *State Estimation* page, right *Load Flow* page

The possible “states to be estimated” options differ from element to element. Possible states to be optimised whilst minimising the sum of the error squares over all measurements are all active and/or reactive power injections at generators and loads as well as all tap positions.

#### Loads

For each load (*ElmLod*, *ElmLodlv*, *ElmLodmv*), the user can specify whether its active and/or reactive power shall be estimated by the State Estimation (see Figure 52.4.6 left side). Alternatively, the State Estimation is able to estimate the scaling factor (for a given P and Q injection).

## Synchronous Machines, Static Generators, PV Systems

For synchronous machines (*ElmSym*), static generators (*ElmGenstat*) and PV systems (*ElmPvsys*), the active and reactive power may serve as a state to be estimated.

## Asynchronous Machines

For asynchronous machines (*ElmAsm*), the active power may serve as a state to be estimated.

## Static var Systems

For static var systems (*ElmSvs*), the reactive power may serve as a state to be estimated.

## Transformers

In the 2-winding transformer elements (*ElmTr2*), the tap position can be specified as a state to be estimated by the State Estimation. Tap positions will be estimated in a continuous way (without paying attention to the given tap limits).

For 3-winding transformers, any two of the three possible tap positions (HV-, MV-, and LV-side) can be selected for estimation.

For 4-winding transformers, any three of the four possible tap positions (HV-, LV-sides) can be selected for estimation.

## Step-Voltage Regulator

In the step-voltage regulator elements (*ElmVoltreg*), the tap position can be specified as a state to be estimated by the State Estimation. Tap positions will be estimated in a continuous way (without paying attention to the given tap limits).

# 52.5 Running SE

The following steps should be performed to execute the State Estimation:

- Start from a case where the conventional power flow converges successfully.
- Select *Additional Functions* from the *Change Toolbox* button (▼)
- Execute the SE by clicking the icon .
- Select the desired options for the State Estimation run (see below).
- Select **Execute**.

## 52.5.1 Basic Setup Options

Recall that the State Estimation in *PowerFactory* consists of three different parts (*Plausibility Check*, *Observability Check*, *Non-linear optimisation (state estimation)*) and an additional precedent *Preprocessing* step. This variation is reflected in the *Basic Options* dialog.

### 52.5.1.1 Calculation Method

Both calculation options *AC Load Flow, balanced, positive sequence* and *AC Load Flow, unbalanced, 3-phase (ABS)* are available. If *AC Load Flow, balanced, positive sequence* is selected, no unbalanced measurements will be used.

### 52.5.1.2 Preprocessing

The algorithm distinguishes between breaker- and tap position-measurements on the one hand, and P-, Q-, I-, and V-measurements on the other hand. Breaker- and tap position-measurements are handled in the preprocessing step, whereas the latter types are processed in the subsequent parts of the State Estimation.

#### Adapt breaker states according to measurements

If this checkbox is selected, all measured breakers statuses will be set to the corresponding measured signal values.

#### Adapt tap positions according to measurements

If this checkbox is selected, all measured tap positions will be set to the corresponding measured values.

### 52.5.1.3 Calculation components

The three components *Plausibility Check*, *Observability Check* and *Non-linear optimisation (state estimation)* can be set active separately, to consider the component in the process. The corresponding settings on the separate pages should be adjusted according to the measurement configuration.

### 52.5.1.4 Plausibility Check

The algorithm offers various kinds of plausibility checks to validate measurements. Each measurement undergoes the checks selected by the user. If a measurement fails any of the required tests, it will be marked as erroneous and will be neglected in all subsequent steps. A complete error report can be obtained via the error status page of each measurements (see Section 52.6).

The following checks can be enabled by marking the corresponding check boxes.

#### Consistent active power flow direction at each branch

Checks for each passive branch, whether all connected P-measurements comply with a consistent power flow direction. More precisely, if some flow out of a passive element is measured while, at the same time, no flow into the element is measured, then all P-measurements connected to this element fail this test. For this check, a P-measurement is said to measure a “non-zero” flow if the measurement value is beyond a value of  $\sigma \cdot \text{rating}$ , where  $\sigma$  and  $\text{rating}$  are the accuracy and the rating, respectively, of the measurement.

#### Branch losses exceed nominal values

Checks for each passive branch, whether the measured active power loss exceeds the nominal loss of the branch by a factor of  $1 + \varepsilon$ . This check only applies to passive branches which have P-measurements  $P_{\text{mea},1}, \dots, P_{\text{mea},r}$  in each of its  $r$  connection devices. The threshold  $\varepsilon$ , by which the nominal loss shall not be exceeded, is given by:  $\varepsilon = \sum_{i=1}^r \sigma_i \cdot \text{rating}_i$ , where  $\sigma_i$  and  $\text{rating}_i$  are the accuracy and the rating, respectively, of measurement  $P_{\text{mea},i}$ .

### Negative losses on passive branches

Checks for each passive branch, whether the measured active power loss is negative, i.e., if a passive branch is measured to generate active power. This check only applies to passive branches which have P-measurements  $P_{\text{mea},1}, \dots, P_{\text{mea},r}$  in each of its r connection devices. The measured power loss of the branch is said to be negative if it is below the threshold  $(-\sum_{i=1}^r \sigma_i \cdot \text{rating}_i)$ .

### Large branch flows on open ended branches

Checks for each connection of the element, whether the connection is an open end (i.e., switch is open, or it is connected to only open detailed switches). If the connection is open and there exists a (P-, Q-, or I-) measurement which measures a “non-zero” flow, then the corresponding measurement fails the test. Again, a measurement is said to measure a “non-zero” flow if the measurement value is beyond a value of  $\sigma \cdot \text{rating}$ .

### Branch loadings exceed nominal values

Checks for each connection of the element, if the measured complex power (which is computed by the corresponding P- and/or Q-measurements) exceeds the rated complex power value by a factor of  $1 + s$ . Here, s is the accuracy of the P- and/or Q-measurement(s).

### Node sum checks for active and reactive power

This check applies to P- and/or Q-measurements. Checks, for each node of the network, if the node sum of the measured values in the adjacent branches is zero. If this is not the case, i.e., if the P- and/or Q-sum exceeds a certain threshold value, all adjacent P- and/or Q-measurements fail the test. Again, “not being zero” means that the sum of the measured values of the adjacent P-measurements  $P_{\text{mea},1}, \dots, P_{\text{mea},r}$  has magnitude below the threshold  $\sum_{i=1}^r \sigma_i \cdot \text{rating}$  (similarly for Q-measurements).

Each Plausibility Check allows for an individual strictness setting. Note that all checks rely on the same principle: namely, the given measurement values are checked against some threshold. Recall, for example, that the “node sum check for P” tests whether the active power sum at a node is below a threshold of  $\varepsilon = \sum_{i=1}^r \sigma_i \cdot \text{rating}$ . The user has the possibility to influence the strictness of this threshold. Therefore, the settings provide to enter so-called “exceeding factors”  $\text{fac} > 0$  such that the new threshold is  $\text{fac} \cdot \varepsilon$  instead of  $\varepsilon$ . E.g., in the case of the node sum check for P, the user may define the corresponding factor `fac_ndSumP`.

The higher the exceeding factor, the less strict the plausibility test will be. Similar exceeding factors can be specified for any of the given tests.

#### 52.5.1.5 Observability Analysis

The Observability Analysis is an optional component of the State Estimation. If activated, it checks whether the specified network is observable, i.e., whether the remaining valid P-, Q-, V-, and I-measurements (which successfully passed the plausibility checks) suffice to estimate the selected P-, Q-, Scaling Factor-, and Tap position-states. In addition, the Observability Analysis detects redundant measurements. Redundancy, in general, yields more accurate results for the following state estimation.

Moreover, if the Observability Analysis detects non-observable states, upon user selection, it tries to fix this unobservability by introducing further pseudo-measurements.

### Rastering of sensitivity matrix

Internally, the Observability Check is based on a thorough sensitivity analysis of the network. For that purpose, the algorithm computes a sensitivity matrix that takes into account all measurements, on the one hand, and all estimated states on the other hand. This sensitivity matrix is discretised by rastering the continuous values.

The user can specify the precision of this process by defining the number of intervals into which the values of the sensitivity matrix shall be rastered (`SensMatNoOfInt`), the threshold below which a continuous value is considered to be a 0 (`SensMatThresh`) in the discrete case, and the mode of rastering (`iOpt_raster`). It is highly recommended to use the predefined values here.

### Min. effect w.r.t. states to estimate

A measured value from a measuring device needs at least the following sensitivity to a state, that it can be used for the observability test. The higher the value, the more likely it is that the state will be identified as unobservable.

- *Powerflow mea.:* power to power or power to tap position sensitivity
- *Voltage mea.:* voltage in % to power or voltage in % to tap position sensitivity

There are applications in which it can make sense to increase the sensitivities in order to trigger the “Treatment of unobservable areas” for certain states. For example, in the case of well-tuned predefined pseudo-measurements.

### Treatment of unobservable areas

In case of unobservable states, the user has different options to cope with the situation:

- **Stop if unobservable regions exist:** The algorithm terminates with the detection of unobservable states. The Observability Analysis groups all non-observable states into different “equivalence classes”. Each equivalence class consists of states that carry the same observability information through the given measurements. In other words, the given measurements can only distinguish between different equivalence classes, but not between various states of a single equivalence class. The results can be viewed by the user (see Section 52.6 Results).
- **Use P-, Q-values as specified by model:** If this option is selected, the algorithm internally drops the “to be estimated” flag of each non-observable state and uses the element specifications of the load flow settings instead (see Figure 52.4.6). For example, if a P-state of a load is unobservable, the algorithm will use the P-value as entered on the load flow page. Hence, the network is made observable by reducing the number of control variables.
- **Use predefined pseudo-measurements:** Using this option, the algorithm “repairs” the unobservability of the network by increasing the degrees of freedom. For that purpose, at the location of each non-observable state, the algorithm tries to activate a pseudo-measurement of the same kind. Hence, if a P-(Q)-state is non-observable in some element, the algorithm searches for a predefined P-(Q)-pseudo-measurement in the cubicle of the element carrying the non-observable state. A predefined P-(Q)-pseudo-measurement is set-up enabling the flag *Used as Pseudo-Measurement* in the external measurement object (see Figure 52.4.2). In case of a non-observable scaling-factor both, a P- and a Q-pseudo-measurement are required. The introduced pseudo-measurements remain active as long as needed to circumvent unobservable areas.
- **Use internally created pseudo-measurements:** This option is similar to the previous one, except the algorithm automatically creates and activates a sufficient number of internal pseudo-measurements to guarantee observability. More precisely, internal pseudo-measurements are created at the locations of all elements that have non-observable P-(Q-, scaling factor-)state. For each such element, the pseudo-measurement value for P (Q, P and Q) is taken from the element’s load flow specification (see Figure 52.4.6 right side). All internally created pseudo-measurements use a common setting for their rating and accuracy, which can be specified on the advanced setup options page for the observability check.
- **Use predefined and internally created meas:** This mode can be considered as a mixture of the latter two options. Here, in case of a non-observable state, the algorithm tries to activate a predefined pseudo-measurement of the same kind. If no corresponding pseudo-measurement has been defined, then the algorithm automatically creates an internal pseudo-measurement.

### Settings for internally created pseudo-measurements

If, on the basic option page, the mode for the treatment of unobservable regions is set to “use only internally created pseudo-measurements” or to “use predefined and internally created pseudo - measurements”, the user may specify a default power rating (`snomPseudo`) and a default accuracy class (accuracy `Pseudo`). These default values are used for all automatically created internal pseudo-measurements.

### 52.5.2 Bad Data Detection

Recall that the State Estimation loops Observability Analysis and State Estimation as long as no further bad measurement is found (see Figure 52.3.1). The following settings allow the user to control the number of iterations performed by the loop.

#### Maximum number of measurements to eliminate

The variable `iBadMeasLimit` specifies an upper limit on the number of bad measurements that will be eliminated in the course of the State Estimation.

#### Tolerance factors for bad measurement elimination

A measurement is declared to be bad, if the deviation of measured against calculated value exceeds the measurement's accuracy, i.e., if

$$\frac{\text{calcVal} - \text{meaVal}}{\text{rating}} \geq \frac{\text{accuracy}}{100} \quad (52.5)$$

where `calVal` and `meaVal` are the calculated value and the measured value, respectively. The user may modify this definition by adjusting tolerance factors for bad measurements. More precisely, a measurement is declared to be bad, if the left-hand side in equation (52.5) exceeds `facErr·accuracy/100`. Here  $\text{facErr} > 0$  is a factor which can be specified by the user for each group of measurements individually. Use the factors `facErrP`, `facErrQ`, `facErrV`, `facErrIMagn`, `facErrIAct`, and `facErrIReact` for P-, Q-, V-measurements, and the three types of the I-measurements (magnitude measure, active current measure, reactive current measure).

### 52.5.3 Non-Linear optimisation (state estimation)

The non-linear optimisation is the central component of the State Estimation. The underlying numerical algorithm to minimise the overall measurement error is the iterative Lagrange-Newton method.

The algorithm stops successfully if the following three criteria are fulfilled:

1. The maximum number of iterations has not yet been reached.
2. All load flow constraint equations  $\vec{g}(\vec{x}) = 0$  are fulfilled to a predefined degree of exactness, which means:
  - (a) all nodal equations are fulfilled.
  - (b) all model equations are fulfilled.
3. The Lagrange function  $L(\vec{x}, \vec{\lambda})$  itself converges. This can be achieved if
  - (a) either the objective function itself converges to a stationary point, or
  - (b) the gradient of the objective function converges to zero.

The following settings serve to adjust these stopping criteria and the initialisation of the process. If the user is unfamiliar with the underlying optimisation algorithm, he is urged to use the default settings here.

### 52.5.3.1 General

#### Solver

The SE offers two solver options to run the Lagrange-Newton method: the build in *Standard* solver and the open source *Ipopt* solver.

#### Convergence of objective function

The user is asked to choose among the following two convergence criteria for the Lagrangian function: Either the function itself is required to converge to a stationary point, or the gradient of the Lagrangian is expected to converge.

- *Based on objective function*: if this option is selected, the user is asked to enter a value for the *Max. change of objective function*. If the change in value between two consecutive iterations falls below this value, the Lagrangian is considered to have converged. The *Max. change of objective function* is the value (in %), below which the Lagrangian is considered to have converged.
- *Based on gradient of objective function*: if this option is selected, the user is asked to enter a value for the *Max. value for gradient of objective function*. If the gradient falls below this value, the Lagrangian is considered to have converged. The *Max. value for gradient of objective function* is the absolute value, below which the Lagrangian is considered to have converged.

It is strongly recommended due to mathematical precision to use the criterion on the gradient. The other option might only be of advantage if the underlying Jacobian matrix behaves numerically unstable which then typically results in a “toggling” of the convergence process in the last iterations.

#### Max. number of iterations

The user is asked to enter the maximum number of iterations.

#### Max.acceptable error for load flow equations

The user should enter a maximum error for nodal equations (where the deviation is measured in kVA), and in addition, a maximally tolerable error for the model equations (in %).

The errors can be entered separately for the high, medium and low voltage level. The thresholds of these levels can be defined in the project settings.

### 52.5.3.2 Initialisation

The non-linear optimisation requires an initialisation step to generate an initial starting configuration.

#### Load Flow

Specifies the settings of the load flow command which is taken for initialisation in case no flat start is used.

#### Initialisation of non-linear optimisation

The user may specify whether the initialisation shall be performed by a load flow calculation or by a flat start. If it is known in advance that the final solution of the optimisation part is close to a valid load flow solution, initialising by a load flow calculation pays off in a faster convergence.

### 52.5.3.3 Advanced

#### Handling of unsuccessful optimisation

The user can specify what to do with the results of unsuccessful optimisation calculations. Instead of discarding the results in this case, the user can also keep a suboptimal result with the appropriate option. This avoids the standard non-convergence pop-up message.

When using the *Standard* solver, the user can additionally decide whether or not to fall back to the standard method if the algorithm diverges using the speed-up iteration mode. Disabling the function can make sense if the *Best found feasible results* option has been selected for the result output, see Section 52.5.4. This selection is activated by default.

## 52.5.4 Results/Output

#### Output mode

Three different levels of output during the iterative process can be selected. With the option *Detailed* more options are available to provide a customised output.

#### Output options

The main objective of the output is to provide some information about the performance of the SE. A huge amount of identified bad data, high nodal and/or high model equation errors as well as a long iteration process are indicators for a bad measurement configuration set-up.

The output window report is explained in more detail in Section 52.6.1.

#### Shown results

There are several options as to which results should be kept and shown for the calculation. The default is to keep the last result, but only if the algorithm converges. Especially in the case of a non-convergent calculation, one of the other options could be selected to identify problem areas based on results.

## 52.6 Results

The presentation of the State Estimation results is integrated into the user interface. The solution of the non-linear optimisation in the State Estimation is available via the complete set of variables of the conventional Load Flow calculations. It can be seen in the single line diagram of the grid or through the browser.

### 52.6.1 Output Window Report

The *PowerFactory* State Estimation reports the main steps of the algorithm in the output window.

For the Plausibility Checks, this implies the information on how many models failed the corresponding checks. For the Observability Analysis, the report contains the information on how many states were determined to be observable, and in addition how many measurements were considered to be relevant for observing these states.

Non-linear optimisation reports, in each iteration step, the following figures:

- The current error of the constraint nodal equations (in VA) (*Error Nodes*).

- The current error of the constraint model equations (`Error ModelEqu`).
- The current value of the gradient of the Lagrangian function (`Gradient LagrFunc`).
- The current value of the Lagrangian function (`LagrFunc`)
- The current value of the objective function f to be minimised (`ObjFunc`).

## 52.6.2 External Measurements

### 52.6.2.1 Deviations

Each branch flow measurement (`StaExtPmea`, `StaExtQmea`) and each voltage measurement (`StaExtVmea`) offers parameters to view its individual deviation between measured value and computed value by the State Estimation. The corresponding variables are:

- `e:Xmea`: measured value as entered in `StaEx*mea`
- `e:cMeaVal`: measured value (including multiplicator)
- `e:Xcal`: calculated value
- `e:Xdif`: deviation in % (based on given rating as reference value)
- `e:Xdif_mea`: deviation in % (based on the measured value as reference value)
- `e:Xdif_abs`: absolute deviation in the measurement's unit

Here X is a placeholder for P, Q, or U in the case of a P-, Q-, or V-measurement. These results will also be available in the edit object of the corresponding external measurements, see for example Figure 52.4.2).

If the unbalanced measurement is selected, these results will be available for each active phase. In the edit object there will be a separate slider on the *Basic Data* called *Result Overview*.

Recall that a `StaExtImea` plays a special role, since a current measurement may serve as up to three measurements (for magnitude, for active current, and/or for reactive current). Hence, a current measurement has the above listed variables (with X being replaced by I) for each of the three measurement types. In order to distinguish between the three types, for a `StaExtImea`, the variables carry the suffixes `Magn` (for magnitude measurement), `Act` (for active current measurement), and `React` (for reactive current measurement).

### 52.6.2.2 Error Status

All measurements (`StaExt*meas`) which possibly participate in the Plausibility Checks, the Observability Analysis, or the State Estimation provide a detailed error description page (see Figure 52.6.1) with the following information:

- General Errors:
  - Is unneeded pseudo-measurement (`e:errUnneededPseudo`)
  - Its input status disallows calculation, i.e., input status does not allow "Read" or is already marked as "Wrong Measurement" (`e:errStatus`)
  - Measurement is out of service (`e:errOutOfService`)
- Plausibility Check Errors:
  - Fails test: Consistent active power flow direction at each side of branch (`e:errConsDir`)
  - Fails test: Large branch losses (`e:errExcNomLoss`)

- Fails test: Negative losses on passive branches  
(e:errNegLoss)
- Fails test: Large branch flows on open ended branches  
(e:errFlwIfOpn)
- Fails test: Branch loadings exceed nominal values  
(e:errExcNomLoading)
- Fails test: Node sum check for P (e:errNdSumP)
- Fails test: Node sum check for Q (e:errNdSumQ)
- Observability Analysis Errors:
  - Measurement is considered to be redundant for observability of the network, i.e., observability is already guaranteed even without this measurement. Nevertheless redundant measurements are used in the non-linear optimisation since, in general, they help to improve the result (e:errRedundant).
  - For redundant measurements, also the redundancy level is indicated on this page (e:RedundanceLevel). The higher the redundancy level, the more measurements with a similar information content for the observability analysis exist.
- State Estimation Errors:
  - Measurement is detected to be bad, has been removed and was not considered in last non-linear optimisation loop (e:errBadData)

This detailed error description is encoded in the single parameter e:error that can be found on the top of the error status page. Again, we have the convention that, for a *StaExtmea*, the variables e:errRedundant, e:RedundanceLevel and e:errBadData carry the suffixes Magn (for magnitude measurement), Act (for active current measurement), and React (for reactive current measurement).

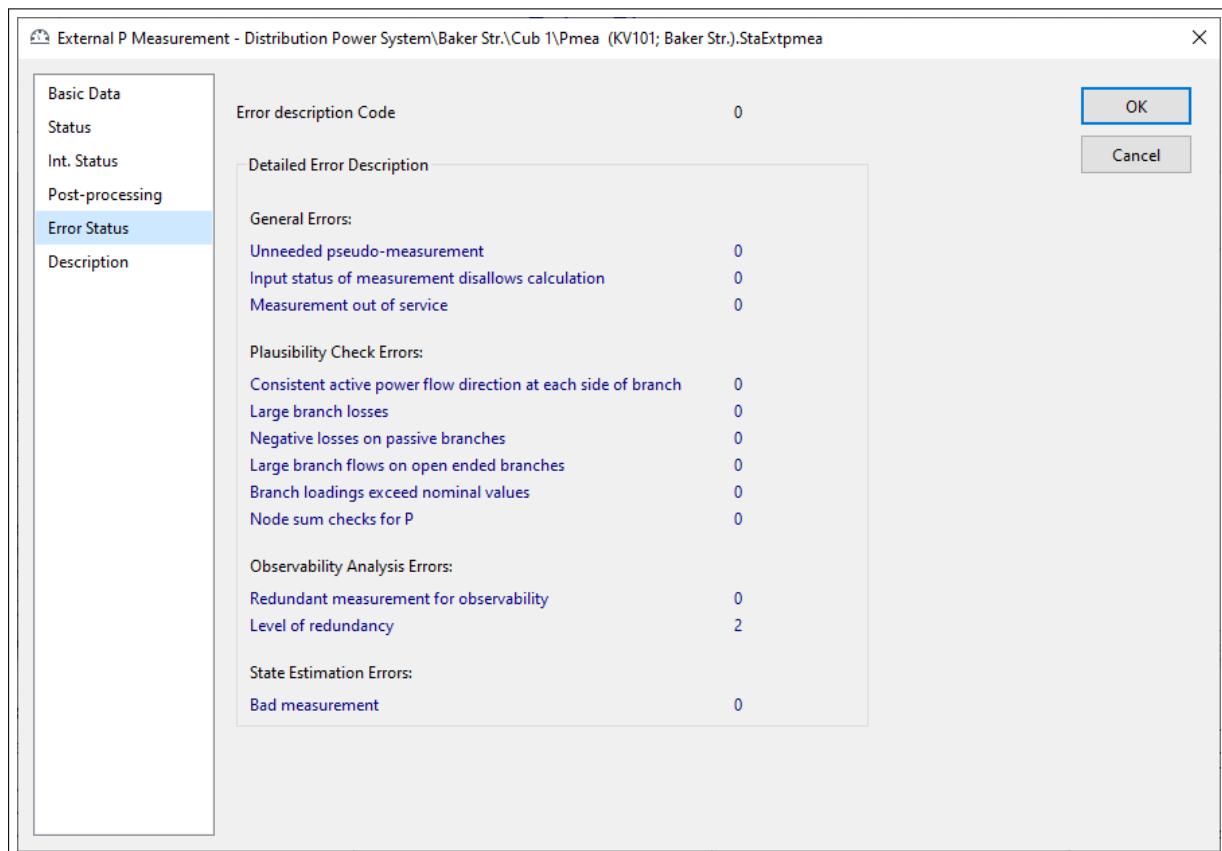


Figure 52.6.1: For description page for external measurements (*StaExtvmea*, *StaExtqmea*, *StaExtvmea*).

### 52.6.3 Estimated States

#### Which states participated as control variables?

Recall that - depending on the selected “treatment of unobservable regions” - not all states that were selected for estimation (see Section 52.4.2: Editing the Element Data) can be estimated. In addition, some states will necessarily be estimated by the algorithm. In case of non-observability, it may happen that some control variables need to be reset.

To access the information which states were actually used as control variables, *PowerFactory* provides a flag for each possible state. These flags are called *c:iPSetup*, *c:iQSetup*, *c:iScaleSetup*, *c:iTapSetup* for P-, Q-, Scaling factor-, and Tap-states, respectively. They can be accessed through the Flexible Data Page as State Estimation calculation parameters for the following elements: *ElmLod*, *ElmLodlv*, *ElmLodmv*, *ElmAsm*, *ElmSym*, *ElmSvs*, *ElmGenstat*, *ElmPvsys*, *ElmVoltreg*, *ElmTr2*, *ElmTr3*, and *ElmTr4*.

#### Observability of individual state

The Observability Analysis identifies, for each state, whether it is observable or not. Moreover, if the network is unobservable, it subdivides all unobservable states into “equivalence-classes”. Each equivalence-class has the property that it is observable as a whole group, even though its members (i.e., the single states) cannot be observed. The equivalence classes are enumerated in ascending order 1, 2, 3, ....

For this purpose, the Observability Analysis uses the flags *c:iObsFlg*, *c:iQobsFlg*, *c:iScaleobsFlg*, *c:iTapobsFlg* for P-, Q-, Scaling factor-, and Tap-states, respectively. These parameters exist as State Estimation calculation parameters for all elements which carry possible states (*ElmLod*, *ElmAsm*, *ElmSym*, *ElmSvs*, *ElmTr2*, *ElmTr3*...). The semantics is as follows:

- a value of -2 means that the correspond state is not estimated at all.
- a value of -1 means that the correspond state is unsupplied.
- a value of 0 means that the corresponding state is observable.
- a value of  $i > 0$  means that the correspond state belongs to equivalence-class  $i$ .

#### System state

If the non-linear optimisation converges, results will be available for all busbars and edge elements in the network analogue to the Load Flow results, e.g. the complex voltage balanced or unbalanced of every busbar. These results can be observed on the single line diagram with the result boxes or by using the *Flexible Data* page of the *Network Model Manager*.

Note, that these results are based on the whole process of the SE command including the bad data treatment. Results should therefore always be used together with the indicators regarding the observability. The colour representation in the single line diagram is suitable for a quick overview.

### 52.6.4 Colour Representation

In addition, *PowerFactory* provides a special colouring mode “State Estimation” for the single line diagram which takes into account the individual measurement error statuses and the states to be estimated (see Figure 52.6.2). The colouring can be accessed by clicking the icon  on the task bar.

The use of colouring in network diagrams is described in Section 10.3.10.1, and more detail about the use of colours and colour palettes can be found in Section 4.7.8.

The colour representation paints the location of measurements (of a specific type) and the location of states (of a specific type) simultaneously.

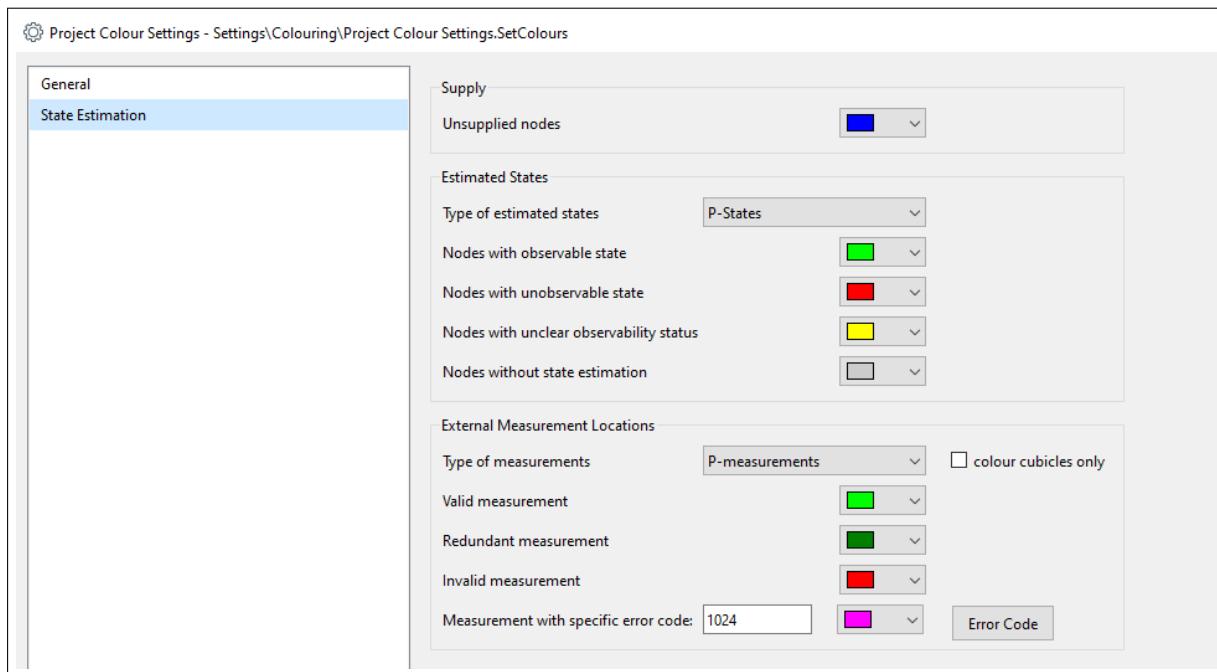


Figure 52.6.2: Colouring of measurement error statuses and estimated states.

### Estimated States

The user selects to colour states of a specific type (P-, Q-, Scaling factor-, or Tap position-states). Distinct colours for observable, unobservable, non-estimated states, and states with unclear observability status can be chosen.

### External Measurement Locations

The user selects to colour measurements of a specific type (P-, Q-, V-, or I-measurements). Distinct colours for valid, redundant and invalid measurements can be chosen. A measurement is said to be valid if its error code (`e:error`) equals 0.

Besides, measurements with a specific error code can be highlighted separately using an extra colour. To select such a specific error code press the **Error Code** button and choose from the detailed error description list any “AND”-combination of possible errors.

# Chapter 53

## Motor Starting

### 53.1 Introduction

The chapter presents *PowerFactory* tools for performing motor starting simulations using the Motor Starting command (*ComMot*). A Motor Starting analysis typically includes an assessment of the following:

- Voltage sag.
- Ability of motor to be started against the load torque.
- Time required to reach nominal speed.
- Supply grid loading.
- Starting methodology (Direct Online, Star-Delta, Variable Rotor Resistance, Reactor, Auto Transformer).

The Motor Starting command makes use of the *PowerFactory* stability module by providing a pre-configured shortcut for easy-to-use motor starting analysis. Pre-selected and pre-configured plots are automatically created and scaled with full flexibility for user-configuration. In *PowerFactory*, there are two “Simulation Types” that may be used to perform a motor starting simulation:

1. *Dynamic Simulation*, which will execute a time-domain motor starting simulation.
2. *Static Simulation*, which will execute a load flow calculation when the motors are disconnected from the system. Then, it will execute a short-circuit calculation, using the complete method, simultaneously with the occurrence of the motors being connected to the network. Finally, a load flow calculation will be executed after the motors have been connected to the system.

### 53.2 How to define a motor

To define the starting method of a motor, a Type must first be selected. This sub-section describes how to define the motor and (optionally) define a motor driven machine (mdm).

#### 53.2.1 How to define a motor Type and starting methodology

A comprehensive library of low-voltage, medium-voltage, and high-voltage motor types are available in the *DIGSILENT* Library. Typical motors supported are: single- and double-cage asynchronous machines and squirrel motors.

To define a motor Type and starting methodology for a dynamic simulation:

1. On the asynchronous machine Basic Data page, press *Select* (▼) and then choose an existing or define a new asynchronous machine Type. Press **OK** twice.
2. From the Data Manager or single line graphic, double-click the asynchronous machine to open the element dialog.
3. Depending on whether a dynamic or static motor starting simulation is to be executed:
  - For a dynamic starting simulation, navigate to the RMS-Simulation page, Advanced tab.
  - For a static starting simulation, navigate to the Complete Short-Circuit page.
4. Check *Use Motor Starting Method*.
5. Use radio buttons to select a starting method (see below).

#### **Directly Online**

For the direct online starting method, select *Directly Online*.

#### **Star-Delta**

For star-delta starting:

1. Select *Star-Delta*.
2. For a dynamic motor starting simulation, on the RMS-Simulation page, Advanced tab:
  - Select *Triggered by...* either *Time* or *Speed*.
  - Enter a simulation time for the motor to switch from the star winding to the delta winding *Switch to 'D' after*, or a speed for the motor to switch from the star winding to the delta winding *Switch to 'D' at Speed >=*.

#### **Variable Rotor Resistance**

For variable rotor resistance starting:

1. Select *Variable Rotor Resistance*.
2. For a static motor starting simulation, on the Complete Short-Circuit page:
  - Enter the *Additional Rotor Resistance*.
3. For a dynamic motor starting simulation, on the RMS-Simulation page, Advanced tab:
  - Select *Triggered by...* either *Time* or *Speed*.
  - In the *Variable Rotor Resistance* table, enter additional rotor resistance, and the time (or speed) at which the rotor resistance should be added.
  - For additional entries, right-click and Append or Insert rows as required. Note that a minimum of two-points must be entered.

#### **Reactor**

For reactor starting:

1. Select *Reactor*.
2. For a static motor starting simulation, on the Complete Short-Circuit page:
  - Enter the *Rated Apparent Power* and *Reactance*.
3. For a dynamic motor starting simulation, on the RMS-Simulation page, Advanced tab:
  - Select *Triggered by...* either *Time* or *Speed*.
  - Enter the *Rated Apparent Power*, *Reactance*.
  - Enter the time at which the reactor should be removed *Bypass after*, or speed at which the reactor should be removed *Bypass at Speed >=*.

#### **Auto Transformer**

For auto transformer starting:

1. Select *Auto Transformer*.
2. For a static motor starting simulation, on the Complete Short-Circuit page:
  - Enter the *Rated Apparent Power*, *Reactance*, and *Tap*.
3. For a dynamic motor starting simulation, on the RMS-Simulation page, Advanced tab:
  - Select *Triggered by...* either *Time* or *Speed*.
  - Enter the *Rated Apparent Power*, *Reactance*, and *Tap*.
  - Enter the time at which the star contactor should be released *Release Star Contactor after* and the time at which the auto-transformer should be bypassed *Bypass after*, or the speed at which the star contactor should be released *Release Star Contactor at Speed >=* and the speed at which the auto-transformer should be bypassed *Bypass at Speed >=*.

### 53.2.2 How to define a motor driven machine

Selection of a motor driven machine model provides enhanced flexibility to define the torque-speed characteristic of the motor. A motor driven machine can be user-defined, or selected from a range of Compressors, Fans, and Pumps available in the *DIGSILENT Library*. Refer to the asynchronous machine Technical Reference [Asynchronous Machine](#) and motor driven machine Technical Reference for further details [Motor Driven Machine](#).

To define a motor driven machine, in a Data Manager or on the Single Line Graphic, right-click on the asynchronous machine and:

- For a new motor driven machine:
  1. Select *Network Models* → *Motor Driven (mdm) machine* → *New...*
  2. Select a motor driven machine element (Type 1, Type 3, or Type 5).
  3. Enter the torque-speed characteristic.
- For a motor driven machine from the library:
  1. Select *Network Models* → *Motor Driven (mdm) machine* → *From Library...*
  2. Select an existing motor driven machine from the project library.

---

**Note:** Motor driven machines may also be defined for Synchronous motors by selecting the “Composite Type Sym frame” (or creating a user-defined frame). Refer to the mdm Technical Reference for further details: [Motor Driven Machine](#).

---

## 53.3 How to run a Motor Starting simulation

To run a motor starting simulation:

1. Select the motor or group of motors for the motor starting simulation.
2. Right-click a selected motor and select *Calculation* → *Motor Starting...*
3. Enter the command options (see following subsections for a description of the command options).

### 53.3.1 Basic Options Page

#### 53.3.1.1 Motor(s)

The motors selected for the Motor Starting command.

### 53.3.1.2 Simulation Type

Select either:

- *Dynamic Simulation* to initiate a dynamic motor starting simulation.
- *Static Simulation* to initiate a static motor starting simulation.

---

**Note:** Load Flow, Initial Conditions, Run Simulation, Simulation Events, Short-Circuit and Results Definitions objects in the active study case will be overwritten by the Motor Starting command.

---

### 53.3.1.3 Simulation Method

Either:

- If *User defined simulation settings* is not checked:
  1. Select to run either a *Balanced* or *Unbalanced* Motor Starting simulation.
  2. Enter the *Simulation Time* in seconds.
- If *User defined simulation settings* is checked:
  1. Define the variables to be monitored.
  2. Modify Load Flow Calculation command (*ComLdf*) settings as required.
  3. Modify Initial Conditions command (*ComInc*) settings as required. Note that motor starting events are automatically created, and that previously defined events are not deleted. Similarly, user-defined variable selections are merged with the Motor Starting command default variables.
  4. Modify Simulation command (*ComSim*) settings as required.

### 53.3.1.4 Monitoring

Click *Select* (▼) and select the *Additional Terminals* to be monitored for the Motor Starting simulation.

### 53.3.1.5 Check Thermal Limits of Cables and Transformers

Optionally select to *Check Thermal Limits of Cables and Transformers*. When this option is selected, the feeding cables and transformers of every motor will automatically be gathered, and its thermal limit will be checked.

The calculation of the thermal limits is performed depending on the type of simulation selected.

- **Dynamic Simulation**

Given the rated thermal overcurrent limit of the cable at 1 second ( $I_{thr1s}$ ), the thermal overcurrent limit of the line at the starting time of the motor ( $I_{thrTs}$ ) is calculated according to equation 53.1:

$$I_{thrTs} = \sqrt{\frac{I_{thr1s}^2}{T_{start}}} \quad (53.1)$$

Where:

$T_{start}$  = is the time calculated during the Motor Starting simulation.

The calculated thermal energy ( $I_{2t}$ ) during the motor starting is defined as:

$$I_{2t} = \int_0^{T_{start}} I^2 dt \approx \sum_0^{T_{start}} I^2 \Delta t \quad (53.2)$$

Where:

$\Delta t$  = is the integration step size of the simulation.

The calculated thermal current ( $I_{thrcalc}$ ) is then calculated as follows:

$$I_{thrcalc} = \sqrt{\frac{I_{2t}}{T_{start}}} \quad (53.3)$$

Finally, the thermal loading is calculated as the relation between rated thermal current and calculated thermal current at starting time:

$$ThermalLoading = \frac{I_{thrcalc}}{I_{thrTs}} \quad (53.4)$$

- **Static Simulation**

Given the rated thermal overcurrent limit of the cable at 1 second ( $I_{thr1s}$ ), the thermal overcurrent limit of the line at the starting time of the motor ( $I_{thrTs}$ ) is calculated according to equation 53.5 :

$$I_{thrTs} = \sqrt{\frac{I_{thr1s}^2}{T_{start}}} \quad (53.5)$$

The starting time is the variable  $tstart$  specified in the “Protection” page of the Asynchronous and the Synchronous Machine dialogs.

The calculated thermal current is the positive-sequence current calculated at the motor starting

$$I_{thrcalc} = I_{start} \quad (53.6)$$

Finally, the thermal loading is calculated as the relation between rated thermal current and calculated thermal current at starting time:

$$ThermalLoading = \frac{I_{thrcalc}}{I_{thrTs}} \quad (53.7)$$

### 53.3.2 Output Page

#### 53.3.2.1 Dynamic Simulation

##### Report

Check *Report* to report results to the output window. By default, report results include voltage before starting, minimum voltage during starting, voltage after starting, starting current and power factor, successful start, and starting time. The user can optionally modify report *Settings*.

##### Starting Tolerance for Simplified Models

Define the *Max. Speed Tolerance*, the maximum deviation from nominal speed at which the motor is considered to be successfully started. This applies only to simplified (i.e. synchronous) motors.

### 53.3.2.2 Static Simulation

#### Report

Optionally modify report *Settings* and *Results*. Figure 53.3.1 shows an example of a Static Simulation Report with the option “Check Thermal Limits of Cables and Transformers” selected.

			DIGSILENT PowerFactory 23.0.3.0	Project: ----- Date:
<b>Motor Starting</b>				
Study Case:	Berechnungsfall		Annex:	/ 1
Simulation Type:	Dynamic Simulation			
<b>Motors</b>				
Motor Name	Terminal Name	Terminal Voltage	Starting Current	Starting P.F.
		Before Starting   After Starting	(kA) (p.u.)	Successful Start?
				Approx. Starting Time (s)
Motor_with_mdm	Load bus	0.979   0.918	0.964   2.045   4.516   0.686	Yes   4.493
<b>Additional Terminals</b>				
Terminal Name		Terminal Voltage		
		Before Starting   Minimum on Starting		
Source Bus		1.000   0.998   0.999		

Figure 53.3.1: Report Example

#### Starting Tolerance for Simplified Models

Define the *Max. Voltage Drop* at which the motor is considered to be successfully started. This applies only to simplified models.

Simplified models are:

- All synchronous motors.
- Asynchronous motors with type Asynchronous Machine Type (*TypAsmo*), and without the Type option *Consider Transient Parameter* (*i\_trans*) checked.
- Asynchronous motors with any Type other than Asynchronous Machine Type (*TypAsmo*).

Detailed models are: Asynchronous motors with type Asynchronous Machine Type (*TypAsmo*), and which have the option *Consider Transient Parameter* checked on the VDE/IEC Short-Circuit page or Complete Short-Circuit page of the Type dialog. This provides a more precise result for the motor starting time.

#### Display results for

Select to display results on the Single Line Graphic:

- After motor starting.
- During motor starting.
- Before motor starting.

### 53.3.3 Motor Starting simulation results

#### 53.3.3.1 Dynamic simulation results

Following a motor starting simulation, *PowerFactory* will automatically create a plot (VI) for each motor showing the active power (m:Psum:bus1), reactive power (m:Qsum:bus1), current (m:I1:bus1), speed (s:speed), mechanical and electrical torques (c:xmt and c:xmem) and voltage of the motor terminal (m:u1). A second plot is created showing the voltage of monitored Terminals. Flexible data results variables available following a dynamic Motor Starting simulation are found on the motor data Motor Starting Calculation page.

The Motor Starting calculation variables are as follows:

- Terminal Pre-start Voltage, Magnitude (c:uprestart).
- Motor Start Voltage, Magnitude (c:ustart).
- Motor Post-start Voltage, Magnitude (c:upoststart).
- Starting current, Magnitude in kA (c:lstart).
- Starting current, Magnitude in p.u. (c:istart).
- Starting Power Factor (c:cosphistart).
- Successfully Started (c:started).
- Approx. Starting Time (c:Tstart).

The criterion of a successful start is as follows:

- Synchronous motors: Successful start if  $ActualSpeed \geq SynchronousSpeed - Tolerance$ , where *Actualspeed* is the value of variable “s:speed”, and *Tolerance* is the value specified in the input field *Max. Speed Tolerance (tolspeed)*.
- Asynchronous motors: Successful start if  $ActualSpeed \geq NominalSpeed - Slip$ , where *ActualSpeed* is the value of variable “s:speed”, and *Slip* is the value of variable “t:aslkp” of the asynchronous motor.

#### 53.3.3.2 Static simulation results

Following a motor starting simulation, new calculation variables are available for asynchronous (*El-mAsm*) and synchronous (*ElmSym*) motors. For the Static Simulation, these variables are found on the Motor Starting Calculation page. Results variables are described in the preceding sub-section.

The criterion of a successful start is as follows:

- Simplified models: Successful start if  $Voltage\ During\ Starting \geq Voltage\ Before\ Starting * (1 - Voltage\ Tolerance)$ , where *Voltage Before Starting* is the voltage value at the terminal before the motor is connected to the system, *Voltage During Starting* is the transient positive-sequence voltage value at the terminal during the motor start, and *Voltage Tolerance* is the value specified in the input field **Max. Voltage Drop (tolvolt)**.
- Detailed models: The electrical and mechanical torque are calculated for the minimum voltage value during the motor start up. A detailed model is considered to be successfully started up if the mechanical torque is always smaller than the electrical torque from zero speed up the peak of the electrical torque.

### 53.3.4 Motor Starting Example

Consider the following dynamic motor starting example for a single 6.6kV asynchronous motor shown in Figure 53.3.2.

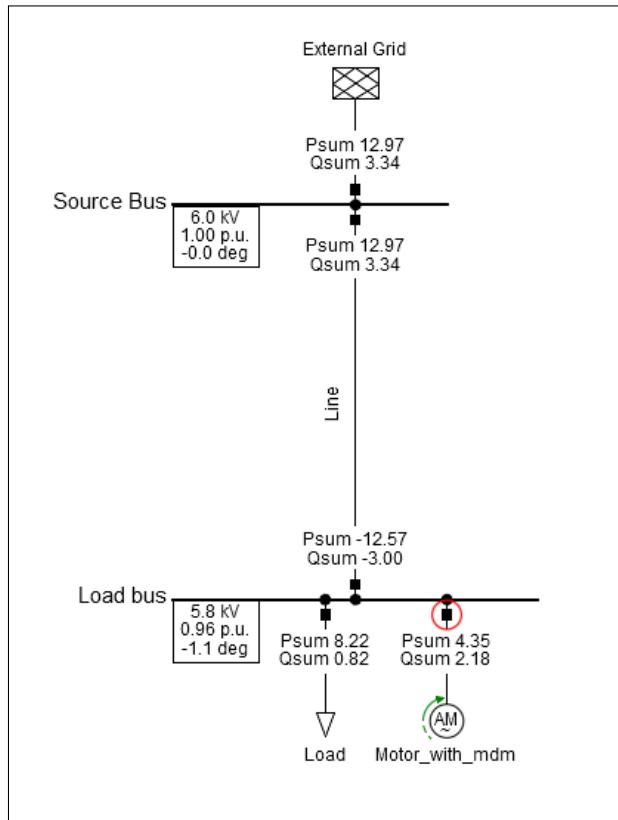


Figure 53.3.2: Motor Starting example Single Line Graphic

The *Variable Rotor Resistance* starting method has been selected, with three values of time-dependent resistance, as shown in Figure 53.3.3.

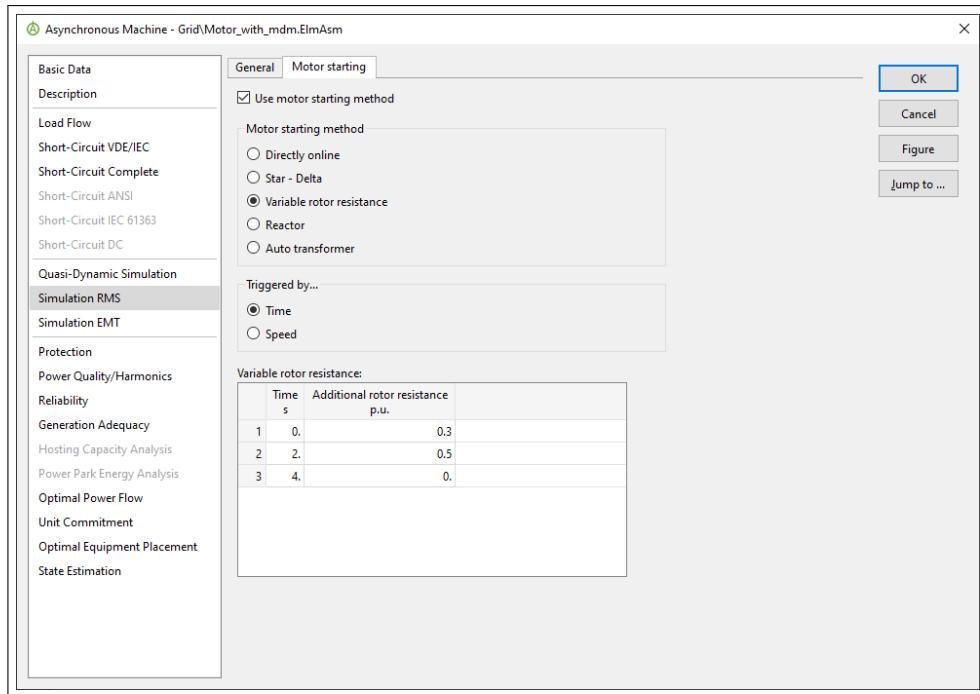


Figure 53.3.3: Motor starting methodology options

A dynamic, balanced Motor Starting simulation is executed and run to 10 seconds, with “Source Bus” selected as an *Additional Terminal* to be monitored, as shown in Figure 53.3.4.

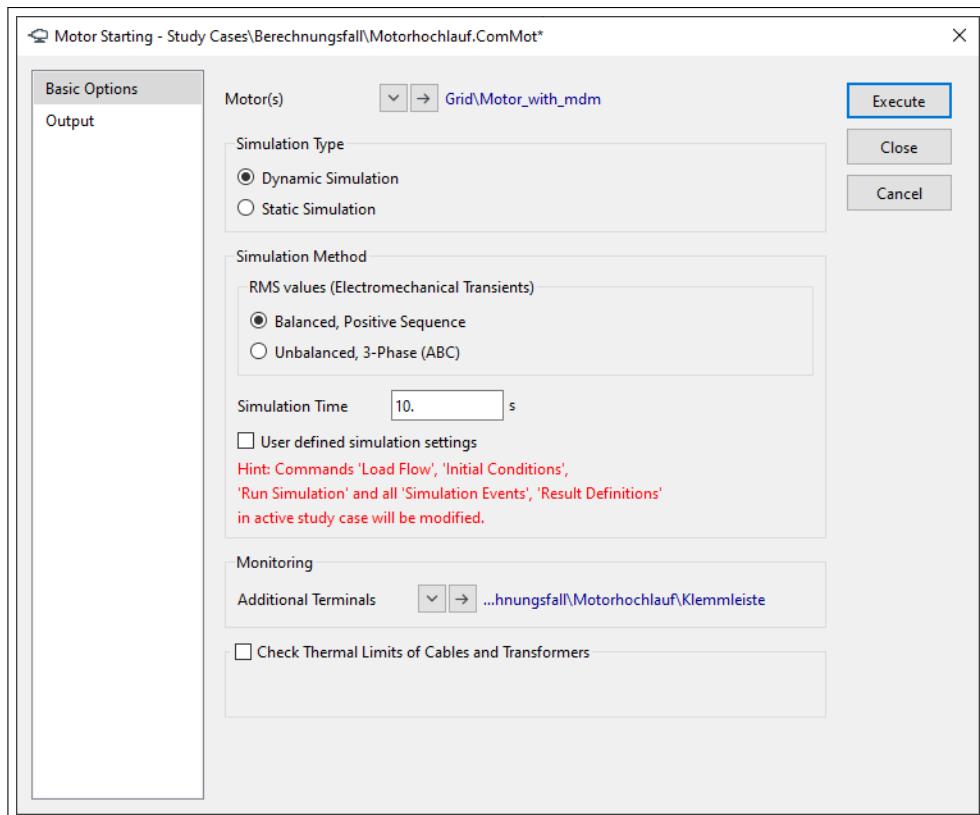


Figure 53.3.4: Motor starting Basic Options

Following execution of the command, *PowerFactory* automatically produces plots showing motor quan-

ties of interest (as described in Section 53.3.3.1) and monitored voltage results as shown in Figure 53.3.5 and Figure 53.3.6.

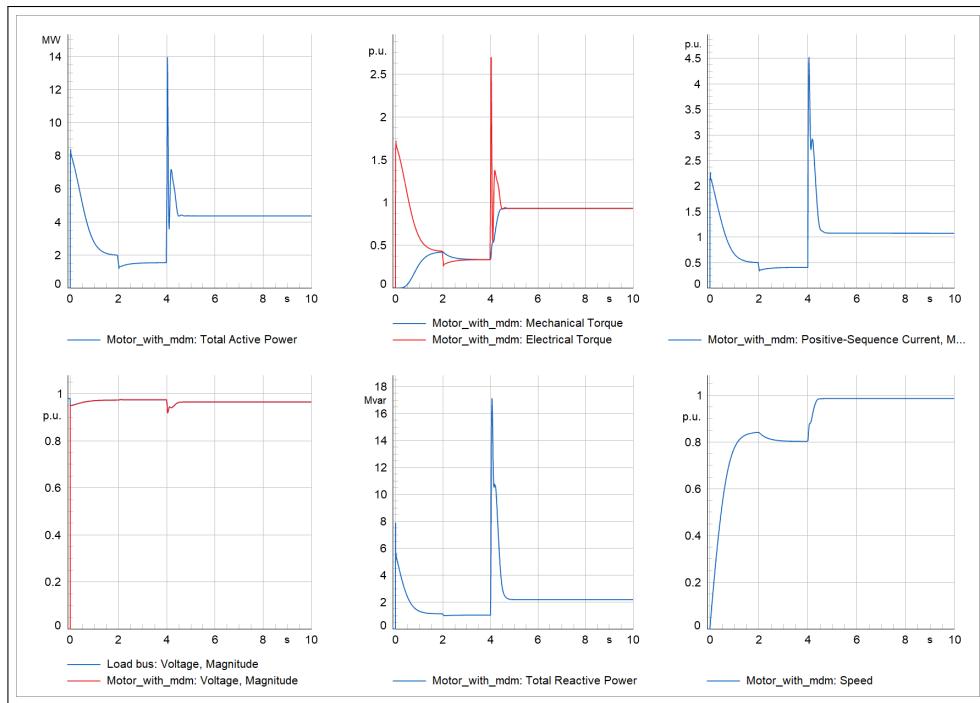


Figure 53.3.5: Motor starting example motor results

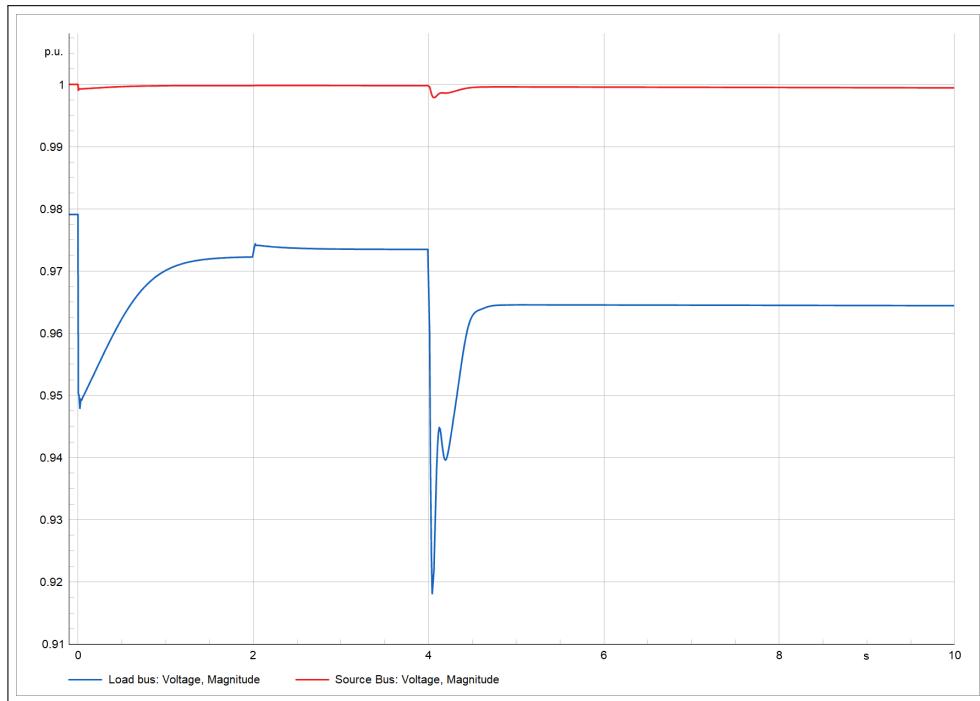


Figure 53.3.6: Motor starting example voltage results

# Chapter 54

# Artificial Intelligence

## 54.1 Introduction

Artificial intelligence describes the modelling of intelligent behaviour and machine learning and is already used in applications such as picture recognition and search engines. One field of artificial intelligence is artificial neural networks that are trained to predict specific values based on training datasets. *PowerFactory*'s *Artificial Intelligence* toolbox can be used to approximate load flow results using an artificial neural network.

In this chapter, the term “neural network” is used synonymously with “artificial neural network”.

Artificial neural networks are always useful if many load flow calculations need to be run on one grid model, as is the case for example with a quasi-dynamic simulation over a longer time period. To do this, a neural network is trained on a dataset of precalculated load flow results, in order to quickly approximate results for various dispatch configurations. This can increase the performance immensely, especially for larger power grids and many load flow calculations.

To use artificial neural networks two steps are necessary:

1. Generation or collection of dataset.
2. Training of neural network on the dataset.

The *PowerFactory Artificial Intelligence* toolbox provides functionality both for generating a training dataset and for training a neural network. For the dataset generation the *PowerFactory Probabilistic Analysis* is used.

The approximation with a neural network is totally different from the standard load flow algorithm. In the load flow algorithm, voltage and power flow are iteratively calculated based on physical laws, e.g. Kirchhoff's circuit law. All other variables are calculated based on these parameters. The neural network in contrast, finds a way to approximate some specific values in a non-iterative way by analysing data and finding coherences.

The neural network is capable of approximating the load flow for one network topology based on its training data. Changes in network topology are not supported at the moment. This includes any new network elements as well as any change in the switching status of the network. Each network topology requires its own neural network to be trained.

This chapter is structured as follows: First of all in Section 54.2 the Technical Requirements for using the *Neural Network Training* module are listed. Section 54.3 explains the technical background of the approximation of load flow results with a neural network and covers the advantages and limitations of this functionality. How to create and use a neural network within *PowerFactory* is described Section 54.4. In Section 54.5 and 54.6 the two commands to set up and create a neural network in *PowerFactory* are

detailed. Section 54.7 focuses on the output. Section 54.8 covers typical issues and possible solutions for the use of neural networks within *PowerFactory*.

## 54.2 Technical Requirements

The *Artificial Intelligence* toolbox uses the external library PyTorch. This library needs to be downloaded before the *Neural Network Training* function can be used. The download and setup of PyTorch is explained in the [Advanced Installation and Configuration Manual](#).

It is possible to train a neural network on a CPU, but for a good performance, the use of a suitable GPU is highly recommended. For compatibility a *Nvidia*-GPU together with the CUDA-API is required. *PowerFactory* automatically selects the GPU. Further information can be found in the [Advanced Installation and Configuration Manual](#).

## 54.3 Technical Background

Artificial neural networks are a special method in the field of artificial intelligence. They are able to learn tasks without being programmed with any task-specific rules just by considering examples. To visualise the way a neural network works, a simple example is used.

Imagine a tiny power grid with 2 generators, 1 load and 2 lines. Depending on the operation point of the generators and the load, the lines will be loaded accordingly. With the help of a neural network the line loading shall be approximated for different operating conditions. The neural network has therefore three inputs; the active power dispatch of Gen1, Gen2 and Load1; and two outputs; the loading of Line1 and Line2. This is illustrated in Figure 54.3.1. You can see that with the sample input parameters, load and generation setpoints, the neural network creates hidden layers and weighting factors  $w_i$  to predict the line loading. The circles represent the neurons. In practice, there is not just one hidden layer but many, and of course many more inputs and outputs.

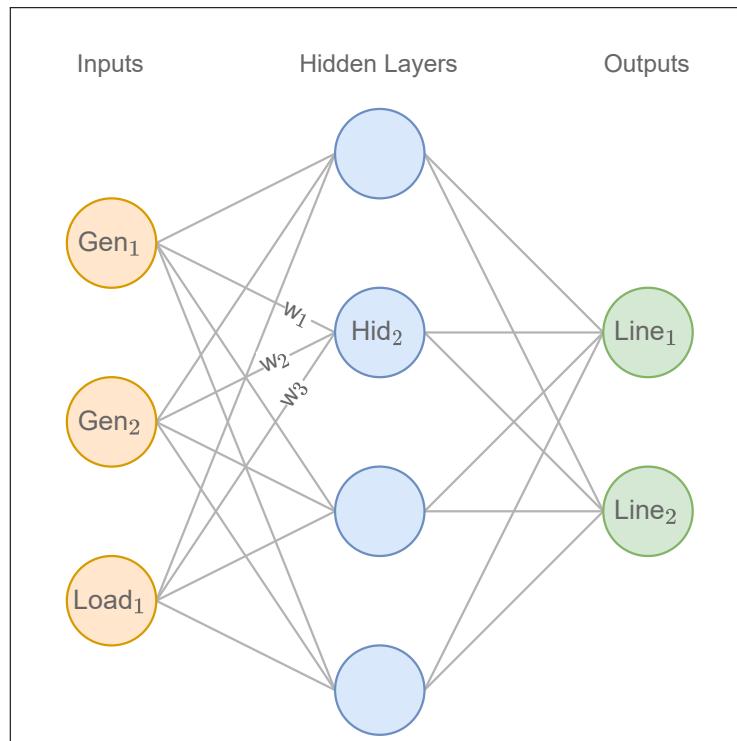


Figure 54.3.1: Example of a neural network with one hidden layer

*PowerFactory* uses a supervised learning approach to train the neural network. For the first load flow approximation, the weights are drawn from a random distribution so that the result is quite arbitrary. The calculated outputs are then compared to the generated training data. This is done by calculating the mean squared error (MSE) for all output values. In the next step the weights are adapted to reduce this error and the process is repeated with the next load flow samples until the whole training dataset has been used.

Then the validation data is used and the MSE for the validation data is calculated and the next iteration is started. These iterations are called epochs.

In *PowerFactory*, it is important to note that when using a neural network for a load flow approximation only input variables of the neural network may be changed, if a precise result is to be obtained; changing other variables may lead to different load flow results from those for which the neural network was trained, and so the results from the neural network may no longer be dependable. For example, consider the sample network above: Assume there is a second load, Load2, that was not selected as an input of the neural network. When the setpoint of this second load is now changed during the calculation, the new setpoint is not known by the neural network and this therefore leads to inaccuracy.

The quality of the approximation depends particularly on the similarity of the training data and the approximated load flow. Therefore, the training data needs to be chosen with regard to common load flow situations.

---

**Note:** Neural networks cannot predict the non convergence of a load flow. In consequence, one need to be careful when calculating load flows close to the stability limit with a neural network. No conclusions can be drawn from a neural network approximation of a load flow situation which is totally different from the training data.

---

### 54.3.1 Neural Network Architecture

*PowerFactory* supports two neural network architectures:

- Fully Connected Network
- Graph Convolutional Network

The “Fully Connected Network” (FCN) architecture is the classic neural network setup as shown in Figure 54.3.1. An FCN consist of input, hidden and output layers for which each neuron is connected with each neuron of the preceding layer.

The “Graph Convolutional Network” (GCN) utilises convolutional layers, which act on correlation-based estimated graphs. That means the value of a neuron is defined by a specific subset of neurons of the previous layer. The GCN has less weights than FCNs, thus is less prone to overfit to the training data. Additionally, it can achieve a potentially better approximation accuracy than FCNs for large power grids. For more details see [41].

## 54.4 How to create and use a Neural Network

This section explains step by step the general process of setting up of a neural network. The corresponding commands can be found within the *Additional Tools* toolbox. To use a neural network for the approximation of load flow parameters following steps need to be done:

1. Ensure that either distributions or characteristics are defined for all parameters that should be inputs to the neural network. Only these variables can be regarded as input values. The distributions should be continuous, not discrete. How to set up a time characteristic is described in Section 18.2.6. More information about distributions can be found in Section 45.2.1.

2. Configure the load flow command with the settings required for the approximate calculation. Bear in mind that settings such as regulation or load options can not be changed after the data was generated.
3. Configure the in- and output variables of neural network, and the distributions for the *Probabilistic Analysis*. There are three options for doing this:
  - Using the *Setup for Neural Network* command . This command is explained in Section 54.5.
  - Defining the variables within the *Neural Network Training* command  and creating the distributions for the *Probabilistic Analysis* with the *Distribution Estimation* command  explained in Section 45.3.3.
  - If the distributions were created manually, only the output variables need to be set within the *Neural Network Training* command. Nevertheless, it might still be advisable to calculate the correlation of the distributions using the *Distribution Estimation* command.

**Note:** It is recommended that the “Estimation of correlation” option be selected. With this option selected, the correlation of the input parameters such as solar power is taken into consideration, and so the generated dataset better represents common load flow situations.

---

4. (Optional) Test the settings by running a *Probabilistic Analysis*  with few samples. It might be necessary to select some kind of distributed slack within the load flow command to ensure the convergence of the probabilistic analysis, as explained in Section 54.8.1.
5. In the next step, a dataset is generated for the neural network training. The command is explained in detail in Section 54.6.2. Depending on the number of load flow calculations, this process might take a while. Once it is complete, a database object will be available for training the neural network.
6. The next step is to carry out the actual training of the neural network. The setup of the command is explained in Section 54.6.3. The result of the training is a neural network object.
7. Check the validation error of the trained neural network. If it is unsatisfactory, increase the training epochs or the numbers of samples of the dataset. In Section 54.8 there is some advice as to how to decide further steps.
8. Use the neural network to approximate a load flow result, e.g. in a *Quasi-Dynamic Simulation* (see Chapter 28). Note that the starting may take a while due to initialisation. The benefit of the neural network approximation process is therefore only realised if a long time period is to be analysed.

If there are doubts about the quality of the generated dataset or the trained network, the simulation can be executed for a short time period to compare the approximated results with the standard load flow results. If the results are not satisfactory, help can be found in Section 54.8.

---

**Note:** The load flow settings used for data generation should be chosen carefully, as the neural network will approximate a load flow with exactly these settings.

---

## 54.5 Setup for Neural Network command

The *Setup for Neural Network* command  creates the setup for a neural network for a *Quasi-Dynamic Simulation*. The load flow settings and the output variables for the creation of the neural network are taken from the *Quasi-Dynamic Simulation*. In addition this command automatically creates the distributions for the probabilistic analysis based on time characteristics.

The intention is that this neural network should be used later on to run this *Quasi-Dynamic Simulation*; see Section 28.3.7.

This command is placed within the *Additional Tools* toolbox.

### 54.5.1 Basic Options

On the Basic Options page the following options are available:

- **Configuration of:** Here the *Neural Network Training* command for which the setup is done is linked.
- **Setup for:** The *Quasi-Dynamic Simulation* (QDS) for which the setup is done is linked. The load flow settings as well as the result variables are taken from this QDS. Note that other settings are not considered. The setup of a QDS is explained in detail in Section 28.3.
- **Variation management:** With this option the user can define whether the newly created distributions and correlations for the *Probabilistic Analysis* should be stored in a new variation or in the grid itself.

### 54.5.2 Input Variables

The data generation for the neural network is carried out using the *Probabilistic Analysis*. The *Probabilistic Analysis* is based on distributions for the changing input parameters, which can be estimated based on the characteristics. For this the *Distribution Estimation* command is used, described in Section 45.3.3. It is recommended to tick the option *Estimation of correlations* as this ensures that the generated load flow cases are similar to the actual ones.

If distributions and correlations are already defined in the *PowerFactory* project, the distribution estimation selection can be deselected.

The button *Show all characteristics* opens a window with all characteristics of the project. Note that in very large networks this may take a while. All characteristics are stored in the *Characteristics* folder in the *Operational Library*.

### 54.5.3 Output Variables

The output variables define which values are approximated by the neural network. By default they are defined according to the *Quasi-Dynamic Simulation*. By clicking on the *Define and edit output variables* button, variables can be deleted or additional ones selected.

It is recommend to focus on the output variables of interest.

## 54.6 Neural Network Training command

This section explains the options of the *Neural Network Training* command . The command is placed within the *Additional Tools* toolbox.

The two main tasks of this command are:

1. Generate a dataset as basis for the neural network training.
2. Train a neural network using this data.

### 54.6.1 Basic Options

On the *Basic Options* page following options are configured:

## Approximated calculation

The load flow for the approximation is linked and can be configured further. The neural network is trained to approximate the result for a specific load flow calculation. Therefore, settings such as the power regulations should be chosen carefully.

As the *Load Flow* command is stored within the *Neural Network* command, this will not affect the load flow of the study case.

## Tasks

There are three options available:

- **Data generation:** If this task is selected, the command generates data for the neural network training. This creates a *Dataset* object.
- **Neural network training:** This task creates a *Neural Network* object and trains the neural network on a generated dataset.
- **Data generation and neural network training:** With this task a dataset is generated and then a neural network is trained on this dataset. In this case, both the *Dataset* and *Neural Network* objects are created.

## Method

There are two methods: *new* or *continue*. If *new* is selected, a new dataset or neural network is created. With *continue*, an existing dataset is extended or the training of a neural network is continued.

Note that if the task *Data generation and neural network training* is used and the method *Continue* is selected there needs to be a dataset as well as a neural network in order to continue with the training.

## 54.6.2 Data Generation

The options presented on the *Data Generation* page differ depending on the selected method.

### 54.6.2.1 New Data Generation

If the method *New* is selected, the following options are available:

#### Probabilistic Analysis

*Probabilistic Analysis* is used to generate a wide range of load flows for the dataset.

- **Command:** Here the *Probabilistic Analysis* command is linked.
- **Consider distribution correlations:** If this option is selected, the correlation of the distributions is considered. If correlations are neglected this may lead to inadequate datasets, see Section 54.8.3

*Probabilistic Analysis* is explained in detail in Chapter 45.

#### Training variables definition

- **Show input variables:** Shows the distributions used for the input variables. The input variables are defined taking account of the distributions and can therefore not be manually set.
- **Edit output variables:** Here the user can select the output variables for the dataset and the neural network training. Only the selected output variables can be approximated with the neural network.
- **Automatic support of currents, voltages and powers:** If this option is ticked, *PowerFactory* automatically selects the required variables as outputs, in order to be able later on to calculate

currents, voltages and powers of all elements. This can lead to many output variables and thus increase the size of the dataset and the neural network.

### Data properties

- **Min. number of samples:** Defines the minimum number of generated data samples. Depending on the chunk size, the number of generated samples may be higher. The number of samples should scale with the number of elements (nodes, loads, generators etc.) in the grid. The more elements there are, the more samples are needed for the training.
- **Chunk size:** Data generation is done in chunks to avoid unwanted large data operations. Each chunk corresponds to one execution of the *Probabilistic Analysis*. The chunk size should be an divisor of the min. number of samples to avoid additional samples. For small grids the chunk size can be increased to speed up the data generation.
- **Data location:** Location on the disk for the data to be stored. As the dataset can be quite big, sufficient free space should be made available.

### Project export for grid state backup.

The project can be exported to the same location the generated data is stored at. This allows the user to save the state of the grid fitting to the generated data.

#### 54.6.2.2 Continuation of Data Generation

If the method *Continue* is selected, an existing dataset is expanded. In this case the following options are available:

- **Extension of dataset:** The dataset to be expanded is selected. It is important that this dataset was generated with the same load flow settings and the same topology.
- **Min. number of samples:** Defines the minimum number of samples for this dataset. This parameter considers also the already created samples.
- **Chunk Size:** See the explanation in the New Data Generation section above.

#### 54.6.3 Neural Network Training

The options presented on the *Neural Network Training* page differ depending on the method selected on the *Basic Options* page.

##### 54.6.3.1 Training of new neural network

If the method *New* is selected on the *Basic Options* page, the following options are available on the *Neural Network Training* page:

- **Dataset:** The dataset on which the neural network should be trained.
- **Location:** The directory in which the neural network is stored. This location should be different from the location of the data to ensure separation of data and neural network.
- **Stopping criteria:** The stopping criterion is defined as *Epochs* or as a mean square *Error*. If the stopping criterion *Error* is selected, the training will continue until the validation error is below the supplied value. In some cases it may happen that this condition is never met. If this happens, the user will have to manually stop the calculation by pressing the Break button .
- **Neural network architecture:** Two different architectures are available; the “Fully Connected Network” and the “Graph Convolutional Network”. The different neural network architectures are explained in Section 54.3.1.

### 54.6.3.2 Continuation of neural network training

If the method *Continue* is selected on the *Basic Options* page, the training of an existing neural network is continued. For this an existing neural network is selected. The other options are similar to those for the training of a new neural network.

### 54.6.4 Random Number Generation

On the *Random Number Generation* page, the *Seeding type* is defined. This value has two functions:

- Data Generation: Seeding of the probabilistic analysis.
- Neural Network Training: Defining the initial weights of the neural network.

The seeding type can be selected as *Automatic* or *User defined*. The *User defined* option is used to create a reproducible neural network.

## 54.7 Output of Neural Network Training

The generated data and the trained network are not stored directly in the database but on disk. For this *PowerFactory* provides two objects to link this external data to the database.

### 54.7.1 Dataset Object

The *Dataset* object (*IntNndata*) is the link to the generated datasamples stored on disk. The following information is defined within the object:

- **Name:** The name of the dataset. This can be edited by the user.
- **Location:** The location on disk of the data samples.
- **Samples:** Number of data samples on disk.
- **Input Variables:** Number of input variables on disk.
- **Output Variables:** Number of output variables on disk.
- **Version used for data generation:** The *PowerFactory* version used for the data generation. If the data or the neural network is to be used within another *PowerFactory* version it is recommended to check the consistency first.
- **Probabilistic Analysis:** Command used for the data generation.

With the two buttons *Show input Variables* and *Show output Variables* one can check the in- and output variables used.

### 54.7.2 Neural Network Object

The *Neural Network* object (*IntNnet*) is a link to the externally saved description of the neural network.

#### General properties

- **Location:** Path to the folder
- **Trained calculation:** Load flow calculation used to create the training data.

- **Dataset:** Dataset used for the training

**Training properties** The *Training properties* are for information only.

- **Number of training samples:** Number of samples used for training. The other samples are used for validation.
- **Number of training epochs:** Number of epochs that were run for the training.
- **Training error (MSE):** The mean square error calculated for all output variables of the training samples.
- **Validation error (MSE):** The mean square error calculated for all output variables of the validation samples. Shows how good the neural network can estimate unknown data.

Since most of this data is not stored in the *PowerFactory* database, but on disk, opening the neural network object might take a short time if the disk is a hard drive disk which is currently on stand-by and has to be started.

### 54.7.3 Consistency Check

Both the Dataset Object and the Neural Network Object have a *Check consistency* button to check whether the current topology fits to the one the dataset was generated with.

By execution of this check, multiple training samples from disk are loaded into *PowerFactory*. A *Load Flow* is then executed for each of these samples. It is then checked whether the results of the *Load Flow* still match the results of the samples.

If they match, it is very likely that the power system is in the same state as when the data was generated or the neural network was trained.

If the results do not match, the power grid is most likely in a different state. For further examination it is recommended to import the grid state backup created by the *Neural Network Training* command and use the *PowerFactory Compare and Merge Tool* (see Section 21.4). A comparison of the current project and the grid state backup will give an indication of where the difference may be coming from.

This function should always be used if there is any doubt about changes in the topology between the creation of the neural network and its use.

## 54.8 Trouble Shooting for Neural Networks

In this section some advice for common issues when training and working with neural networks are given. Note that the handling of neural networks is quite different to other functionalities in *PowerFactory*, as the neural network trains itself.

### 54.8.1 The Probabilistic Analysis does not converge.

The *Probabilistic Analysis* generates random load flow situations according to the parameter distributions. Especially if loads and generators are modified, this can lead to extreme situations and a big power mismatch that needs to be covered by the slack.

Some not converging cases are typical in larger power grids. But if there are more than 20%, distributed slack might be necessary. This can be configured in the *Load Flow* command. The distributed slack options are explained in detail in Section 25.3.2.

For more information regarding *Probabilistic Analysis* see Chapter 45.

### 54.8.2 The validation error does not decrease during the epochs.

It might be the case that the training error decreases while the validation error stays constant or even increases. This indicates that the neural networks only memorises the trained data instead of learning patterns. This phenomenon is called overfitting. Generally this issue can be solved by increasing the training dataset using the *continue* function. A sample training process is shown in Figure 54.8.1. The training error decreases but the validation error stays constant.

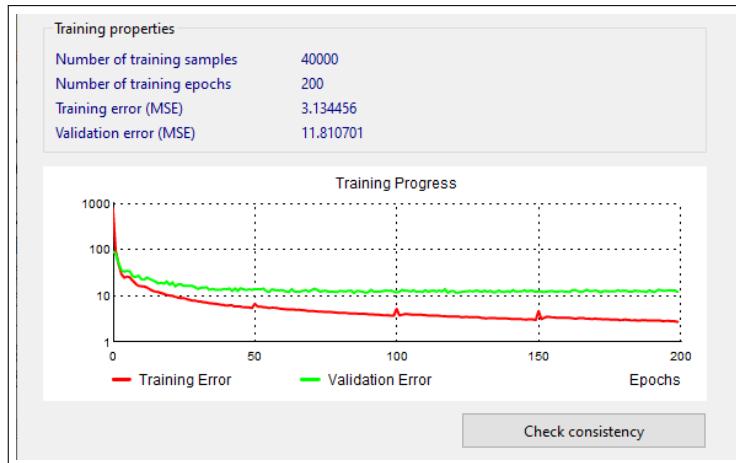


Figure 54.8.1: Training process for neural network with inadequate training dataset

### 54.8.3 The approximation of the neural network is not good enough

There are several reasons why a neural network approximation may significantly differ from the load flow results even if the validation error of the neural network is small. Some typical reasons are listed here.

#### The network topology has changed.

To make sure the neural network was not trained on a different network topology, use the "Check Consistency" button inside the *Neural Network* object.

If a grid backup was done during the data generation, this grid model can be used for the simulation to ensure same topology.

#### Different load flow settings.

If the load flow settings have changed between the data generation and the use of the neural network, this may lead to false results. Therefore it is important to check whether the settings of the load flow inside the neural network are the same as the ones of the current calculation.

#### The training data does not fit the actual dispatch.

If the actual load flow calculations are quite different from those of the training dataset, it is hard for the neural network to guess the result as it does not know similar situations. In Figure 54.8.2 the result of a *Quasi-Dynamic Simulation* with a neural network approximation and the standard load flow are plotted. It can be seen that the results fit good for a low line loading and differs for higher line loading. The reason is that the correlations of the loads were not considered when the training samples were generated. Therefore, the high-load situations were not represented in the training dataset and the neural network could not learn these situations. As the neural network cannot run a plausibility check the approximation can take unrealistic values, e.g. a negative loading, as shown in the example.

By considering the correlations of the distributions, this issue can often be solved.

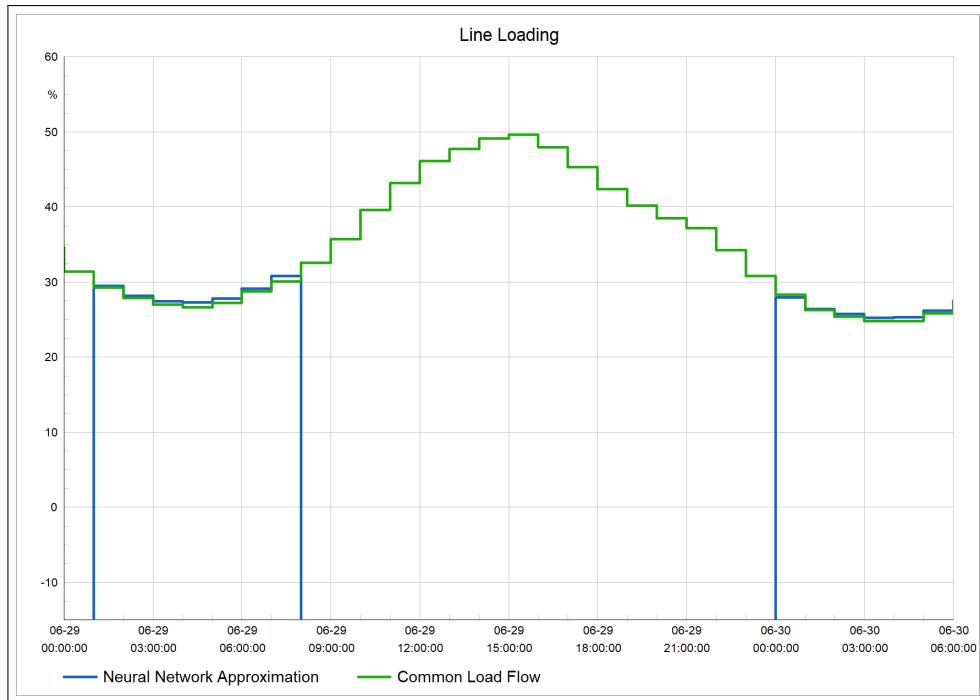


Figure 54.8.2: Example for neural network approximation where the network was not trained on a suitable dataset

**Part V**

## **Appendix**

## Chapter 55

# The *DlgsILENT* Output Language

When more than just the variable name, value and unit has to be displayed, if the use colours is preferred, or other special formats, the *DlgsILENT* Output Language can be used.

By selecting the *Format Editor* input mode, the editor is activated (see Figure 55.0.1).

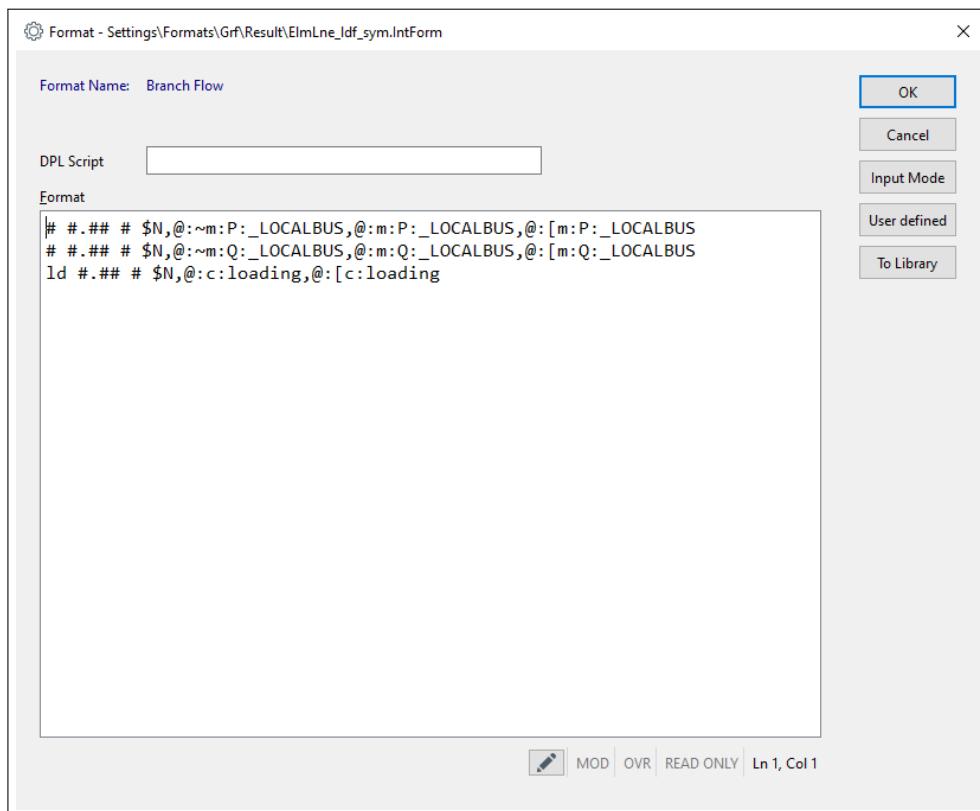


Figure 55.0.1: The Form text editor

Almost all textual output that is produced in *PowerFactory*, is defined by a report form. The use of report forms range from the simple and small result forms that specify the contents of the single line result boxes to large and complex forms that are used to print out complete system reports.

In all cases, the text in the editor field of a *IntForm* object specifies the report that is to be generated. For result boxes, that text is normally created automatically in the *IntForm* dialog by selecting “Predefined Variables”, or any other set of variables, and some extra’s such as the number of decimals and if an unit

or name should be shown. These options will automatically create a report form. That automatic form is normally used as it is, but it may be altered manually. This is shown in Figure 55.0.1, where report format is changed such that the variable name of the loading factor is deleted and replaced by the fixed text 'ld', because the variable name "loading" is felt too long compared with the names of the other two variables ("P" and "Q"). The shown format will produce result boxes like

```
P 12.34 MW
Q 4.84 Mvar
ld 103.56 %
```

Defining single line result boxes only asks for a basic understanding of the *DlgSILENT* output language. For more complex reports, many different variables from all kinds of objects have to be printed as listings or tables. Such a report would require macro handling, container loops, selection of parameters, headers, footers, titles, colours, etc. The *DlgSILENT* output language offers all this, and more.

The basic syntax, which is primary used for defining result boxes is given in the following overview.

- Format string, Variable names and text Lines
- Placeholders
- Variables, Units and Names
- Colour
- Advanced Syntax Elements
- Line Types and Page Breaks
- Predefined Text Macros
- Object Iterations, Loops, Filters and Includes

## 55.1 Format string, Variable names and text Lines

A standard line consists of three parts (see Figure 55.1.1):

1. A format string, containing placeholders, macros and/or user defined text.
2. An 'end of line' character like '\$N', '\$E' or '\$F'
3. Variable names, separated by commas, which are used to fill in the placeholders.

```
#.## $N,@:m:P:_LOCALBUS
#.## $N,@:m:Q:_LOCALBUS
#.## $N,@:c:loading
```

(1)	(2)	(3)
-----	-----	-----

Figure 55.1.1: Basic parts of the report format

The format string is normally much longer.

## 55.2 Placeholders

A placeholder for strings like variable names or whole numbers is a single '#' -sign. For real numbers, the placeholder consists of

- a single '#' for the integer part
- a point or comma
- one or more '#' -signs for the fractional part

The number of '#' -signs after the decimal point/comma defines the number of decimals. The '#' -sign itself can be included in user-defined text by typing '\#'.

## 55.3 Variables, Units and Names

The variable name can be used to display the name of the variable, its value or its unit. The possible formats are ('xxx' = name of variable):

xxx returns the value  
%xxx returns the long variable name, as used in the edit dialogs  
&xxx returns the short variable name, as used in the database browser  
[xxx returns the unit  
xxx the object dependent name of the variable (default name)  
"%width.precision,xxx"  
uses special formatting.

The special formatting %width.precision is explained by the following examples:

- "%.60,TITLE:sub1z" outputs TITLE:sub1z 60 column width, left aligned.
- "@:"%1.0,s:nt" inserts s:nt as an integer at the placeholder's position
- ""%1.3,s:nt" writes s:nt with 3 digits precision at the placeholder's position

The centring code | may be used in front of the formatting code for centring at the placeholder, for example "|%.60,TITLE:sub1z".

The insertion code | is used to switch to insert mode, for example,

```
| # | $N,:loc_name  
will output  
| aElmSym| .
```

The cformat string may be used to alternatively reserve place for a value or text. A cformat of '%10.3' will reserve 10 characters for a number with 3 decimals. The first number can be omitted for text: '%.6' will reserve 6 characters for the text field. The cformat syntax allows for centring text by adding the '|'-sign to the '%' -sign:

'|%.10' will reserve 10 characters and will centre the text.

Free, language dependent text can be defined by use of the format

{E[a text;G|ein Text]. This will produce 'a text' when the user has selected the English language (see the user settings dialog), and 'ein Text' when the language has been chosen to be German.

Special commands for access of Elements

### OBJECT(cls)

Gets Element of class cls. Used to access a variable name or unit without actually accessing such an object. Used in header lines.

#### argument

cls (obligatory): The name of the class

#### example:

```
[OBJECT (ElmTerm) :m:Skss
```

writes the unit of the busbar variable Skss

### EDGE

Gets an arbitrary object with at least one connection, i.e. a Load, a Line, etc. Used to access a variable name or unit without actually accessing such an object.

**example:**

```
%EDGE:m:U1:bus1
```

writes description of the variable U1

**CUBIC(idx)**

Returns the cubicle (*StaCubic*) at bus index idx of branch

**argument**

idx: index of branch, the currently set bus index is used when idx<0

**example:**

```
CUBIC(0):e:loc_name
```

returns name of cubicle at busindex 0

**TITLE**

Gets the title that is set in the output command (*ComSh* or *ComDocu*)

**example:**

```
TITLE:e:annex
```

writes annex of title

**VARIANT**

Gets the active variant in which the current object is stored

**example:**

```
VARIANT:e:loc_name
```

writes the name of the variant

**NET**

Gets the grid in which the current object is stored

**example:**

```
NET:e:loc_name
```

writes the name of the grid

**CMD**

Returns the last calculation command, i.e. a Short-Circuit (*ComShc*), Load-flow (*ComLdf*),...

**example:**

```
CMD:pabs
```

writes the short-circuit position on the line after calculation of a short-circuit.

**CASE**

Returns the currently active calculation case

**example:**

```
CASE:e:loc_name
```

writes the name of the active calculation case

**DEF**

Returns the default object. The default object depends on the currently processed output.

**example:**

```
DEF:e:loc_name
```

writes the name of the default object

**STALNE**

Returns the station if the current object is a busbar. Returns a line if the current object is a terminal between line routes. Otherwise, nothing is returned, and the entry will be ignored.

**example:**

```
STALNE:e:locname
```

writes the name of the line or station.

**RES**

Returns the currently active results object (*ElmRes*) used by simulation, harmonics or other calculation modules

**example:**

```
RES:e:desc
```

writes the first line of the description of the results object

## 55.4 Colour

A line can be set to another colour by adding a '\_LCOL(c)' command directly after the '\$N' marker. This will colour the whole line according to the colour number c.

<i>a</i>	<i>black</i>	<i>i</i>	<i>gray</i>
<i>b</i>	<i>black</i>	<i>j</i>	<i>lightgray</i>
<i>c</i>	<i>red</i>	<i>k</i>	<i>bordeaux</i>
<i>d</i>	<i>green</i>	<i>l</i>	<i>darkred</i>
<i>e</i>	<i>blue</i>	<i>m</i>	<i>darkgreen</i>
<i>f</i>	<i>brown</i>	<i>n</i>	<i>lightgreen</i>
<i>g</i>	<i>cyan</i>	<i>o</i>	<i>marine</i>
<i>h</i>	<i>magenta</i>	<i>p</i>	<i>darkblue</i>

Table 55.4.1: Colour Codes

A single item can be coloured by using the '\_COLOR(Variable name; color code)'.

## 55.5 Advanced Syntax Elements

The advanced syntax is mainly used for writing forms for larger and more complex reports. An example is a short-circuit result form, which lists all the short-circuit parameters for all busbars and for each busbar for all connected elements.

## 55.6 Line Types and Page Breaks

The character '\$' ends a format line. A line without this ending will be interpreted as a normal '\$N' line type. The following line type are available:

- '\$N' Normal line
- '\$H' Header on the top of each page
- '\$F' Footer on the bottom of each page
- '\$T' Title line, only appears on top of the first page
- '\$C' Comment line (not used for output)
- '\$R' Marker that make that the line will only be used when the specified results are valid

The line type '\$H', '\$F' and '\$T' will be treated as normal ('\$N') line types when used inside a loop command. Line type codes may be made language dependent by adding a 'E', for English lines or a 'G' for German lines, i.e. '\$HG' specifies a German header line.

A report format must at least contain one normal (\$N) line.

The following commands are used for page and line controls. They can only be used directly behind the line type codes '\$N', '\$F' or '\$H'.

- \_PAGEBREAK** Forces a page break after the current line
- \_AVAILBREAK** Enables page breaking after the current line (default)
- \_NOBREAK** Disables page breaking directly after the current line
- \_LCOL(c)** Disables page breaking directly after the current line
- \_OBJ(ClsNam)** The current line will only be used for objects from the class "ClsNam".
- \_BUS(inum)** The current line will only be used for objects which connect to exactly inum nodes
- \_FIRST** The current line will only be used when the loop index is 0 (first passage)
- \_NFIRST** The current line will only be used when the loop index is not 0 (all but the first passages)
- \_IF(boolean expression)** The current line will only be written when the expression is true. Example:  
  `_IF(m:u:bus1>0.95)`
- \_IFNOT(boolean expression)** The current line will only be written when the expression is false. Example:  
  `\IF(m:u:bus1<0.95)`

Example:

```
| #.## # .## # .## |$R,_NOBREAK, ..
```

## 55.7 Predefined Text Macros

The following macros will produce specific names or other texts.

- \_DATE(c)** present date: c='e' give the English format, c='g' the German one.
- \_TIME** present time
- \_VERSION** version number of the DlgSILENT PowerFactory software.
- \_BUILD** build number of the DlgSILENT PowerFactory software.
- \_VERBUILD** combines \_VERSION and \_BUILD
- \_ORDER** order title, if a title has been defined previously

**\_CLASS** class name of the object  
**\_LINE** current line number in page  
**\_ALLLINE** current line number in report  
**\_PAGE** current page number  
**\_LOCALBUS** name of the local busbar  
**\_CALC(c)** name of last performed calculation. c=1 returns a long description.  
**\_SHORT** short object name  
**\_FSHORT** short name of parent object  
**\_CLS** class name without the 'Elm', 'Sta', 'Typ', etc. part.  
**\_ANNEX** the annex number  
**\_NGB** neighbourhood depth  
**\_TEXT(E | text;G | Text)** language dependent text (E=English, G=German)

## 55.8 Object Iterations, Loops, Filters and Includes

To create a report that creates a table with the voltages for all busbars, command are needed to filter the busbar objects and to create a loop that outputs a line of text for each busbar. A loop or filter command consists of the following parts:

- the keyword "\$LOOP" or "\$CLOOP"
- the filter or loop name
- the format text
- the keyword "\$END"

# Chapter 56

## Standard Functions DPL and DSL

These functions are present in both DPL and DSL, click on the link to go to the corresponding chapter.

- [Models for Dynamic Simulations \(DSL\)](#)
- [Scripting \(DPL\)](#)

Function	Description	Example
<b>sin(x)</b>	sine	$\sin(1.2)=0.93203$
<b>cos(x)</b>	cosine	$\cos(1.2)=0.36236$
<b>tan(x)</b>	tangent	$\tan(1.2)=2.57215$
<b>asin(x)</b>	arcsine	$\text{asin}(0.93203)=1.2$
<b>acos(x)</b>	arccosine	$\text{acos}(0.36236)=1.2$
<b>atan(x)</b>	arctangent	$\text{atan}(2.57215)=1.2$
<b>atan2(x,y)</b>	arctangent	$\text{atan2}(-2.57215,-1)=-1.9416$
<b>sinh(x)</b>	hyperbolic sine	$\sinh(1.5708)=2.3013$
<b>cosh(x)</b>	hyperbolic cosine	$\cosh(1.5708)=2.5092$
<b>tanh(x)</b>	hyperbolic tangent	$\tanh(0.7616)=1.0000$
<b>exp(x)</b>	exponential value	$\exp(1.0)=2.718281$
<b>ln(x)</b>	natural logarithm	$\ln(2.718281)=1.0$
<b>log(x)</b>	log10	$\log(100)=2$
<b>sqrt(x)</b>	square root	$\sqrt{9.5}=3.0822$
<b>sqr(x)</b>	power of 2	$\text{sqr}(3.0822)=9.5$
<b>pow(x,y)</b>	power of y	$\text{pow}(2.5, 3.4)=22.5422$
<b>abs(x)</b>	absolute value	$\text{abs}(-2.34)=2.34$
<b>min(x,y)</b>	smaller value	$\text{min}(6.4, 1.5)=1.5$
<b>max(x,y)</b>	larger value	$\text{max}(6.4, 1.5)=6.4$
<b>modulo(x,y)</b>	remainder of x/y	$\text{modulo}(15.6,3.4)=2$
<b>trunc(x)</b>	integral part	$\text{trunc}(-4.58823)=-4.0000$
<b>frac(x)</b>	fractional part	$\text{frac}(-4.58823)=-0.58823$
<b>round(x)</b>	closest integer	$\text{round}(1.65)=2.000$
<b>ceil(x)</b>	smallest larger integer	$\text{ceil}(1.15)=2.000$
<b>floor(x)</b>	largest smaller integer	$\text{floor}(1.78)=1.000$
<b>time()</b>	current simulation time	$\text{time}()=0.1234$
<b>pi()</b>	3.141592...	$\pi()=3.141592...$

Function	Description	Example
<b>twopi()</b>	6.283185...	twopi()=6.283185...
<b>e()</b>	2,718281...	e()=2,718281...

# Bibliography

- [1] IEEE std. c37.010 IEEE Application Guide for AC High-Voltage Circuit Breakers Rated on a Symmetrical Current Basis, 1979.
- [2] IEEE std. c37.5 IEEE Guide for calculation of Fault Currents for Application of AC High-Voltage Circuit Breakers Rated on a Total Current Basis, 1979.
- [3] IEEE std. 242. IEEE Recommended Practice for Protection and Coordination of Industrial and Comercial Power Systems. Buff Book, 1986.
- [4] IEEE std. c37.13 IEEE Standard for Low Voltage Power Circuit Breakers Used in Enclosures, 1990.
- [5] IEEE std. 946. IEEE Recommended Practice for the Design of DC Auxiliary Power Systems for Generating Stations, 1992.
- [6] IEEE std. 141. IEEE Recommended Practice for Electric Power Distribution for Industrial Power Plants. Red Book, 1993.
- [7] DIN VDE 0276-1000 Power cables – Part 1000: Current-carrying capacity, general, conversion factors, 1995.
- [8] IEC 61660-1 Short-circuit currents in d.c. auxiliary installations in power plants and substations – Part 1: Calculation of short-circuit currents, 1997.
- [9] IEC 61363-1 Electrical installations of ships and mobile and fixed offshore units - Part 1: Procedures for calculating short-circuit currents in three-phase a.c., 1998.
- [10] IEC 60076-5 Power transformers - Part 5: Ability to withstand short circuit, 200.
- [11] NF C 15-100 Installations électriques à basse tension, 2002.
- [12] IEC 1000-4-15 Electromagnetic Compatibility (EMC) - Part 4: Testing and measurement techniques - Section 15: Flickemeter - Functional and desing specifications, 2003.
- [13] 60287-3-3 edition 1, 2007.
- [14] D-A-CH-CZ Technical Rules for the Assessment of Network Disturbances, 2007.
- [15] BDEW Technical Guideline for Generating Plants Connected to the Medium Voltage Network, 2008.
- [16] IEC 61000-3-6 Electromagnetic Compatibility (EMC) - Part 3: Limits - Section 6: Assessment of emission limits for distorting loads in MV and HV power systems - Basic EMC publication, 2008.
- [17] IEC 61400-21 Wind turbines - Part 21: Measurement and assessment of power quality characteristics of grid connected wind turbines, 2008.
- [18] IEC 60364-5-52 Electrical installations of buildings – Part 5-52: Selection and erection of electrical equipment - Wiring systems, 2009.
- [19] NF C 13-200 Installations électriques à haute tension, 2009.
- [20] BS 7671 Requirements for Electrical Installations – IET Wiring Regulations, 2011.

## BIBLIOGRAPHY

---

- [21] VDE Power generation systems connected to the low-voltage distribution network, 2011.
- [22] D-A-CH-CZ Technical Rules for the Assessment of Network Disturbances - Extension Document, 2012.
- [23] BDEW Technical Guideline for Generating Plants Connected to the Medium Voltage Network - 4th Supplement, 2013.
- [24] 60287-1-1 edition 2, 2014.
- [25] IEC 60502-2 Power cables with extruded insulation and their accessories for rated voltages from 1 kv ( $U_m = 1,2$  kv) up to 30 kv ( $U_m = 36$  kv) – Part 2: Cables for rated voltages from 6 kv ( $U_m = 7,2$  kv) up to 30 kv ( $U_m = 36$  kv), 2014.
- [26] 60287-2-1 edition 2, 2015.
- [27] IEC 60909 Short-circuit currents in three-phase A.C. systems, 2016.
- [28] VDE-AR-N 4105 G connected to the low-voltage distribution network - Technical requirements for the connection to and parallel operation with low-voltage distribution networks, 2018.
- [29] VDE-AR-N 4110 Technical requirements for the connection and operation of customer installations to the medium voltage network (TAR medium voltage), 2018.
- [30] VDE-AR-N 4100 Technical rules for the connection and operation of customer installations to the low voltage network (TAR low voltage), 2019.
- [31] IEC 61400-27-1 Electrical Simulation Models for Wind Power Generation; Edition 2.0, 2020.
- [32] IEEE std. 946. IEEE Recommended Practice for the Design of DC Power Systems for Stationary Applications, 2020.
- [33] D-A-CH-CZ Technical Rules for the Assessment of Network Disturbances - Part a: Basics, 2021.
- [34] D-A-CH-CZ Technical Rules for the Assessment of Network Disturbances - Part B: Requirements and Assessment - Section I: Low voltage, 2021.
- [35] D-A-CH-CZ Technical Rules for the Assessment of Network Disturbances - Part B: Requirements and Assessment - Section II: Medium voltage, 2021.
- [36] D-A-CH-CZ Technical Rules for the Assessment of Network Disturbances - Part B: Requirements and Assessment - Section III: High voltage, 2023.
- [37] DIN VDE 0298-4 Application of cables and cords in power installations – Part 4: Recommended current-carrying capacity for sheathed and nonsheathed cables for fixed wirings in and around buildings and for flexible cables and cords, 2023.
- [38] VDE-AR-N 4110 Technical requirements for the connection and operation of customer installations to the medium voltage network (TAR medium voltage), 2023.
- [39] Power cable rating examples for calculation tool verification. Technical report, Cigre Working Group B1.56, September 2022.
- [40] Janusz W. Bialek. Tracing the Flow of Electricity. In *IET Proceedings - Generation, Transmission and Distribution*, volume 143, page 313 –320, 1996.
- [41] V. Bolz, J. Rueß, and A. Zell. Power Flow Approximation Based on Graph Convolutional Networks. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 1679–1686, 2019.
- [42] H. Cramér. *Mathematical Methods of Statistics (PMS-9)*, volume 9. Princeton university press, 2016.
- [43] DGUV. Thermische Gefährdung durch Störlichtbögen Hilfe bei der Auswahl der persönlichen Schutzausrüstung. september 2020.
- [44] General Electric. *GE Industrial Power Systems Data Book*. General Electric, 1956.

- [45] EPRI. Arc Flash Issues in Transmission and Substation Environments.
- [46] Ammerman et al. DC-Arc Models and Incident-Energy Calculations. *IEEE Transactions on Industry Applications*, 46(5), Sep/Oct 2010.
- [47] J. VAN DIJK et al. Decomposing power flows in networks with hvdc lines in the fld methodology. *CIGRE Science and Engineering*, 19:20–36, 2020.
- [48] IEEE. IEEE 1584-2002. Guide for Performing Arc-Flash Hazard Calculations.
- [49] IEEE. IEEE 1584-2018. Guide for Performing Arc-Flash Hazard Calculations.
- [50] J. E. Kolassa. *Series approximation methods in statistics*, volume 88. Springer Science & Business Media, 2006.
- [51] R. L. Heinhold. *Kabel und Leitungen für Starkstrom*. Pirelli Kabel und Systeme GmbH & Co, 2005.
- [52] J. H. Neher and M. H. McGrath. The calculation of the temperature rise and load capability of cable systems. *AIEE Transactions*, 76:752–772, 1957.
- [53] NFPA. NFPA 70E. Standard for Electrical Safety. Requirements for Employee Workplaces. 2021 Edition.
- [54] M. Pavesi, J. van Casteren, and S. A. de Graaff. The full line decomposition method - a further development for causation-based cost sharing. *CIGRE Science and Engineering*, 9:27–43, 2017.



# Glossary

**Appliance** A specific physical, installed, power system component: a specific generator, transformer, busbar, etc. Example: a piece of NKBA 0.6/1kV 4 x 35sm cable, 12.4 metres long.

**Base Case** A base case is considered to be the basic power system design, from which one or more alternative designs may be created and analysed.

**Block Definition** A block definition is a mathematical model which may be used in other block definitions or in a composite model. Examples are all default controllers (i.e. VCO's, PSS's, MDM's), and all additional user-defined DSL models. A block definition is called "primitive" when it is directly written in DSL, or "complex" when it is build from other block definitions, by drawing a block diagram.

**Block Diagram** A block diagram is a graphical representation of a DSL model, i.e. a voltage controller, a motor driven machine model or a water turbine model. Block diagrams combine DSL primitive elements and block definitions created by drawing other block diagram. The block models thus created may (again) be used in other block diagrams or to create a Composite Model Frame. See also: DSL primitive, Composite Model Frame

**Branch Element** An element connected to two or more nodes, examples being lines, switches and transformers. See also nodes, edge elements.

**Branch Element (ElmBranch)** Not to be confused with the generic term branch element, the ElmBranch object is a composite two-port element which can contain a number of lines, terminals etc.

**Busbars** Busbars are particular representations of nodes. Busbars are housed in a Station folder and several busbars may be part of a station.

**Class** A class is a template for an element, type or other kind of objects like controller block diagrams, object filters, calculation settings, etc. Examples:

- The 'TypLne' class is the type model for all lines and cables
- The 'ElmLne' class is an element model for a specific line or cable
- The 'ComLdf' class is a load-flow command
- The 'EvtSwitch' class is an event for a switch to open or close during simulation

**Composite Model** A composite model is a specific combination of mathematical models. These models may be power system elements such as synchronous generators, or block definitions, such as voltage controllers, primary mover models or power system stabilisers. Composite models may be used to create new objects, such as protection devices, to 'dress-up' power system elements such as synchronous machines with controllers, prime movers models, etc., or for the identification of model parameters on the basis of measurements.  
See also: Frame, Slot

**Cubicle** A cubicle is the connection point between a edge element and a node (represented by a busbar or terminal). It may be visualised as a bay in a switch yard or a panel in a switchgear board. Elements such as CT's, protection equipment, breakers and so forth, are housed in the cubicle, as one would expect to find in reality.

**DAQ** Abbreviation for “Data Acquisition”.

**Device** A certain kind of physical power system components: certain synchronous machines, two-winding transformers, busbars, or other kinds of equipment. Example: a NKBA 0.6/1kV 4 x 35sm cable.

**DGS** Abbreviation for “*DlgSILENT*Interface for Geographical Information Systems”.

**DOLE** Abbreviation for “*DlgSILENT*Object Language for Data Exchange”. DOLE was used in previous *PowerFactory* versions, but replaced by DGS meanwhile. Now, use DGS instead, please.

The DOLE import uses a header line with the parameter name. This header must have the following structure:

- The first header must be the class name of the listed objects.
- The following headers must state a correct parameter name.

**DPL** Abbreviation for “*DlgSILENT*Programming Language”. For further information, refer to Section 23.1 (The *DlgSILENT*Programming Language - DPL).

**Drag & Drop** “Drag & Drop” is a method for moving an object by left clicking it and subsequently moving the mouse while holding the mouse button down (“dragging”). Releasing the mouse button when the new location is reached is called “dropping”. This will move the object to the new location.

**DSL** Abbreviation for “*DlgSILENT*Simulation Language”. For further information, refer to Section 30.4 (The *DlgSILENT*Simulation Language (DSL)).

**DSL Primitive** A DSL primitive is the same as a primitive block definition. A DSL primitive is written directly in DSL without the use of a block diagram.

Examples are PID controllers, time lags, simple signal filters, integrators, limiters, etc. DSL primitives are normally used to build more complex block definitions.

See also: Block Definition, Block Diagram

**Edge Elements** An element connected to a node or to more than one node. Includes single-port elements such as loads, and multi-port elements such as transformers. See also nodes, branch elements.

**Element** A mathematical model for specific appliances. Most element models only hold the appliance-specific data while the more general type-specific data comes from a type-reference. Example: a model of a piece of NKBA 0.6/1kV 4 x 35sm cable, 12.4 metres long, named “FC 1023.ElmLne”.

**Expansion Stage** An Expansion Stage is part of a network Variation, which includes all changes that apply to the network at a specific activation time.

See also: Variation

**Foreign key** The foreign key is an attribute of all the objects in *PowerFactory* (`e:for_name`). It can be found on the *Description* page of the element’s edit dialog. By default, the attribute is empty and therefore needs to be defined manually. It is recommended to define a foreign key using only letters and numbers, avoiding special characters and blank spaces.

**Frame** A frame is a special block diagram which defines a new stand-alone model, mostly without in- or outputs. A frame is principally a circuit in which one or more slots are connected to each other. A frame is used to create composite models by filling the slots with appropriate objects. The frame thus acts as template for a specific kind of composite models.

See also: Block Diagram, Composite Model, Slot

**Graphic Window** The main graphics window is a docked window that contains one or more graphical pages, in the form of tabs. These pages may be single line diagrams, plots, block diagrams etc., but other pages such as the Data Manager or reports can also be shown here.

The tabs may be used to change the visible page or to change the page order by drag&drop on the page tab. It is also possible to move tabs between docked windows and floating windows.

See also: Plot, Block Diagram, Tab, Drag&Drop

**Grid** A Grid is a collection of power system elements which are all stored in the *Network Data* folder of the project. Normally, a grid forms a logical part of a power system design, like a the MV distribution system in a province, or the HV transport system in a state.

See also: Project

**Node** The mathematical or generic description for what are commonly known as busbars in the electrical world. In *PowerFactory* nodes may be represented by “Busbars” or “Terminals” of various kinds. These are treated in the same manner in mathematical terms but treated slightly differently in the database. As far as possible the user should use terminals as Busbars can be somewhat inflexible.

See also Busbars, Edge Elements, Branch Elements.

**Object** An object is a specific item stored in the database. Examples are specific type or element models which have been edited to model specific devices or appliances. Examples: the element “FC 1023.ElmLne”, the type “NKBA\_4x35.TypLne”, the load-flow command “3Phase.ComLdf”

**Operation Scenario** An Operation Scenario defines a certain operation point of the system under analysis, such as different generation dispatch, low or high load, etc. Operation Scenarios are stored inside the Operation Scenarios folder in the Network Model.

**Plot** A plot is a graphical representation of calculation results. It may be a line or bar graph, a gauge, a vector diagram, etc. A plot gets its values from a results object.

See also: Results Object.

**Project** All power system definitions and calculations are stored and activated in a project. The project folder therefore is a basic folder in the user’s database tree. All grids that make out the power system design, with all design variants, study cases, commands, results, etc. are stored together in a single project folder.

**Results Object** A results object keeps one or more lists of parameters which are to be monitored during a calculation. Results objects are used for building calculation result reports and for defining a virtual instrument.

See also: Plot

**Slot** A slot is a place-holder for a block definition in a composite model frame. A composite model is created from a composite model frame by filling one or more slots with an appropriate object.

See also: Block Definition, Frame, Composite Model

**Study Case** A study case is a folder which stores a list of references or shortcuts to grid, variations or operation scenarios folders. These folders are (de)activated when the calculation case folder is (de)activated.

Elements in the grid folders that are referenced by the study case form the ‘calculation target’ for all calculation functions. Elements in all other, non-active, grid folders are not considered for calculation.

Besides the list of active folders, the calculation case also stores all calculations commands, results, events, and other objects which are, or have been, used to analyse the active power system.

See also: Grid

**Tab** Tabs are small indexes at the edge (mostly on the top) of a multi-page window. The tabs show the titles of the pages. Left-clicking the tab opens the corresponding page. Tabs are used in object dialogs, which often have different pages for different calculation functions, and in the desktop, when more than one graphical page is present.

**Type** A mathematical model for devices: general models for two-winding transformers, two-winding transformers, busbars, etc. A type model only contains the non-specific data valid for whole groups of power system elements. Example: a NKBA 0.6/1kV 4 x 35sm cable type, named “NKBA\_4x35.TypLne”

See also: Grid

**Variation** A Variation defines an expansion plan composed of one or more expansion stages which are chronologically activated. Variations, like all other network data, are stored inside the Network Model folder.

See also: Expansion Stage

# ABOUT DIGSILENT

DIGSILENT was founded in 1985 and is a fully independent and privately owned company located in Gomaringen close to Stuttgart, Germany. DIGSILENT continued expansion by establishing offices in Australia, South Africa, Italy, Chile, Spain, France, the USA and Oman, thereby facilitating improved service following the world-wide increase in usage of its software products and services. DIGSILENT has established a strong partner network in many countries such as Mexico, Malaysia, UK, Colombia, Brazil, Peru, China and India. DIGSILENT services and software installations are used in more than 170 countries.

## POWERFACTORY

DIGSILENT produces the leading integrated power system analysis software PowerFactory, which covers the full range of functionality from standard features to highly sophisticated and advanced applications including wind power, distributed generation, real-time simulation and performance monitoring for system testing and supervision. For various applications, PowerFactory has become the power industry's de-facto standard tool, due to PowerFactory models and algorithms providing unrivalled accuracy and performance.

## STATIONWARE

StationWare is a central asset management system for primary and secondary equipment. In addition to handling locations and devices in a user-definable hierarchy, the system allows manufacturer-independent protection settings to be stored and managed in line with customer-specific workflows. It facilitates the management of a wide variety of business processes within a company and centralises the storage of documents. StationWare can be integrated seamlessly into an existing IT environment and the interface with PowerFactory enables the transfer of calculation-relevant data for protection studies.

## MONITORING SYSTEMS

Our Power System Monitoring PFM300 product line features grid and plant supervision, fault recording, and power quality and grid characteristics analysis. The Grid Code Compliance Monitoring PFM300-GCC system also offers compliance auditing of power plants with respect to grid code requirements. This monitoring and non-compliance detection provides the complete transparency and assurance required by both plant operators and utilities.

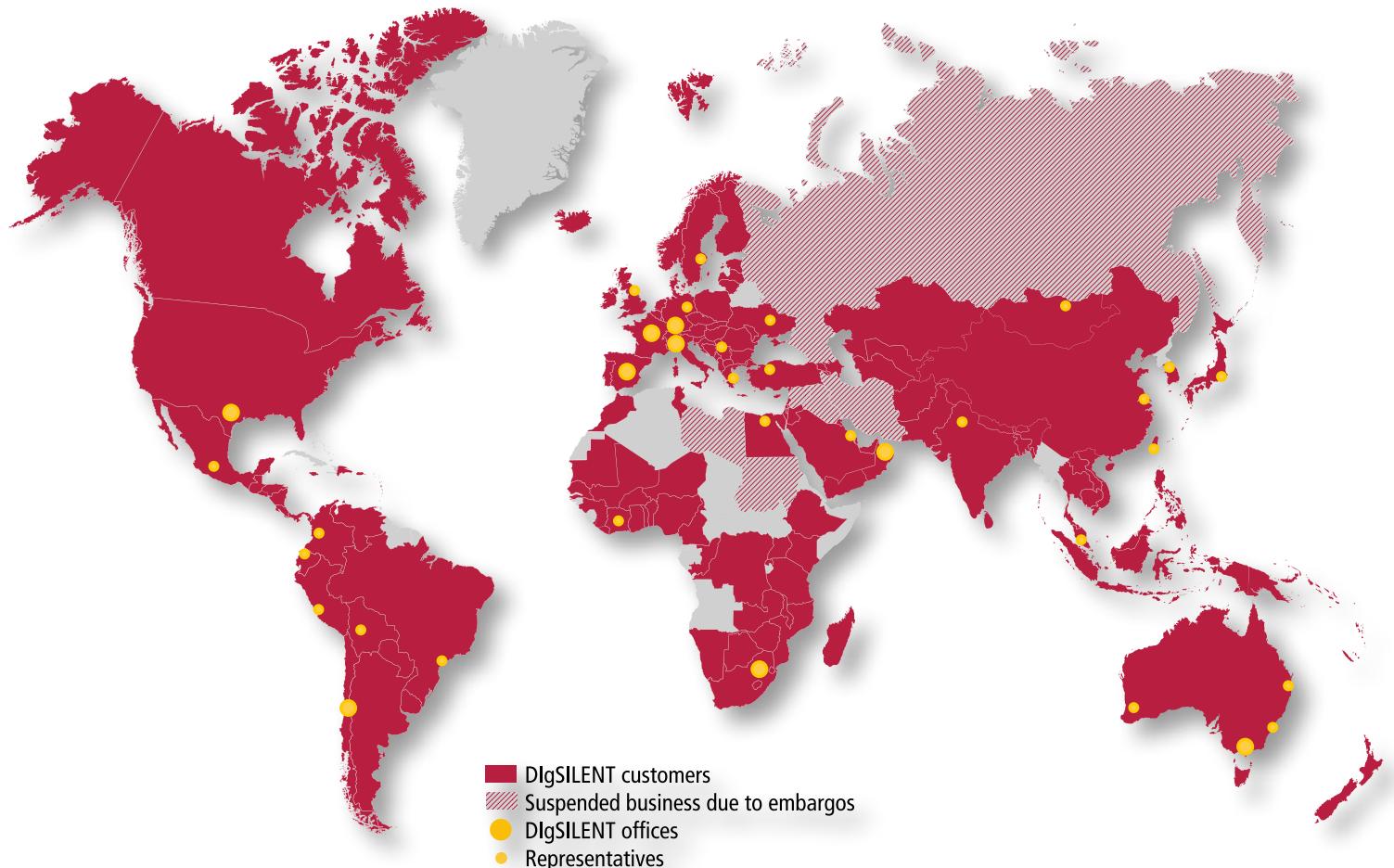
## TESTING AND CERTIFICATION

The DIN EN ISO/IEC 17025 accredited DIGSILENT Test Laboratory for NAR Conformity carries out measurements in accordance with FGW TR3 on the operational type 1 generation plant (directly coupled synchronous machines). These measurements are carried out in accordance with the "individual verification procedure" as required by the German grid connection guidelines VDE-AR-N 4110/20/30. DIGSILENT has many years of international expertise in the field of generation and consumption/load systems testing. The in-house developed and produced measuring systems enable the testing laboratory to offer customised measuring solutions for a wide range of power plants and applications.

## SERVICES

DIGSILENT GmbH is staffed with experts of various disciplines relevant for performing consulting services, research activities, user training, educational programs and software development. Highly specialised expertise is available in many fields of electrical engineering applicable to liberalised power markets and to the latest developments in power generation technologies such as wind power and distributed generation. DIGSILENT has provided expert consulting services to several prominent PV and wind grid integration studies.

# SERVING MORE THAN 170 COUNTRIES



For more information, visit  
[www.digsilent.de](http://www.digsilent.de)



**DLgSILENT GmbH**  
Heinrich-Hertz-Straße 9  
72810 Gomaringen (Germany)  
T: +49 7072 9168-0  
mail@digsilent.de



DLgSILENT GmbH is certified  
to the ISO 9001:2015 standard.  
More information is available at  
[www.tuv-sud.com/ms-cert](http://www.tuv-sud.com/ms-cert)