



# POWERFACTORY

# PowerFactory 2020

# User Manual

PF2020

# **POWER SYSTEM SOLUTIONS**

## MADE IN GERMANY

# **DIGSILENT PowerFactory**

Version 2020

## **User Manual**

---

Online Edition

DIGSILENT GmbH

Gomaringen, Germany

June 2020

**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.

June 2020  
r6805

# 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 . . . . .	4
<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 Global Library . . . . .	14
4.5.2 User Defined Global Library . . . . .	15
4.5.3 Project Library . . . . .	16
4.5.4 Diagrams . . . . .	16
4.5.5 Network Data . . . . .	17
4.5.6 Variations . . . . .	17

4.5.7	Operation Scenarios . . . . .	18
4.5.8	Study Cases . . . . .	18
4.5.9	Settings . . . . .	18
4.6	Project Structure . . . . .	18
4.6.1	Nodes . . . . .	19
4.6.2	Edge elements . . . . .	19
4.6.3	Branches . . . . .	19
4.6.4	Cubicles . . . . .	20
4.6.5	Switches . . . . .	20
4.6.6	Substations . . . . .	20
4.6.7	Secondary Substations . . . . .	20
4.6.8	Sites . . . . .	20
4.6.9	Branch Elements . . . . .	20
4.7	User Interface . . . . .	21
4.7.1	Overview . . . . .	21
4.7.2	Menu Bar . . . . .	23
4.7.3	Main Toolbar . . . . .	24
4.7.4	The Output Window . . . . .	27
4.8	Scripting in <i>PowerFactory</i> . . . . .	30
4.8.1	DlgsILENT Programming Language (DPL) Scripts . . . . .	30
4.8.2	Python Scripts . . . . .	32
<b>II</b>	<b>Administration</b>	<b>33</b>
<b>5</b>	<b>Program Administration</b>	<b>34</b>
5.1	Program Installation and Configuration . . . . .	34
5.2	<i>PowerFactory</i> Configuration . . . . .	35
5.2.1	General Page . . . . .	35
5.2.2	Database Page . . . . .	35
5.2.3	Workspace Page . . . . .	35
5.2.4	External Applications Page . . . . .	35
5.2.5	Network Page . . . . .	36
5.2.6	Geographic Maps Page . . . . .	36

5.2.7	Advanced Page . . . . .	37
5.3	Licence . . . . .	37
5.3.1	Select Licence . . . . .	37
5.3.2	Activate / Update / Deactivate / Move Licence . . . . .	38
5.4	Workspace Options . . . . .	38
5.4.1	Show Workspace Directory . . . . .	38
5.4.2	Import and Export Workspace . . . . .	38
5.4.3	Show Default Export Directory . . . . .	38
5.4.4	Import Workspace from 14.X or 15.0... . . . . .	39
5.5	Offline Mode User Guide . . . . .	39
5.5.1	Functionality in Offline Mode . . . . .	39
5.5.2	Functionality in Online Mode . . . . .	42
5.5.3	Terminate Offline Session . . . . .	42
5.6	Housekeeping . . . . .	42
5.6.1	Introduction . . . . .	42
5.6.2	Configuring Permanently Logged-On Users . . . . .	42
5.6.3	Configuring Housekeeping Tasks . . . . .	42
5.6.4	Project Archiving . . . . .	43
5.6.5	Configuring Deletion of Old Projects . . . . .	43
5.6.6	Configuring Purging of Projects . . . . .	44
5.6.7	Configuring Emptying of Recycle Bins . . . . .	45
5.6.8	Monitoring Housekeeping . . . . .	45
5.6.9	Summary of Housekeeping Deployment . . . . .	45
<b>6</b>	<b>User Accounts, User Groups, and Profiles</b> . . . . .	<b>46</b>
6.1	<i>PowerFactory</i> Database Overview . . . . .	46
6.2	The Database Administrator . . . . .	47
6.3	Administration Menu . . . . .	48
6.3.1	User Management . . . . .	48
6.3.2	Security and Privacy . . . . .	48
6.3.3	Calculation Settings . . . . .	48
6.3.4	Housekeeping . . . . .	48
6.4	Security and Privacy . . . . .	49

6.4.1	Audit Log . . . . .	49
6.4.2	Login Policy Options . . . . .	49
6.4.3	Idle Session Timeout . . . . .	50
6.4.4	External Data Access . . . . .	50
6.4.5	Privacy . . . . .	50
6.5	Creating and Managing User Accounts . . . . .	50
6.6	Creating User Groups . . . . .	52
6.7	User Interface Customisation (Profiles) . . . . .	52
6.7.1	Tool Configuration . . . . .	53
6.7.2	Configuration of Toolbars . . . . .	54
6.7.3	Configuration of Menus . . . . .	56
6.7.4	Configuration of Dialog Pages . . . . .	56
6.7.5	Configuration of Dialog Parameters . . . . .	57
6.7.6	References . . . . .	57
<b>7</b>	<b>User Settings</b> . . . . .	<b>58</b>
7.1	Data/Network Model Manager Settings . . . . .	58
7.2	Window Layout . . . . .	59
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 . . . . .	61
7.8	Colours Settings . . . . .	62
7.9	StationWare Settings . . . . .	62
7.10	Offline Settings . . . . .	63
7.11	Parallel Computing . . . . .	63
7.12	Miscellaneous Settings . . . . .	64

<b>III Handling</b>	<b>66</b>
<b>8 Basic Project Definition</b>	<b>67</b>
8.1 Defining and Configuring a Project . . . . .	67
8.1.1 Project Dialog . . . . .	69
8.1.2 Project Settings . . . . .	70
8.1.3 Activating and Deactivating Projects . . . . .	73
8.1.4 Exporting and Importing Projects . . . . .	74
8.1.5 External References . . . . .	76
8.1.6 Including Additional Documents . . . . .	76
8.2 Creating New Grids . . . . .	76
8.3 Project Overview . . . . .	77
<b>9 Network Graphics</b>	<b>80</b>
9.1 Introduction . . . . .	80
9.2 Graphic Windows and Database Objects . . . . .	80
9.2.1 Network Diagrams and other graphics . . . . .	80
9.2.2 Active Graphics, Graphics Board and Study Cases . . . . .	81
9.2.3 Single Line Graphics and Data Objects . . . . .	82
9.2.4 Creating New Graphic Windows . . . . .	83
9.2.5 Page Tab . . . . .	83
9.2.6 Drawing Tools . . . . .	84
9.2.7 Active Grid Folder (Target Folder) . . . . .	85
9.3 Graphic Commands, Options, and Settings . . . . .	85
9.3.1 Freeze Mode . . . . .	85
9.3.2 Rebuild . . . . .	86
9.3.3 View commands . . . . .	86
9.3.4 Select commands . . . . .	87
9.3.5 Graphic Options . . . . .	87
9.3.6 Layers . . . . .	91
9.3.7 Colouring Options . . . . .	95
9.3.8 Graphic Legends . . . . .	99
9.3.9 Node Default Options . . . . .	100

9.3.10	Page, Print and Export Options . . . . .	100
9.3.11	Diagram Layout Tool . . . . .	101
9.3.12	Insert New Graphic . . . . .	101
9.3.13	Element Options . . . . .	102
9.4	Editing and Changing Symbols of Elements . . . . .	103
9.5	Result Boxes, Text Boxes and Labels . . . . .	103
9.5.1	Result Boxes . . . . .	104
9.5.2	Text Boxes . . . . .	106
9.5.3	Labels . . . . .	106
9.5.4	Free Text Labels . . . . .	106
9.6	Annotations . . . . .	106
9.7	Annotation of Protection Device . . . . .	107
9.8	Navigation Pane . . . . .	107
9.9	Graphic search facility . . . . .	108
9.10	Geographic Diagrams . . . . .	108
9.10.1	Using an External Map Provider . . . . .	110
9.10.2	Using Local Maps . . . . .	113
<b>10</b>	<b>Data Manager</b> . . . . .	<b>114</b>
10.1	Introduction . . . . .	114
10.2	Using the Data Manager . . . . .	114
10.2.1	Navigating the Database Tree . . . . .	116
10.2.2	Adding New Items . . . . .	117
10.2.3	Deleting an Item . . . . .	118
10.2.4	Cut, Copy, Paste and Move Objects . . . . .	118
10.2.5	The Data Manager Status Bar . . . . .	119
10.2.6	Additional Features . . . . .	119
10.3	Searching for Objects in the Data Manager . . . . .	120
10.3.1	Sorting Objects . . . . .	120
10.3.2	Searching by Name . . . . .	121
10.3.3	Using Filters for Search . . . . .	121
10.4	Auto-Filter functions in Data Manager and browser windows . . . . .	123
10.5	Editing Data Objects in the Data Manager . . . . .	124

10.5.1	Editing in Object Mode . . . . .	124
10.5.2	Editing in “Detail” Mode . . . . .	124
10.5.3	Copy and Paste while Editing . . . . .	126
10.6	The Flexible Data Page . . . . .	126
10.6.1	Customising the Flexible Data Page . . . . .	127
10.7	The Input Window in the Data Manager . . . . .	129
10.7.1	Input Window Commands . . . . .	129
10.8	Save and Restore Parts of the Database . . . . .	130
10.8.1	Notes . . . . .	130
10.9	Spreadsheet Format Data Import/Export . . . . .	131
10.9.1	Export to Spreadsheet Programs (e. g. MS EXCEL) . . . . .	131
10.9.2	Import from Spreadsheet Programs (e. g. MS EXCEL) . . . . .	132
<b>11</b>	<b>Building Networks</b> . . . . .	<b>137</b>
11.1	Introduction . . . . .	137
11.2	Defining Network Models using the Graphical Editor . . . . .	137
11.2.1	Adding New Power System Elements . . . . .	137
11.2.2	Nodes . . . . .	138
11.2.3	Edge Elements . . . . .	138
11.2.4	Cubicles . . . . .	139
11.2.5	Marking and Editing Power System Elements . . . . .	140
11.2.6	Interconnecting Power Subsystems . . . . .	141
11.2.7	Substations . . . . .	142
11.2.8	Sites . . . . .	146
11.2.9	Composite Branches . . . . .	147
11.2.10	Single and Two Phase Elements . . . . .	147
11.3	Lines and Cables . . . . .	148
11.3.1	Defining a Line ( <i>ElmLne</i> ) . . . . .	149
11.3.2	Defining Line Sections . . . . .	150
11.3.3	Defining Line Couplings . . . . .	151
11.3.4	Defining Cable Systems . . . . .	153
11.4	Neutral Winding Connection in Network Diagrams . . . . .	158
11.5	Defining Network Models using the Data Manager . . . . .	160

11.5.1	Defining New Network Components in the Data Manager . . . . .	160
11.5.2	Connecting Network Components in the Data Manager . . . . .	160
11.5.3	Defining Substations in the Data Manager . . . . .	160
11.5.4	Defining Composite Branches in the Data Manager . . . . .	161
11.5.5	Defining Sites in the Data Manager . . . . .	162
11.6	Drawing Existing Elements using the Diagram Layout Tool . . . . .	162
11.6.1	Action . . . . .	162
11.6.2	Node Layout . . . . .	165
11.6.3	Edge Elements . . . . .	165
11.6.4	Protection Devices . . . . .	166
11.6.5	Bays and Sites . . . . .	167
11.6.6	Interchanges . . . . .	167
11.7	Drawing Existing Elements using Drag & Drop . . . . .	167
<b>12</b>	<b>Network Model Manager</b>	<b>169</b>
12.1	Introduction . . . . .	169
12.2	Using the Network Model Manager . . . . .	169
<b>13</b>	<b>Study Cases</b>	<b>172</b>
13.1	Introduction . . . . .	172
13.2	Creating and Using Study Cases . . . . .	173
13.2.1	The Study Case Manager . . . . .	174
13.3	Summary Grid . . . . .	175
13.4	Study Time . . . . .	175
13.5	The Study Case Dialog . . . . .	176
13.6	Variation Configuration . . . . .	177
13.7	Operation Scenarios . . . . .	177
13.8	Commands . . . . .	177
13.9	Events . . . . .	178
13.9.1	Dispatch Event . . . . .	178
13.9.2	External Measurement Event . . . . .	178
13.9.3	Inter-Circuit Fault Events . . . . .	179
13.9.4	Events of Loads . . . . .	179

---

13.9.5 Message Event . . . . .	179
13.9.6 Outage of Element ( <i>EvtOutage</i> ) . . . . .	179
13.9.7 Parameter Events . . . . .	180
13.9.8 Save Results . . . . .	180
13.9.9 <i>Save Snapshot</i> event . . . . .	180
13.9.10 Short-Circuit Events . . . . .	180
13.9.11 Stop Events . . . . .	180
13.9.12 Switch Events . . . . .	181
13.9.13 Synchronous Machine Event . . . . .	181
13.9.14 Tap Event . . . . .	181
13.9.15 Power Transfer Event . . . . .	181
13.10 Simulation Scan . . . . .	182
13.11 Results Objects . . . . .	182
13.12 Triggers . . . . .	182
13.13 Graphics Board . . . . .	182
<b>14 Project Library</b> . . . . .	<b>184</b>
14.1 Introduction . . . . .	184
14.2 Equipment Type Library . . . . .	184
14.3 Operational Library . . . . .	186
14.3.1 Circuit Breaker Ratings . . . . .	186
14.3.2 Characteristics . . . . .	188
14.3.3 Demand Transfers . . . . .	188
14.3.4 Fault Cases and Fault Groups . . . . .	189
14.3.5 Capability Curves (Mvar Limit Curves) for Generators . . . . .	190
14.3.6 Planned Outages . . . . .	191
14.3.7 Planned Outages <i>IntOutage</i> . . . . .	192
14.3.8 QP-Curves . . . . .	194
14.3.9 Remedial Action Schemes (RAS) . . . . .	194
14.3.10 Running Arrangements . . . . .	194
14.3.11 Thermal Ratings . . . . .	196
14.3.12 V-Control-Curves . . . . .	197
14.4 Templates Library . . . . .	197

---

14.4.1 General Templates . . . . .	198
14.4.2 Substation Templates . . . . .	199
14.4.3 Busbar Templates . . . . .	199
14.4.4 Composite Branch Templates . . . . .	199
14.4.5 Example Power Plant Template . . . . .	199
14.4.6 Wind Turbine Templates according to IEC 61400-27-1 . . . . .	200
<b>15 Grouping Objects</b>	<b>201</b>
15.1 Areas . . . . .	201
15.2 Virtual Power Plants . . . . .	202
15.2.1 Defining and Editing a New Virtual Power Plant . . . . .	202
15.2.2 Applying a Virtual Power Plant . . . . .	203
15.2.3 Inserting a Generator into a Virtual Power Plant and Defining its Virtual Power Plant Properties . . . . .	203
15.3 Boundaries . . . . .	203
15.3.1 Boundary Definition Tool . . . . .	204
15.3.2 Element Boundary . . . . .	205
15.4 Circuits . . . . .	206
15.5 Feeders . . . . .	206
15.5.1 Feeder Tools . . . . .	209
15.6 Meteo Stations . . . . .	213
15.7 Operators . . . . .	213
15.8 Owners . . . . .	214
15.9 Paths . . . . .	214
15.10 Routes . . . . .	215
15.11 Zones . . . . .	215
<b>16 Operation Scenarios</b>	<b>216</b>
16.1 Introduction . . . . .	216
16.2 Operation Scenarios Background . . . . .	216
16.3 How to use Operation Scenarios . . . . .	217
16.3.1 How to create an Operation Scenario . . . . .	218
16.3.2 How to save an Operation Scenario . . . . .	218
16.3.3 How to activate an existing Operation Scenario . . . . .	220

16.3.4 How to deactivate an Operation Scenario . . . . .	220
16.3.5 How to identify operational data parameters . . . . .	220
16.4 Administering Operation Scenarios . . . . .	221
16.4.1 How to view objects missing from the Operation Scenario data . . . . .	222
16.4.2 How to compare the data in two operation scenarios . . . . .	222
16.4.3 How to view the non-default Running Arrangements . . . . .	222
16.4.4 How to transfer data from one Operation Scenario to another . . . . .	222
16.4.5 How to update the default data with operation scenario data . . . . .	224
16.4.6 How exclude a grid from the Operation Scenario data . . . . .	224
16.4.7 How to create a time based Operation Scenario . . . . .	225
16.5 Advanced Configuration of Operation Scenarios . . . . .	226
16.5.1 How to change the automatic save settings for Operation Scenarios . . . . .	226
16.5.2 How to modify the data stored in Operation Scenarios . . . . .	226
<b>17 Network Variations and Expansion Stages</b>	<b>228</b>
17.1 Introduction . . . . .	228
17.2 Variations . . . . .	229
17.3 Expansion Stages . . . . .	230
17.4 The Study Time . . . . .	230
17.5 The Recording Expansion Stage . . . . .	231
17.6 The Variation Scheduler . . . . .	231
17.7 Variation and Expansion Stage Example . . . . .	231
17.8 Variation and Expansion Stage Housekeeping . . . . .	233
17.8.1 Applying Changes from Expansion Stages . . . . .	233
17.8.2 Consolidating Variations . . . . .	233
17.8.3 Splitting Expansion Stages . . . . .	234
17.8.4 Comparing Variations and Expansion Stages . . . . .	234
17.8.5 Colouring Variations the Single Line Graphic . . . . .	235
17.8.6 Variation Conflicts . . . . .	236
17.8.7 Error Correction Mode . . . . .	236
17.9 Compatibility with Previous Releases . . . . .	237
17.9.1 General . . . . .	237
17.9.2 Converting System Stages . . . . .	238

---

<b>18 Parameter Characteristics, Load States, and Tariffs</b>	<b>241</b>
18.1 Introduction . . . . .	241
18.2 Parameter Characteristics . . . . .	241
18.2.1 Time Characteristics . . . . .	244
18.2.2 Profile Characteristics . . . . .	247
18.2.3 Scaling Factor . . . . .	248
18.2.4 Linear Functions . . . . .	249
18.2.5 Vector Characteristics . . . . .	249
18.2.6 Matrix Parameter Characteristics . . . . .	250
18.2.7 Parameter Characteristics from Files . . . . .	251
18.2.8 Characteristic References . . . . .	251
18.2.9 Edit Characteristic Dialog . . . . .	252
18.2.10 Characteristics Tab in Data Filters . . . . .	252
18.2.11 Example Application of Characteristics . . . . .	253
18.3 Load States . . . . .	255
18.3.1 Creating Load States . . . . .	256
18.3.2 Viewing Existing Load States . . . . .	256
18.3.3 Load State Object Properties . . . . .	257
18.3.4 Example Load States . . . . .	257
18.4 Load Distribution States . . . . .	259
18.4.1 Creating Load Distribution States . . . . .	259
18.4.2 Viewing Existing Load Distribution States . . . . .	259
18.4.3 Load Distribution State Object Properties . . . . .	259
18.4.4 Example Load Distribution States . . . . .	260
18.5 Tariffs . . . . .	261
18.5.1 Defining Time Tariffs . . . . .	261
18.5.2 Defining Energy Tariffs . . . . .	262
<b>19 Reporting and Visualising Results</b>	<b>264</b>
19.1 Introduction . . . . .	264
19.2 Result Boxes . . . . .	264
19.2.1 Editing Result Boxes . . . . .	264
19.3 Variable Selection . . . . .	266

---

19.3.1 Variable Selection Filter . . . . .	268
19.4 Output Reports . . . . .	269
19.4.1 Documentation of Device Data . . . . .	269
19.4.2 Output of Results . . . . .	270
19.5 Comparisons Between Calculations . . . . .	270
19.6 Results Objects . . . . .	271
19.6.1 Exporting Results . . . . .	272
19.7 Plots . . . . .	273
19.7.1 Curve Plot . . . . .	276
19.7.2 Curve Plot (2 y-axes) . . . . .	280
19.7.3 Curve Plot (multiple x-axes) . . . . .	281
19.7.4 Binary Bar Plot . . . . .	283
19.7.5 Specifying Curves for Plots . . . . .	283
19.7.6 Bar Plot . . . . .	284
19.7.7 Vector Plot . . . . .	285
19.7.8 Curve-Input Plot . . . . .	286
19.7.9 Virtual Instruments . . . . .	288
19.7.10 Voltage Profile Plot . . . . .	289
19.7.11 Duration curve . . . . .	290
19.7.12 Plant Categories . . . . .	291
19.7.13 Complete Generation . . . . .	293
19.7.14 Renewable and Fossil . . . . .	293
19.7.15 Plots Toolbar . . . . .	294
19.7.16 Context Sensitive Menu Tools . . . . .	300
19.7.17 The Status Bar . . . . .	301
19.7.18 User-Defined Styles . . . . .	302
<b>20 Data Extensions</b>	<b>304</b>
20.1 Introduction . . . . .	304
20.2 Data Extension Configuration . . . . .	304
20.2.1 Creating Data Extensions . . . . .	304
20.2.2 User Defined Classes . . . . .	305
20.3 Using Data Extensions . . . . .	305

---

20.4 Sharing Data Extensions . . . . .	305
<b>21 Data Management</b>	<b>307</b>
21.1 Introduction . . . . .	307
21.2 Project Versions . . . . .	307
21.2.1 What is a Version? . . . . .	307
21.2.2 How to Create a Version . . . . .	308
21.2.3 How to Rollback a Project . . . . .	309
21.2.4 How to Check if a Version is the Base for a Derived Project . . . . .	309
21.2.5 How to Delete a Version . . . . .	310
21.3 Derived Projects . . . . .	310
21.3.1 Derived Projects Background . . . . .	310
21.3.2 How to Create a Derived Project . . . . .	312
21.4 Comparing and Merging Projects . . . . .	313
21.4.1 Compare and Merge Tool Background . . . . .	313
21.4.2 How to Merge or Compare Two Projects Using the Compare and Merge Tool . . . . .	313
21.4.3 How to Merge or Compare Three Projects Using the Compare and Merge Tool . . . . .	314
21.4.4 Compare and Merge Tool Advanced Options . . . . .	315
21.4.5 Compare and Merge Tool 'diff browser' . . . . .	316
21.5 How to Update a Project . . . . .	322
21.5.1 Updating a Derived Project from a new Version . . . . .	322
21.5.2 Updating a Base Project from a Derived Project . . . . .	323
21.5.3 Tips for Working with the Compare and Merge Tool . . . . .	323
21.6 Sharing Projects . . . . .	324
21.7 Combining Projects . . . . .	325
21.7.1 Project Combination Assistant . . . . .	325
21.7.2 Project Connection Assistant . . . . .	326
21.7.3 Final Project State . . . . .	327
21.7.4 Project Normalisation . . . . .	328
21.8 Database Archiving . . . . .	328
<b>22 Task Automation</b>	<b>329</b>
22.1 Introduction . . . . .	329

22.2 Configuration of Task Automation . . . . .	329
22.2.1 Basic Options Page . . . . .	330
22.2.2 Parallel Computing Page . . . . .	331
22.2.3 Output Page . . . . .	332
22.3 Task Automation Results . . . . .	332
22.4 Parallel Computing Manager . . . . .	333
22.4.1 Basic Options Page . . . . .	334
22.4.2 Communication page . . . . .	334
<b>23 Scripting</b>	<b>335</b>
23.1 The DIgSILENT Programming Language - DPL . . . . .	335
23.1.1 The Principle Structure of a DPL Command . . . . .	336
23.1.2 The DPL Command . . . . .	337
23.1.3 The DPL Script Editor . . . . .	339
23.1.4 The DPL Script Language . . . . .	339
23.1.5 Access to Other Objects . . . . .	345
23.1.6 Access to Locally Stored Objects . . . . .	346
23.1.7 Accessing the General Selection . . . . .	346
23.1.8 Accessing External Objects . . . . .	347
23.1.9 Remote Scripts and DPL Command Libraries . . . . .	348
23.1.10 DPL Functions and Subroutines . . . . .	351
23.2 Tabular Reports . . . . .	351
23.2.1 Basic Structure of a Tabular Report . . . . .	351
23.2.2 The Table Report Command . . . . .	352
23.2.3 A minimal Tabular Report . . . . .	353
23.2.4 Handling different kinds of data . . . . .	354
23.2.5 Advanced Features . . . . .	354
23.2.6 Table Report Callback Script Reference . . . . .	358
23.3 Python . . . . .	359
23.3.1 Installation of a Python Interpreter . . . . .	360
23.3.2 The Python <i>PowerFactory</i> Module . . . . .	360
23.3.3 The Python Command ( <i>ComPython</i> ) . . . . .	362
23.3.4 Running <i>PowerFactory</i> in Non-interactive Mode . . . . .	364

23.3.5 Performance of Python Scripts . . . . .	365
23.3.6 Debugging Python Scripts . . . . .	365
23.3.7 Example of a Python Script . . . . .	366
23.4 Editor . . . . .	367
23.5 Add On Modules . . . . .	368
23.5.1 Add On Module framework . . . . .	368
23.5.2 Creating a new Add-on Module command . . . . .	369
23.5.3 Executing an Add-on Module command . . . . .	371
23.5.4 Adding Add On Modules to the User-Defined Tools toolbar . . . . .	372
<b>24 Interfaces</b> . . . . .	<b>373</b>
24.1 Introduction . . . . .	373
24.2 DGS Interface . . . . .	373
24.2.1 DGS Interface Typical Applications . . . . .	374
24.2.2 DGS Structure (Database Schemas and File Formats) . . . . .	375
24.2.3 DGS Import . . . . .	375
24.2.4 DGS Export . . . . .	377
24.3 PSS/E File Interface . . . . .	379
24.3.1 PSS/E File Types and Versions . . . . .	379
24.3.2 Importing PSS/E Steady-State Data . . . . .	379
24.3.3 Import of PSS/E file (Dynamic Data) . . . . .	381
24.3.4 Exporting a project to a PSS/E file . . . . .	382
24.4 ELEKTRA Interface . . . . .	383
24.4.1 Import of Elektra Data . . . . .	383
24.4.2 General Settings . . . . .	384
24.4.3 Advanced Settings . . . . .	384
24.4.4 Importing Elektra Network Data . . . . .	385
24.4.5 Importing Elektra Type Data . . . . .	385
24.4.6 Output Window . . . . .	386
24.5 NEPLAN Interface . . . . .	386
24.5.1 Importing NEPLAN Data . . . . .	386
24.6 INTEGRAL Interface . . . . .	387
24.6.1 Importing Integral Data . . . . .	388

---

24.6.2 Export Integral Data . . . . .	388
24.7 PSS SINCAL Interface . . . . .	388
24.7.1 Importing PSS SINCAL Data . . . . .	388
24.8 UCTE-DEF Interface . . . . .	389
24.8.1 Importing UCTE-DEF Data . . . . .	389
24.8.2 Exporting UCTE-DEF Data . . . . .	390
24.9 CIM Interface . . . . .	391
24.9.1 Importing CIM Data . . . . .	391
24.9.2 General Page . . . . .	391
24.9.3 Exporting CIM Data . . . . .	391
24.10 CGMES Tools . . . . .	392
24.10.1 CIM Data Import . . . . .	392
24.10.2 CIM Data Export . . . . .	393
24.10.3 CIM to Grid Conversion . . . . .	393
24.10.4 Grid to CIM Conversion . . . . .	394
24.10.5 CIM Data Validation . . . . .	395
24.10.6 Import and Export of the EIC as additional parameter . . . . .	395
24.11 MATLAB Interface . . . . .	395
24.12 OPC Interface . . . . .	396
24.12.1 OPC Interface Typical Applications . . . . .	396
24.13 StationWare Interface . . . . .	397
24.13.1 About StationWare . . . . .	397
24.13.2 Component Architecture . . . . .	397
24.13.3 Fundamental Concepts . . . . .	399
24.13.4 Configuration . . . . .	403
24.13.5 Getting Started . . . . .	403
24.13.6 Description of the Menu and Dialogs . . . . .	414
24.14 API (Application Programming Interface) . . . . .	418
<b>IV Power System Analysis Functions</b>	<b>419</b>
<b>25 Load Flow Analysis</b>	<b>420</b>
25.1 Introduction . . . . .	420

---

---

25.2 Technical Background . . . . .	422
25.2.1 Network Representation and Calculation Methods . . . . .	424
25.3 Executing Load Flow Calculations . . . . .	427
25.3.1 Basic Options . . . . .	427
25.3.2 Active Power Control . . . . .	428
25.3.3 Advanced Options . . . . .	432
25.3.4 Calculation Settings . . . . .	434
25.3.5 Outputs . . . . .	436
25.3.6 Load/Generation Scaling . . . . .	437
25.3.7 Low Voltage Analysis . . . . .	438
25.4 Detailed Description of Load Flow Calculation Options . . . . .	439
25.4.1 Active and Reactive Power Control . . . . .	439
25.4.2 Voltage Dependency of Loads . . . . .	444
25.4.3 Feeder Load Scaling . . . . .	445
25.4.4 Coincidence of Low Voltage Loads . . . . .	450
25.4.5 Temperature Dependency of Lines and Cables . . . . .	451
25.5 Results Analysis . . . . .	452
25.5.1 Viewing Results in the Single Line Diagram . . . . .	452
25.5.2 Flexible Data Page . . . . .	452
25.5.3 Predefined Report Formats (ASCII Reports) . . . . .	452
25.5.4 Diagram Colouring . . . . .	453
25.5.5 Load Flow Sign Convention . . . . .	453
25.5.6 Results for Unbalanced Load Flow Calculations . . . . .	454
25.5.7 Update Database . . . . .	454
25.6 Troubleshooting Load Flow Calculation Problems . . . . .	455
25.6.1 General Troubleshooting . . . . .	455
25.6.2 Data Model Problem . . . . .	456
25.6.3 Some Load Flow Calculation Messages . . . . .	457
25.6.4 Too many Inner Loop Iterations . . . . .	457
25.6.5 Too Many Outer Loop Iterations . . . . .	458
<b>26 Short-Circuit Analysis</b>	<b>461</b>
26.1 Introduction . . . . .	461

---

---

26.2 Technical Background . . . . .	462
26.2.1 The IEC 60909/VDE 0102 Method . . . . .	464
26.2.2 The ANSI Method . . . . .	467
26.2.3 The Complete Method . . . . .	469
26.2.4 The IEC 61363 Method . . . . .	470
26.2.5 The IEC 61660 (DC) Method . . . . .	471
26.2.6 The ANSI/IEEE 946 (DC) Method . . . . .	473
26.3 Executing Short-Circuit Calculations . . . . .	473
26.3.1 Toolbar/Main Menu Execution . . . . .	473
26.3.2 Context-Sensitive Menu Execution . . . . .	474
26.3.3 Faults on Busbars/Terminals . . . . .	474
26.3.4 Faults on Lines and Branches . . . . .	475
26.3.5 Multiple Faults Calculation . . . . .	475
26.4 Short-Circuit Calculation Options . . . . .	477
26.4.1 Basic Options (All Methods) . . . . .	477
26.4.2 Verification (Except for IEC 61363, IEC 61660 and ANSI/IEEE 946) . . . . .	480
26.4.3 Basic Options (IEC 60909/VDE 0102 Method) . . . . .	480
26.4.4 Advanced Options (IEC 60909/VDE 0102 Method) . . . . .	481
26.4.5 Basic Options (ANSI C37 Method) . . . . .	483
26.4.6 Advanced Options (ANSI C37 Method) . . . . .	484
26.4.7 Basic Options (Complete Method) . . . . .	485
26.4.8 Advanced Options (Complete Method) . . . . .	486
26.4.9 Basic Options (IEC 61363) . . . . .	488
26.4.10 Advanced Options (IEC 61363) . . . . .	489
26.4.11 Basic Options (IEC 61660 Method) . . . . .	489
26.4.12 Advanced Options (IEC 61660 Method) . . . . .	490
26.4.13 Basic Options (ANSI/IEEE 946 Method) . . . . .	490
26.4.14 Advanced Options (ANSI/IEEE 946 Method) . . . . .	491
26.5 Results Analysis . . . . .	491
26.5.1 Viewing Results in the Single Line Diagram . . . . .	491
26.5.2 Flexible Data Page . . . . .	491
26.5.3 Predefined Report Formats (ASCII Reports) . . . . .	492

---

26.5.4 Diagram Colouring . . . . .	492
26.6 Capacitive Earth-Fault Current . . . . .	493
<b>27 Contingency Analysis</b>	<b>494</b>
27.1 Introduction . . . . .	494
27.2 Short Overview . . . . .	494
27.2.1 Contingency Analysis Objects . . . . .	495
27.2.2 Results Recording . . . . .	496
27.2.3 Configuring Network Restoration . . . . .	496
27.2.4 Visualisation . . . . .	496
27.3 Contingency Analysis Toolbar . . . . .	497
27.3.1 Contingency Definition . . . . .	497
27.3.2 Contingency Analysis Command . . . . .	497
27.3.3 Contingency Comparison . . . . .	497
27.3.4 Show Contingencies . . . . .	498
27.3.5 Show Fault Cases / Groups . . . . .	498
27.3.6 Remedial Action Schemes . . . . .	498
27.3.7 Edit Results Variables . . . . .	498
27.3.8 Tracing Buttons . . . . .	498
27.3.9 Contingency Analysis Reports . . . . .	498
27.3.10 Load Contingency Analysis Results . . . . .	498
27.4 Command dialog and Options . . . . .	498
27.4.1 Basic Options . . . . .	498
27.4.2 Recording of Results . . . . .	500
27.4.3 Time Phases . . . . .	501
27.4.4 Effectiveness . . . . .	504
27.4.5 Time Sweep . . . . .	505
27.4.6 Topology . . . . .	506
27.4.7 Screening . . . . .	507
27.4.8 Output . . . . .	507
27.4.9 Linearised Calculation . . . . .	508
27.4.10 Parallel Computing . . . . .	508
27.4.11 Calculating an Individual Contingency . . . . .	508

27.4.12 Representing Contingency Situations	
Contingency Cases . . . . .	509
27.5 Reporting Results . . . . .	510
27.5.1 Predefined Reports . . . . .	510
27.5.2 Customised reports . . . . .	513
27.6 Trace Function for Multiple Time Phase and/or RAS . . . . .	513
27.7 Creating Contingencies . . . . .	514
27.7.1 Creating Contingencies Using the Contingency Definition Command . . . . .	514
27.7.2 Creating Contingencies Using Fault Cases and Groups . . . . .	515
27.7.3 Creating Dynamic Contingencies . . . . .	516
27.8 Fault Cases and Groups . . . . .	516
27.8.1 Browsing Fault Cases and Fault Groups . . . . .	517
27.8.2 Defining a Fault Case from Network Element(s) . . . . .	518
27.8.3 Defining Fault Cases using the Contingency Definition Command . . . . .	518
27.8.4 Representing Contingency Situations with Post-Fault Actions . . . . .	519
27.8.5 Defining a Fault Group . . . . .	520
27.9 Comparing Contingency Results . . . . .	520
27.10 Managing variables to be recorded . . . . .	521
27.10.1 Using filters to enable selective results recording . . . . .	522
27.11 Remedial Action Schemes (RAS) . . . . .	522
27.11.1 Creating a RAS object . . . . .	522
27.11.2 Trigger Conditions . . . . .	523
27.11.3 Logical combinations of triggers . . . . .	523
27.11.4 Remedial actions . . . . .	523
27.11.5 RAS groups . . . . .	524
27.11.6 Using Remedial Action Schemes in Contingency Analysis . . . . .	524
27.11.7 Results and reporting . . . . .	524
27.11.8 Visualising RAS using the Trace Function . . . . .	526
27.12 Load Contingency Analysis Results . . . . .	527
27.12.1 Load Individual Contingencies . . . . .	527
<b>28 Quasi-Dynamic Simulation</b>	<b>528</b>

---

28.1	Introduction . . . . .	528
28.2	Technical background . . . . .	528
28.3	How to execute a Quasi-Dynamic Simulation . . . . .	530
28.3.1	Defining the variables for monitoring in the Quasi-Dynamic simulation . . . . .	531
28.3.2	Considering maintenance outages . . . . .	531
28.3.3	Considering simulation events . . . . .	532
28.3.4	Running the Quasi-Dynamic simulation . . . . .	533
28.3.5	Configuring the Quasi-Dynamic Simulation for parallel computation . . . . .	535
28.3.6	Configure the Quasi-Dynamic Simulation for real time simulation . . . . .	535
28.4	Analysing the QDS results . . . . .	536
28.4.1	Plotting . . . . .	536
28.4.2	Quasi-Dynamic simulation reports . . . . .	536
28.4.3	Statistical summary of monitored variables . . . . .	537
28.4.4	Loading Results . . . . .	537
28.5	Developing QDSL models . . . . .	537
28.5.1	<i>PowerFactory</i> objects for implementing user defined models . . . . .	538
28.5.2	Overview of modelling approach . . . . .	541
28.5.3	Algorithm flow of user defined Quasi-Dynamic Simulation models . . . . .	542
28.5.4	Scripting Functions for Quasi-Dynamic Simulation . . . . .	545
28.5.5	Example: Modelling a battery as a <i>Quasi-Dynamic</i> user defined model . . . . .	550
28.5.6	QDSL Model Encryption . . . . .	557
<b>29</b>	<b>RMS/EMT Simulations</b>	<b>558</b>
29.1	Introduction . . . . .	558
29.2	Calculation Methods . . . . .	560
29.2.1	Balanced RMS Simulation . . . . .	560
29.2.2	Three-Phase RMS Simulation . . . . .	560
29.2.3	Three-Phase EMT Simulation . . . . .	561
29.3	<i>Calculation of Initial Conditions</i> command . . . . .	561
29.3.1	Initial Conditions - Basic Options . . . . .	562
29.3.2	Initial Conditions - Step Size . . . . .	563
29.3.3	Initial Conditions - Solver Options . . . . .	565
29.3.4	Initial Conditions - Simulation Scan . . . . .	568

---

29.3.5 Initial Conditions - Noise Generation . . . . .	569
29.3.6 Initial Conditions - Snapshot . . . . .	569
29.3.7 Advanced Simulation Options - Load Flow . . . . .	569
29.4 Results Objects . . . . .	570
29.4.1 Saving Results from Previous Simulations . . . . .	572
29.5 Simulation Events . . . . .	572
29.6 Executing the Simulation . . . . .	575
29.7 Creating Simulation Plots . . . . .	576
29.8 Simulation Scan . . . . .	576
29.8.1 Fault Ride Through Scan Module . . . . .	576
29.8.2 Frequency Scan Module . . . . .	578
29.8.3 Loss of Synchronism Scan Module . . . . .	579
29.8.4 Synchronous Machine Speed Scan Module . . . . .	580
29.8.5 Variable Scan Module . . . . .	581
29.8.6 Voltage Scan Module . . . . .	581
29.9 Save/Load Snapshot . . . . .	583
29.9.1 Saving a snapshot . . . . .	583
29.9.2 Loading a snapshot . . . . .	583
29.10 Single/Multiple Domain Co-simulation . . . . .	584
29.10.1 Terms and Definitions for Co-simulation . . . . .	584
29.10.2 Overview of the Single/Multiple Domain Co-simulation . . . . .	585
29.10.3 Configuring the Co-simulation . . . . .	586
29.10.4 Performing a Co-simulation and Retrieving Results . . . . .	588
29.10.5 The “ <i>Initial conditions for co-simulation</i> ” ( <i>ComCosim</i> ) Command . . . . .	589
29.11 Co-simulation with External Solver . . . . .	590
29.11.1 Overview of the External Solver Co-simulation . . . . .	591
29.11.2 Prepare Case for Co-simulation with External Solver . . . . .	594
29.11.3 Performing a Co-simulation and Retrieving Results . . . . .	596
29.11.4 The “ <i>Prepare co-simulation with ext. solver</i> ” ( <i>ComCosimsetup</i> ) Command . . . . .	597
29.11.5 The “ <i>Initial conditions for co-simulation</i> ” ( <i>ComCosim</i> ) Command . . . . .	597
29.12 Frequency Response Analysis . . . . .	598
29.12.1 Basic Usage . . . . .	598

29.12.2 Basic Data Page . . . . .	599
29.12.3 Output Page . . . . .	602
29.12.4 Advanced Page . . . . .	603
29.12.5 Output Plots . . . . .	603
29.12.6 Output Results Files . . . . .	604
29.13 Frequency Analysis . . . . .	604
29.13.1 Prony Analysis Overview . . . . .	604
29.13.2 Basic Usage . . . . .	606
29.13.3 Basic Options Page . . . . .	607
29.13.4 FFT Page . . . . .	608
29.13.5 Prony Analysis Page . . . . .	610
29.13.6 Recalculation . . . . .	611
29.13.7 Output Plots . . . . .	611
29.13.8 Output Results Files . . . . .	612
29.13.9 General Recommendations on the Use of <i>Prony Analysis</i> . . . . .	613
29.13.10 Quick Overview of Used Formulas . . . . .	616
<b>30 Models for Dynamic Simulations</b> . . . . .	<b>618</b>
30.1 System Modelling Approach . . . . .	619
30.1.1 The Composite Model . . . . .	624
30.1.2 The Composite Frame . . . . .	626
30.1.3 The Common Model . . . . .	628
30.2 The Composite Block Definition . . . . .	633
30.2.1 Drawing Composite Block Diagrams and Composite Frames . . . . .	635
30.3 User Defined (DSL) Models . . . . .	640
30.3.1 Modelling and Simulation Tools . . . . .	642
30.3.2 DSL Implementation: an Introduction . . . . .	642
30.3.3 Defining DSL Models . . . . .	646
30.4 The <i>DIGSILENT</i> Simulation Language (DSL) . . . . .	651
30.4.1 Terms and Abbreviations . . . . .	652
30.4.2 General DSL Syntax . . . . .	652
30.4.3 DSL Variables . . . . .	653
30.4.4 DSL Structure . . . . .	653

---

30.4.5	Definition Code . . . . .	654
30.4.6	Initial Conditions . . . . .	655
30.4.7	Equation Code . . . . .	659
30.4.8	Equation Statement . . . . .	659
30.4.9	DSL Macros . . . . .	660
30.4.10	Events and Messages . . . . .	661
30.4.11	DSL Modelling Examples . . . . .	662
30.4.12	Example of a Complete DSL Model . . . . .	662
30.5	DSL Reference . . . . .	664
30.5.1	DSL Standard Functions . . . . .	664
30.5.2	DSL Special Functions . . . . .	665
30.5.3	DSL Global Library Macros . . . . .	682
30.6	Interfaces for Dynamic Models . . . . .	682
30.6.1	C Interface . . . . .	682
30.6.2	External C Interface acc. to IEC 61400-27-1 . . . . .	685
30.6.3	MATLAB Interface . . . . .	686
<b>31</b>	<b>System Parameter Identification</b> . . . . .	<b>695</b>
31.1	Introduction . . . . .	695
31.2	Performing a Parameter Identification . . . . .	696
31.2.1	Basic Options . . . . .	696
31.2.2	Controls / Compared Signals . . . . .	697
31.2.3	PSO (Particle Swarm Optimisation) . . . . .	699
31.2.4	Nelder Mead . . . . .	699
31.2.5	DIRECT (DIviding RECTangle) . . . . .	699
31.2.6	Random Number Generation . . . . .	699
31.2.7	Gradient Calculation . . . . .	699
31.2.8	Stopping Criteria . . . . .	700
31.2.9	Output . . . . .	700
31.3	Algorithms . . . . .	701
31.3.1	Problem Description . . . . .	701
31.3.2	Particle Swarm Optimisation (PSO) . . . . .	701
31.3.3	Nelder Mead . . . . .	703

---

31.3.4 DIRECT . . . . .	705
31.3.5 BFGS . . . . .	706
31.3.6 Legacy (Quasi-Newton) . . . . .	706
31.4 What solver should I pick? (Pros and Cons of the different solvers) . . . . .	706
31.4.1 PSO - Particle Swarm Optimisation . . . . .	706
31.4.2 Nelder-Mead . . . . .	706
31.4.3 DIRECT . . . . .	707
31.4.4 BFGS . . . . .	707
31.4.5 Legacy (Quasi-Newton) . . . . .	707
31.5 What can I do if a solver has difficulties in finding good parameters? . . . . .	707
31.5.1 PSO - Particle Swarm Optimisation . . . . .	707
31.5.2 Nelder Mead . . . . .	707
31.5.3 DIRECT . . . . .	708
31.5.4 BFGS . . . . .	708
31.5.5 Legacy . . . . .	708
<b>32 Modal Analysis / Eigenvalue Calculation</b>	<b>709</b>
32.1 Theory of Modal Analysis . . . . .	709
32.2 How to Execute a Modal Analysis . . . . .	712
32.2.1 Modal Analysis Command - Basic Options . . . . .	712
32.2.2 Modal Analysis Command - Advanced Options . . . . .	714
32.2.3 Modal Analysis Command - Output . . . . .	715
32.3 Viewing Modal Analysis Results . . . . .	718
32.3.1 Modal/Eigenvalue Analysis Results Command . . . . .	718
32.3.2 Modal Analysis Results in Built-in Plots . . . . .	720
32.3.3 Eigenvalues Results in Single Line Diagrams . . . . .	725
32.3.4 Modal Analysis Results in the Output Window . . . . .	725
32.3.5 Modal Analysis Results in the Data Browser . . . . .	726
<b>33 Protection</b>	<b>728</b>
33.1 Introduction . . . . .	728
33.1.1 The modelling structure . . . . .	729
33.1.2 The relay frame . . . . .	730

---

33.1.3 The relay type . . . . .	730
33.1.4 The relay element . . . . .	731
33.2 How to define a protection scheme in <i>PowerFactory</i> . . . . .	732
33.2.1 Overview . . . . .	732
33.2.2 Adding protective devices to the network model . . . . .	733
33.2.3 Graphical representations of protection devices in single line diagrams . . . . .	736
33.2.4 Locating protection devices within the network model . . . . .	739
33.3 Basics of an overcurrent protection scheme . . . . .	739
33.3.1 Overcurrent relay model setup - basic data page . . . . .	739
33.3.2 Overcurrent relay model setup - max/min fault currents tab . . . . .	741
33.3.3 Configuring the current transformer . . . . .	741
33.3.4 Configuring the voltage transformer . . . . .	744
33.3.5 Configuring a combined Instrument transformer . . . . .	746
33.3.6 How to add a fuse to the network model . . . . .	746
33.3.7 Basic relay blocks for overcurrent relays . . . . .	748
33.4 The time-overcurrent plot . . . . .	753
33.4.1 How to create a time-overcurrent plot . . . . .	754
33.4.2 Understanding the time-overcurrent plot . . . . .	755
33.4.3 Showing the calculation results on the time-overcurrent plot . . . . .	755
33.4.4 Displaying the grading margins . . . . .	755
33.4.5 Adding a user defined permanent current line to the time-overcurrent plot . . . . .	756
33.4.6 Configuring the auto generated protection diagram . . . . .	757
33.4.7 Overcurrent plot options . . . . .	757
33.4.8 Altering protection device characteristic settings from the time-overcurrent plot . . . . .	759
33.4.9 How to split the relay/fuse characteristic . . . . .	759
33.4.10 Equipment damage curves . . . . .	762
33.5 Basics of a distance protection scheme . . . . .	772
33.5.1 Distance relay model setup - basic data page . . . . .	773
33.5.2 Primary or secondary Ohm selection for distance relay parameters . . . . .	773
33.5.3 Basic relay blocks used for distance protection . . . . .	773
33.6 The impedance plot (R-X diagram) . . . . .	780

---

33.6.1 How to create an R-X diagram . . . . .	780
33.6.2 Understanding the R-X diagram . . . . .	781
33.6.3 Configuring the R-X plot . . . . .	781
33.6.4 Modifying the relay settings and branch elements from the R-X plot . . . . .	784
33.7 The relay operational limits plot (P-Q diagram) . . . . .	785
33.7.1 How to create a P-Q diagram . . . . .	785
33.7.2 Understanding the P-Q diagram . . . . .	785
33.7.3 Configuring the P-Q plot . . . . .	786
33.7.4 Modifying the starting element settings from the R-X plot . . . . .	788
33.8 The time-distance plot . . . . .	788
33.8.1 Forward and reverse plots . . . . .	789
33.8.2 The path axis . . . . .	789
33.8.3 Methods of calculating tripping times . . . . .	790
33.8.4 Short-circuit calculation settings . . . . .	791
33.8.5 The distance axis units . . . . .	791
33.8.6 The reference relay . . . . .	791
33.8.7 Capture relays . . . . .	792
33.8.8 Double-click positions . . . . .	792
33.8.9 The context sensitive menu . . . . .	792
33.9 Basics of a differential protection scheme . . . . .	792
33.9.1 Differential relay model setup-basic data page . . . . .	793
33.9.2 Basic relay blocks used for differential protection . . . . .	793
33.10 Differential Plots . . . . .	793
33.10.1 Magnitude biased differential diagram . . . . .	794
33.10.2 Phase comparison differential diagram . . . . .	795
33.11 The Short-Circuit Sweep command . . . . .	795
33.12 Short-Circuit Sweep Plots . . . . .	797
33.12.1 Configuration of Short-Circuit Sweep plots . . . . .	798
33.13 Protection coordination assistant . . . . .	800
33.13.1 Protection coordination assistant - technical background . . . . .	800
33.13.2 Protection coordination assistant - General Handling . . . . .	802
33.13.3 Protection Coordination Assistant Command - Basic Options . . . . .	802

33.13.4 Protection Coordination Assistant Command - Distance Protection . . . . .	803
33.13.5 Protection Coordination Assistant Command - Advanced Options . . . . .	815
33.13.6 Prerequisites for using the distance protection coordination tool . . . . .	816
33.13.7 How to run the distance protection coordination calculation . . . . .	816
33.13.8 How to output results from the protection coordination assistant . . . . .	816
33.14 Accessing results . . . . .	819
33.14.1 Quick access to protection plots . . . . .	819
33.14.2 Tabular protection setting report . . . . .	821
33.14.3 Results in single line graphic . . . . .	823
33.15 Protection Audit . . . . .	824
33.15.1 Protection Audit Command Handling . . . . .	824
33.15.2 Protection Audit Results Command Handling . . . . .	826
33.15.3 Report Handling and interpretation . . . . .	829
33.16 Short circuit trace . . . . .	832
33.16.1 Short Circuit Trace Handling . . . . .	834
33.17 Protection Graphic Assistant . . . . .	837
33.17.1 Reach Colouring . . . . .	837
33.17.2 Short-Circuit Sweep Plot . . . . .	839
33.17.3 Diagram Update . . . . .	841
33.18 Building a basic overcurrent relay model . . . . .	842
33.19 Appendix - other commonly used relay blocks . . . . .	845
33.19.1 The frequency measurement block . . . . .	846
33.19.2 The frequency block . . . . .	846
33.19.3 The under-/overvoltage block . . . . .	846
33.20 Relay block technical references . . . . .	846
<b>34 Arc-Flash Hazard Analysis</b> . . . . .	<b>847</b>
34.1 Introduction . . . . .	847
34.2 Arc-Flash Hazard Analysis Background . . . . .	847
34.2.1 General . . . . .	847
34.2.2 Data Inputs . . . . .	848
34.3 Arc-Flash Hazard Analysis Calculation Options . . . . .	850
34.3.1 Arc-Flash Hazard Analysis Basic Options Page . . . . .	850

34.3.2 Arc-Flash Hazard Analysis Advanced Options Page . . . . .	851
34.4 Arc-Flash Hazard Analysis Results . . . . .	851
34.4.1 Viewing Results in the Single Line Graphic . . . . .	851
34.4.2 Arc-Flash Reports Dialog . . . . .	852
34.4.3 Arc-Flash Labels . . . . .	852
34.5 Example Arc-Flash Hazard Analysis Calculation . . . . .	853
<b>35 Cable Analysis</b>	<b>856</b>
35.1 Cable Sizing . . . . .	857
35.2 Calculation Options . . . . .	857
35.2.1 Basic Options Page . . . . .	857
35.2.2 Constraints Page . . . . .	858
35.2.3 Output Page . . . . .	859
35.2.4 Advanced Options Page . . . . .	861
35.3 Cable Ampacity . . . . .	862
35.3.1 Cable Ampacity calculation options . . . . .	863
35.4 Reporting command (ComCableReport) . . . . .	863
35.5 Model Parameters . . . . .	864
35.5.1 Line Type Parameters . . . . .	864
35.5.2 Line Element Parameters . . . . .	865
35.5.3 Single Core and multicore/pipe cables . . . . .	865
35.5.4 Cable Layout object . . . . .	865
<b>36 Power Quality and Harmonics Analysis</b>	<b>869</b>
36.1 Introduction . . . . .	869
36.2 Harmonic Load Flow . . . . .	870
36.2.1 Basic Options . . . . .	870
36.2.2 IEC 61000-3-6 . . . . .	872
36.2.3 Advanced Options . . . . .	872
36.2.4 Harmonics Indices . . . . .	873
36.3 Frequency Sweep . . . . .	874
36.3.1 Basic Options . . . . .	875
36.3.2 Advanced Options . . . . .	876

---

36.4 Filter Analysis . . . . .	876
36.5 Modelling Harmonic Sources . . . . .	877
36.5.1 Definition of Harmonic Injections . . . . .	878
36.5.2 Assignment of Harmonic Injections . . . . .	883
36.5.3 Frequency Dependent Parameters . . . . .	883
36.5.4 Waveform Plot . . . . .	885
36.6 Flicker Analysis (IEC 61400-21) . . . . .	887
36.6.1 Continuous Operation . . . . .	887
36.6.2 Switching Operations . . . . .	888
36.6.3 Flicker Contribution of Wind Turbine Generator Models . . . . .	889
36.6.4 Definition of Flicker Coefficients . . . . .	889
36.6.5 Assignment of Flicker Coefficients . . . . .	890
36.6.6 Flicker Results Variables . . . . .	890
36.7 Short-Circuit Power . . . . .	890
36.7.1 Balanced Harmonic Load Flow . . . . .	890
36.7.2 Unbalanced Harmonic Load Flow . . . . .	891
36.7.3 Sk Result Variables . . . . .	891
36.7.4 Short-Circuit Power of the External Grid . . . . .	892
36.8 Connection Request Assessment . . . . .	892
36.8.1 Connection Request Assessment: D-A-CH-CZ . . . . .	892
36.8.2 Connection Request Assessment: BDEW, 4th Supplement . . . . .	894
36.8.3 Connection Request Assessment: VDE-AR-N 4105 . . . . .	896
36.8.4 Connection Request Assessment: VDE-AR-N 4110 . . . . .	898
36.8.5 Connection Request Assessment Report . . . . .	900
36.9 Definition of Result Variables . . . . .	901
36.9.1 Definition of Variable Sets . . . . .	902
36.9.2 Selection of Result Variables within a Variable Set . . . . .	903
<b>37 Flickermeter</b>	<b>904</b>
37.1 Introduction . . . . .	904
37.2 Flickermeter (IEC 61000-4-15) . . . . .	904
37.2.1 Calculation of Short-Term Flicker . . . . .	904
37.2.2 Calculation of Long-Term Flicker . . . . .	905

---

37.3 Flickermeter Calculation . . . . .	905
37.3.1 Flickermeter Command . . . . .	905
37.3.2 Data Source . . . . .	905
37.3.3 Signal Settings . . . . .	906
37.3.4 Advanced Options . . . . .	907
37.3.5 Input File Types . . . . .	908
<b>38 Optimal Power Flow</b>	<b>911</b>
38.1 Introduction . . . . .	911
38.2 AC Optimisation (Interior Point Method) . . . . .	911
38.2.1 AC Optimisation - Basic Options . . . . .	911
38.2.2 AC Optimisation - Initialisation . . . . .	918
38.2.3 AC Optimisation - Advanced Options . . . . .	919
38.2.4 AC Optimisation - Iteration Control . . . . .	919
38.2.5 AC Optimisation - Output . . . . .	921
38.3 DC Optimisation (Linear Programming) . . . . .	922
38.3.1 DC Optimisation - Basic Options . . . . .	923
38.3.2 DC Optimisation - Initialisation . . . . .	925
38.3.3 DC Optimisation - Advanced Options . . . . .	926
38.3.4 DC Optimisation - Iteration Control . . . . .	927
38.4 Contingency Constrained DC Optimisation (LP Method) . . . . .	928
38.4.1 Contingency Constrained DC Optimisation - Basic Options . . . . .	928
38.4.2 Contingency Constrained DC Optimisation - Initialisation . . . . .	930
38.4.3 Contingency Constrained DC Optimisation - Advanced Options . . . . .	930
38.4.4 Contingency Constrained DC Optimisation - Iteration Control . . . . .	931
38.4.5 Contingency Constrained DC Optimisation - Output . . . . .	931
38.5 Troubleshooting Optimal Power Flow Problems . . . . .	932
38.5.1 Verification of Load Flow Options and Results . . . . .	932
38.5.2 Verifications of OPF Constraints . . . . .	932
38.5.3 Verification of the OPF Controls . . . . .	933
38.5.4 Step-by-Step Approach . . . . .	933
<b>39 Unit Commitment and Dispatch Optimisation</b>	<b>935</b>

---

39.1	Introduction . . . . .	935
39.2	Application Cases for the Unit Commitment . . . . .	935
39.2.1	Full Unit Commitment . . . . .	935
39.2.2	Market Simulation . . . . .	936
39.2.3	Redispatch Calculation . . . . .	936
39.3	Unit Commitment Command . . . . .	936
39.3.1	Basic Options . . . . .	936
39.3.2	Objective Function . . . . .	937
39.3.3	Controls/Constraints . . . . .	938
39.3.4	Results/Output . . . . .	940
39.3.5	Maintenance . . . . .	940
39.3.6	Algorithm . . . . .	941
39.4	Handling of results . . . . .	942
39.4.1	Reports . . . . .	942
39.4.2	Plots . . . . .	943
39.4.3	Colouring Mode . . . . .	943
39.4.4	Load Unit Commitment Results . . . . .	943
39.4.5	Flexible Data Page . . . . .	944
39.5	Generating Units . . . . .	944
39.5.1	Controls and Limits . . . . .	944
39.5.2	Operating Costs . . . . .	945
39.5.3	Redispatch Costs . . . . .	946
39.5.4	Start-Up/Shut-down Costs . . . . .	946
39.5.5	Constraints . . . . .	946
39.6	Storage Units . . . . .	947
39.6.1	Efficiency curves . . . . .	947
39.7	Virtual Power Plants . . . . .	947
39.8	Other Network Elements . . . . .	948
39.8.1	Transformers, voltage regulators and shunts . . . . .	948
39.8.2	Loads . . . . .	948
39.8.3	Boundaries . . . . .	948
39.8.4	Terminals . . . . .	949

---

39.8.5 Lines . . . . .	949
39.8.6 Regions: Grids, Zones and Areas . . . . .	949
39.8.7 HVDC Converters . . . . .	949
39.9 Troubleshooting . . . . .	949
39.9.1 Solver Selection . . . . .	949
39.9.2 Soft Constraints . . . . .	949
39.9.3 Voltage Controlling Elements . . . . .	949
39.9.4 Reference Machine . . . . .	949
39.9.5 Not Regarded Constraints . . . . .	950
39.9.6 Rolling horizon and time dependencies . . . . .	950
<b>40 Transmission Network Tools</b>	<b>951</b>
40.1 Introduction . . . . .	951
40.2 PV Curves . . . . .	952
40.2.1 PV Curves Calculation . . . . .	952
40.2.2 PV Curves Plot . . . . .	954
40.2.3 Outputs and Results . . . . .	954
40.3 QV Curves . . . . .	955
40.3.1 QV Curves Calculation . . . . .	956
40.3.2 QV Curves Plot . . . . .	958
40.4 Power Transfer Distribution Factors (PTDF) . . . . .	958
40.4.1 Calculation Options . . . . .	959
40.5 Transfer Capacity Analysis . . . . .	960
40.5.1 Basic Data . . . . .	961
40.5.2 Constraints . . . . .	962
40.5.3 Output . . . . .	962
40.5.4 Iteration Control . . . . .	963
40.5.5 Advanced . . . . .	963
<b>41 Distribution Network Tools</b>	<b>965</b>
41.1 Introduction . . . . .	965
41.2 Voltage Sag . . . . .	965
41.2.1 Calculation Options . . . . .	966

41.2.2 How to Perform a Voltage Sag Table Assessment . . . . .	966
41.2.3 Voltage Sag Table Assessment Results . . . . .	967
41.3 Voltage Profile Optimisation . . . . .	968
41.3.1 Optimisation Procedure . . . . .	969
41.3.2 Basic Options Page . . . . .	972
41.3.3 Output Page . . . . .	972
41.3.4 Advanced Options Page . . . . .	973
41.3.5 Results of Voltage Profile Optimisation . . . . .	973
41.4 Tie Open Point Optimisation . . . . .	974
41.4.1 Tie Open Point Optimisation Background . . . . .	974
41.4.2 How to run a Tie Open Point Optimisation . . . . .	975
41.5 Backbone Calculation . . . . .	978
41.5.1 Basic Options Page . . . . .	979
41.5.2 Scoring Settings Page . . . . .	980
41.5.3 Tracing Backbones . . . . .	981
41.5.4 Example Backbone Calculation . . . . .	981
41.6 Optimal Capacitor Placement . . . . .	982
41.6.1 OCP Objective Function . . . . .	983
41.6.2 OCP Optimisation Procedure . . . . .	984
41.6.3 Basic Options Page . . . . .	985
41.6.4 Available Capacitors Page . . . . .	986
41.6.5 Load Characteristics Page . . . . .	987
41.6.6 Advanced Options Page . . . . .	987
41.6.7 Results . . . . .	988
41.7 Phase Balance Optimisation . . . . .	989
41.7.1 Objective functions . . . . .	989
41.7.2 Methods . . . . .	990
41.7.3 Considered elements . . . . .	991
41.7.4 Representation of solution . . . . .	991
41.7.5 Output . . . . .	991
41.8 Optimisation Algorithms . . . . .	991
41.8.1 Genetic Algorithm . . . . .	992

41.8.2 Simulated Annealing . . . . .	993
41.9 Hosting Capacity . . . . .	993
41.9.1 Technical Background . . . . .	994
41.9.2 Hosting Capacity Analysis Configuration . . . . .	995
41.9.3 Results of the Hosting Capacity . . . . .	999
<b>42 Outage Planning</b>	<b>1002</b>
42.1 Introduction . . . . .	1002
42.2 Creating Planned Outages . . . . .	1002
42.2.1 Creating Planned Outages from Graphic or Network Model Manager . . . . .	1002
42.2.2 Creating Planned Outages in Data Manager . . . . .	1003
42.2.3 Recurrent Outages . . . . .	1003
42.2.4 Adding Additional Events to an Outage . . . . .	1003
42.3 Handling Planned Outages using the Outage Planning toolbar . . . . .	1003
42.3.1 Show Planned Outages . . . . .	1003
42.3.2 Apply Planned Outages . . . . .	1004
42.3.3 Reset All Planned Outages . . . . .	1004
42.3.4 Start Recording . . . . .	1004
42.3.5 Outage Schedule Report . . . . .	1004
<b>43 Probabilistic Analysis</b>	<b>1005</b>
43.1 Introduction . . . . .	1005
43.2 Technical Background . . . . .	1006
43.2.1 Distributions . . . . .	1006
43.2.2 Modelling Dependencies . . . . .	1009
43.2.3 Probabilistic Analysis Methods . . . . .	1010
43.2.4 Statistics . . . . .	1012
43.2.5 Distribution Estimation . . . . .	1014
43.2.6 Distribution Fitting . . . . .	1015
43.3 Object Settings . . . . .	1016
43.3.1 Distributions . . . . .	1016
43.3.2 Dependencies . . . . .	1018
43.3.3 Distribution Estimation Command . . . . .	1019

---

43.3.4 Probabilistic Analysis Command . . . . .	1021
43.3.5 Continue Probabilistic Analysis . . . . .	1022
43.3.6 Probabilistic Analysis Player . . . . .	1022
43.3.7 Results File Handling . . . . .	1022
43.3.8 Representation of results . . . . .	1023
<b>44 Reliability Analysis</b>	<b>1027</b>
44.1 Introduction . . . . .	1027
44.2 Probabilistic Reliability Assessment Technical Background . . . . .	1029
44.2.1 Reliability Assessment Procedure . . . . .	1030
44.2.2 Stochastic Models . . . . .	1031
44.2.3 Calculated Results for Reliability Assessment . . . . .	1032
44.2.4 System State Enumeration in Reliability Assessment . . . . .	1037
44.3 Setting up the Network Model for Reliability Assessment . . . . .	1038
44.3.1 How to Define Stochastic Failure and Repair models . . . . .	1039
44.3.2 How to Create Feeders for Reliability Calculation . . . . .	1044
44.3.3 Configuring Switches for the Reliability Calculation . . . . .	1045
44.3.4 Load Modelling for Reliability Assessment . . . . .	1047
44.3.5 Modelling Load Interruption Costs . . . . .	1048
44.3.6 System Demand and Load States . . . . .	1048
44.3.7 Fault Clearance Based on Protection Device Location . . . . .	1049
44.3.8 How to Consider Planned Maintenance . . . . .	1049
44.3.9 Specifying Individual Component Constraints . . . . .	1049
44.3.10 Consider switching rules . . . . .	1050
44.4 Running The Reliability Assessment Calculation . . . . .	1050
44.4.1 How to run the Reliability Assessment . . . . .	1050
44.5 Results of the Reliability Analysis . . . . .	1058
44.5.1 Contribution to Reliability Indices . . . . .	1058
44.5.2 Viewing Results in the Single Line Diagram . . . . .	1058
44.5.3 Viewing Results in the Data Browser . . . . .	1061
44.5.4 Reliability Reports . . . . .	1061
<b>45 Optimal Power Restoration</b>	<b>1063</b>

---

45.1 Failure Effect Analysis . . . . .	1063
45.2 Animated Tracing of Individual Cases . . . . .	1068
45.3 Optimal RCS Placement . . . . .	1068
45.3.1 Basic Options Page . . . . .	1069
45.3.2 Output Page . . . . .	1069
45.3.3 Advanced Options Page . . . . .	1070
45.3.4 Example Optimal RCS Calculation . . . . .	1070
45.4 Optimal Manual Restoration . . . . .	1071
45.4.1 OMR Calculation Prerequisites . . . . .	1072
45.4.2 Basic Options Page . . . . .	1072
45.4.3 Advanced Options Page . . . . .	1073
45.4.4 Definition of the objective function . . . . .	1074
45.4.5 Example of an Optimal Manual Restoration Calculation . . . . .	1075
<b>46 Generation Adequacy Analysis</b>	<b>1077</b>
46.1 Introduction . . . . .	1077
46.2 Technical Background . . . . .	1077
46.3 Database Objects and Models . . . . .	1080
46.3.1 Stochastic Model for Generation . . . . .	1080
46.3.2 Power Curve Type . . . . .	1081
46.3.3 Meteorological station . . . . .	1081
46.4 Assignment of Stochastic Model for Generation . . . . .	1082
46.4.1 Definition of a Stochastic Multi-State Model . . . . .	1082
46.4.2 Stochastic Wind Model . . . . .	1083
46.4.3 Time Series Characteristic for Wind Generation . . . . .	1083
46.4.4 Demand definition . . . . .	1085
46.5 Generation Adequacy Analysis toolbar . . . . .	1085
46.5.1 Generation Adequacy Initialisation command . . . . .	1085
46.5.2 Run Generation Adequacy command . . . . .	1086
46.6 Generation Adequacy results . . . . .	1088
46.6.1 Distribution (Cumulative Probability) Plots . . . . .	1088
46.6.2 Monte-Carlo Draws (Iterations) Plots . . . . .	1090
46.6.3 Convergence Plots . . . . .	1091

---

46.6.4 Summary of variables calculated during the Generation Adequacy Analysis . . . . .	1093
<b>47 Sensitivities / Distribution Factors</b>	<b>1094</b>
47.1 Overview of Sensitivity / Distribution Factors Calculations . . . . .	1094
47.1.1 Terminology . . . . .	1094
47.1.2 Result Quantities . . . . .	1095
47.2 Sensitivities / Distribution Factors Options . . . . .	1095
47.2.1 Basic Options . . . . .	1095
47.2.2 Results . . . . .	1096
47.2.3 Advanced Options . . . . .	1097
47.2.4 Output . . . . .	1098
47.2.5 Modal/Eigenvalue Analysis . . . . .	1098
47.3 Reporting . . . . .	1099
47.3.1 General . . . . .	1099
47.3.2 Thresholds . . . . .	1099
47.3.3 Used Format . . . . .	1099
47.3.4 Report output . . . . .	1099
47.4 Troubleshooting . . . . .	1100
<b>48 Network Reduction</b>	<b>1101</b>
48.1 Introduction . . . . .	1101
48.2 Technical Background . . . . .	1101
48.2.1 Network Reduction for Load Flow . . . . .	1102
48.2.2 Network Reduction for Short-Circuit . . . . .	1102
48.2.3 Network Reduction using REI Method . . . . .	1102
48.2.4 Network Reduction for Dynamic Equivalent . . . . .	1102
48.3 How to Carry Out a Network Reduction . . . . .	1103
48.3.1 How to Backup the Project (optional) . . . . .	1103
48.3.2 How to run the Network Reduction tool . . . . .	1103
48.3.3 Expected Output of the Network Reduction . . . . .	1104
48.4 Network Reduction Command . . . . .	1105
48.4.1 Basic Options . . . . .	1105
48.4.2 Ward Equivalent . . . . .	1106

48.4.3 REI Equivalent . . . . .	1107
48.4.4 Dynamic Equivalent . . . . .	1108
48.4.5 Outputs . . . . .	1110
48.4.6 Advanced Options . . . . .	1110
48.4.7 Verification . . . . .	1111
48.5 Network Reduction Example . . . . .	1112
48.6 Tips for using the Network Reduction Tool . . . . .	1113
48.6.1 Network Reduction doesn't Reduce Isolated Areas . . . . .	1113
48.6.2 The Reference Machine is not Reduced . . . . .	1114
<b>49 Techno-Economical Calculation</b>	<b>1115</b>
49.1 Introduction . . . . .	1115
49.2 Requirements for Calculation . . . . .	1116
49.3 Calculation Options . . . . .	1116
49.3.1 Basic Options Page . . . . .	1116
49.3.2 Costs Page . . . . .	1117
49.3.3 Output Page . . . . .	1118
49.3.4 Parallel Computing Page . . . . .	1118
49.3.5 Results Reports . . . . .	1118
49.4 Example Calculation . . . . .	1118
<b>50 State Estimation</b>	<b>1122</b>
50.1 Introduction . . . . .	1122
50.2 Objective Function . . . . .	1123
50.3 Components of the <i>PowerFactory</i> State Estimation . . . . .	1123
50.3.1 Plausibility Check . . . . .	1125
50.3.2 Observability Analysis . . . . .	1125
50.3.3 State Estimation (Non-Linear Optimisation) . . . . .	1126
50.4 State Estimation Data Input . . . . .	1126
50.4.1 Measurements . . . . .	1127
50.4.2 Editing the Element Data . . . . .	1131
50.5 Running SE . . . . .	1132
50.5.1 Basic Setup Options . . . . .	1132

---

50.5.2 Advanced Setup Options for the Plausibility Check . . . . .	1134
50.5.3 Advanced Setup Options for the Observability Check . . . . .	1135
50.5.4 Advanced Setup Options for Bad Data Detection . . . . .	1135
50.5.5 Advanced Setup Options for Iteration Control . . . . .	1135
50.6 Results . . . . .	1137
50.6.1 Output Window Report . . . . .	1137
50.6.2 External Measurements . . . . .	1137
50.6.3 Estimated States . . . . .	1139
50.6.4 Colour Representation . . . . .	1140
<b>51 Motor Starting</b>	<b>1142</b>
51.1 Introduction . . . . .	1142
51.2 How to define a motor . . . . .	1142
51.2.1 How to define a motor Type and starting methodology . . . . .	1142
51.2.2 How to define a motor driven machine . . . . .	1144
51.3 How to run a Motor Starting simulation . . . . .	1144
51.3.1 Basic Options Page . . . . .	1144
51.3.2 Output Page . . . . .	1146
51.3.3 Motor Starting simulation results . . . . .	1148
51.3.4 Motor Starting Example . . . . .	1149
<b>V Appendix</b>	<b>1152</b>
<b>A Hotkeys Reference</b>	<b>1153</b>
A.1 Calculation Hotkeys . . . . .	1153
A.2 Graphic Windows Hotkeys . . . . .	1153
A.3 Data Manager Hotkeys . . . . .	1155
A.4 Dialog Hotkeys . . . . .	1157
A.5 Output Window Hotkeys . . . . .	1157
A.6 Editor Hotkeys . . . . .	1158
<b>B Standard Models in <i>PowerFactory</i></b>	<b>1161</b>
B.1 AVR Models . . . . .	1161
B.2 Turbine-Governor Models . . . . .	1169

B.3	PSS Models . . . . .	1174
B.4	Excitation Limiter Models . . . . .	1176
B.5	Static Compensator Models . . . . .	1177
B.6	Frames for Dynamic Models . . . . .	1178
B.7	Typical Arrangements . . . . .	1179
<b>C</b>	<b>ENTSO-E Dynamic Models in <i>PowerFactory</i></b>	<b>1180</b>
C.1	Excitation Models . . . . .	1180
C.2	Governor Models . . . . .	1183
C.3	Power System Stabiliser Models . . . . .	1184
C.4	Voltage Compensator Models . . . . .	1185
C.5	Over-Excitation Limiter Models . . . . .	1185
C.6	Under-Excitation Limiter Models . . . . .	1186
C.7	Frames for ENTSO-E Dynamic Models . . . . .	1186
<b>D</b>	<b>The <i>DIGSILENT</i> Output Language</b>	<b>1187</b>
D.1	Format string, Variable names and text Lines . . . . .	1188
D.2	Placeholders . . . . .	1188
D.3	Variables, Units and Names . . . . .	1189
D.4	Colour . . . . .	1191
D.5	Advanced Syntax Elements . . . . .	1191
D.6	Line Types and Page Breaks . . . . .	1192
D.7	Predefined Text Macros . . . . .	1192
D.8	Object Iterations, Loops, Filters and Includes . . . . .	1193
<b>E</b>	<b>Element Symbol Definition</b>	<b>1194</b>
E.1	Introduction . . . . .	1194
E.2	General Symbol Definition . . . . .	1194
E.3	Geometrical Description . . . . .	1195
E.4	Including Graphic Files as Symbols . . . . .	1197
<b>F</b>	<b>Standard Functions DPL and DSL</b>	<b>1198</b>
<b>Bibliography</b>		<b>1200</b>

<b>Glossary</b>	<b>1202</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 \*.pdf 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)

**Portal log-in and Registration:** <https://www.digsilent.de/en/user-registration.html>

New Incident

Incident Type: Question/Support request

Title: abc Line loadings

Product: PowerFactory

Version: 2018

Service Pack: SP4

External ID:

Description: abc

Dear PowerFactory support,  
In my model it is not clear to me why the lines Line-A and Line-B are not loaded equally. They have the same type data and length. Could you assist? I attach my project.  
Regards,  
A. User

Copyright © 1996-2018, DlgSILENT GmbH, All Rights Reserved.

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 → 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 *DlgSILENT* or your *PowerFactory* licence, please contact us via:

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

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

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

## Chapter 3

# Documentation and Help System

DIGSILENT 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:

- **Getting Started:** a document describing the first steps to follow after receiving the installation DVD or downloading the software from the *DIGSILENT* 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 5.5: Offline Mode User Guide.
- **Tutorial:** basic information for new users and hands-on tutorial. 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 *DIGSILENT* download area.
- **Technical References:** description of the models implemented in *PowerFactory* for the different power systems components. The technical reference documents are attached to the online help ([Technical References Document](#)).
- **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
  - DSL models C-Interface
  - DSL functions C-Interface
  - Dynamic Models C Interface
- **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 from the *DlgSILENT* download area.
- **Knowledge base:** see Chapter 2: Contact
- **Technical Support:** see Chapter 2: Contact
- **Portal log-in and Registration:** <https://www.digsilent.de/index.php/support.html>
- **Website:** [www.digsilent.de](http://www.digsilent.de)

# Chapter 4

## ***PowerFactory* Overview**

The calculation program DLgSILENT *PowerFactory*, 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.

“*DlgSILENT*” is an acronym for “**D**igital **S**imu**L**ation of **E**lectrical **N**e**T**works”. *DlgSILENT* 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

DIGSILENT *PowerFactory* software is implemented as a single executable program, and is fully compatible with Windows 7, Windows 8 and Windows 10. 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

DIGSILENT *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 DIGSILENT *PowerFactory* can accommodate everything within one integrated program and one integrated database.

### Database Integration

**One Database Concept:** DIGSILENT *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-sensitive menu commands.

Chapter 6: User Accounts, User Groups, and Profiles (Section 6.7 Creating 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 [47](#))
- Low Voltage Network Analysis (Section [25.4.2](#): Advanced Load Options)
- Contingency Analysis (Chapter [27](#))
- Quasi-Dynamic simulation, which allows to perform several load flow calculations in a period of time (Chapter [28](#))
- Network Reduction (Chapter [48](#))
- 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 (Section [36.8](#))
- Transmission Network Tools (Chapter [40](#)), including:
  - PV curves calculation
  - QV curves calculation
  - Power Transfer Distribution Factors
  - Transfer Capacity Analysis
- Distribution Network Tools (Chapter [41](#)), including:
  - Tie Open Point Optimisation
  - Voltage Profile Optimisation
  - Phase Balance Optimisation
  - Optimal Capacitor Placement
  - Hosting Capacity Analysis
- Outage Planning: tool for management of planned outages (Chapter [42](#))
- Probabilistic Analysis (Chapter [43](#))
- Reliability Analysis Functions:
  - Reliability Assessment (Chapter [44](#))
  - Optimal Power Restoration (Chapter [45](#))
  - Optimal Remote Control Switch (RCS) Placement (Section [45.3](#))
  - Optimal Manual Restoration (Section [45.4](#))
  - Generation Adequacy Analysis (Chapter [46](#))
- Optimal Power Flow (Chapter [38](#))
- Unit Commitment (Chapter [39](#))
- Techno-Economical Calculation (Chapter [49](#))
- State Estimation (Chapter [50](#))
- RMS Simulation: time-domain simulation for electromechanical transients (Chapter [29](#))
- EMT Simulation: time-domain simulation of electromagnetic transients (Chapter [29](#))
- Motor Starting Functions (Chapter [51](#))
- Modal / Eigenvalue Analysis (Chapter [32](#))
- Model Parameter Identification (Chapter [31](#))

## 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

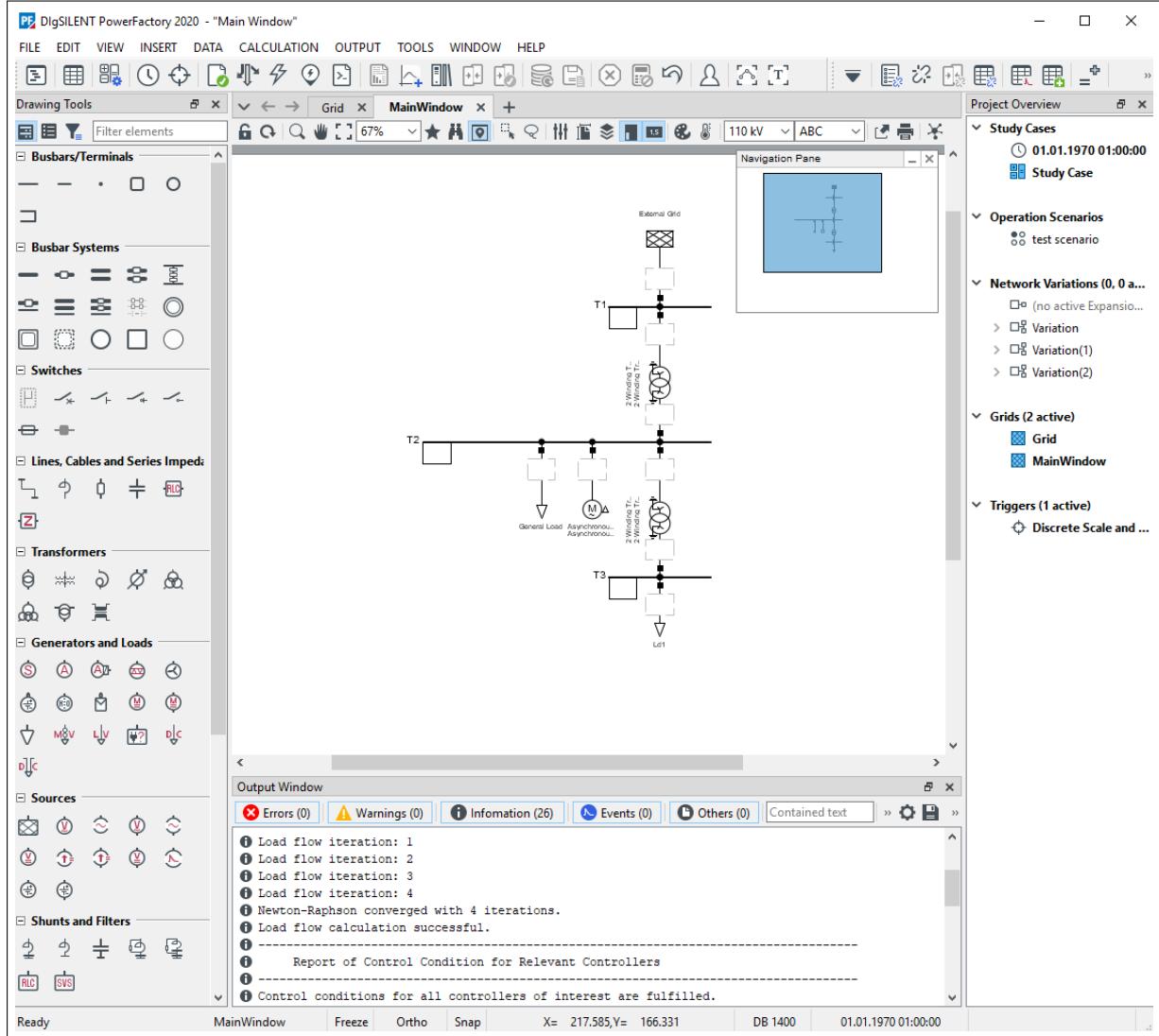


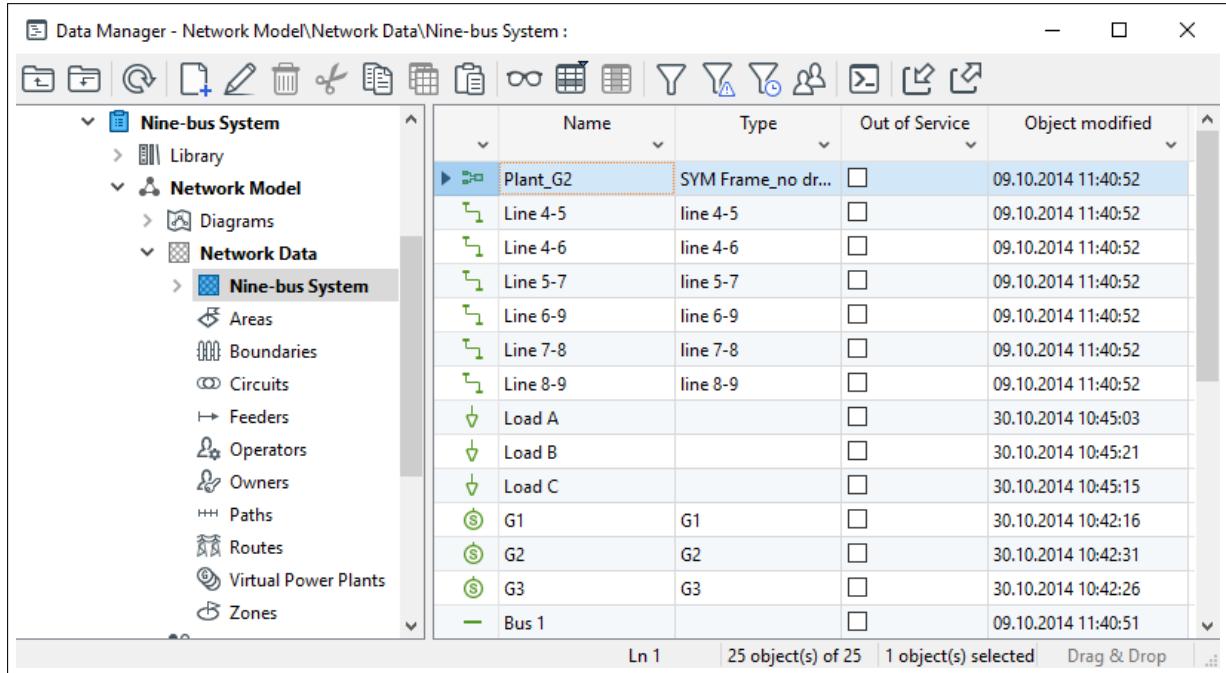
Figure 4.3.1: *PowerFactory* Main Window

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.



	Name	Type	Out of Service	Object modified
▶	Plant_G2	SYM Frame_no dr...	<input type="checkbox"/>	09.10.2014 11:40:52
↳	Line 4-5	line 4-5	<input type="checkbox"/>	09.10.2014 11:40:52
↳	Line 4-6	line 4-6	<input type="checkbox"/>	09.10.2014 11:40:52
↳	Line 5-7	line 5-7	<input type="checkbox"/>	09.10.2014 11:40:52
↳	Line 6-9	line 6-9	<input type="checkbox"/>	09.10.2014 11:40:52
↳	Line 7-8	line 7-8	<input type="checkbox"/>	09.10.2014 11:40:52
↳	Line 8-9	line 8-9	<input type="checkbox"/>	09.10.2014 11:40:52
↳	Load A		<input type="checkbox"/>	30.10.2014 10:45:03
↳	Load B		<input type="checkbox"/>	30.10.2014 10:45:21
↳	Load C		<input type="checkbox"/>	30.10.2014 10:45:15
↳	G1	G1	<input type="checkbox"/>	30.10.2014 10:42:16
↳	G2	G2	<input type="checkbox"/>	30.10.2014 10:42:31
↳	G3	G3	<input type="checkbox"/>	30.10.2014 10:42:26
↳	Bus 1		<input type="checkbox"/>	09.10.2014 11:40:51

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 the Global 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:

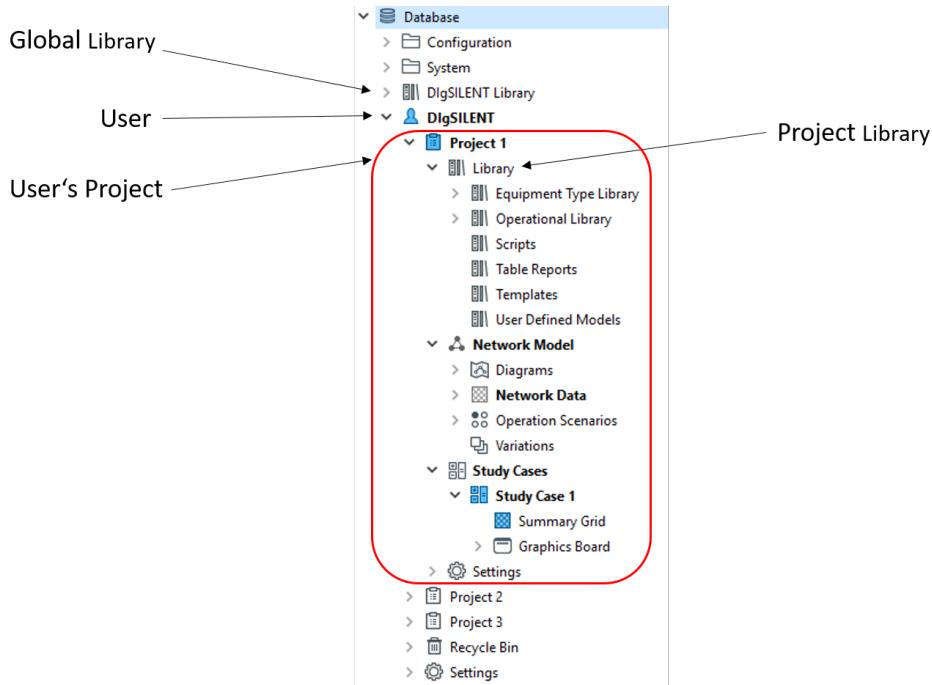


Figure 4.5.1: Structure of a *PowerFactory* project in the Data Manager

### 4.5.1 Global Library

The Global Library (called *DlgsILENT Library*) is supplied with *PowerFactory* and should not be edited by the customer. When a database is migrated to a new version, the Global 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. If the user wants to include user-specific models etc, a User-defined library can be created (see section 4.5.2)

The Global Library contains a wide range of pre-defined models, including:

- Type data for standard components such as conductors, motors, generators, and transformers.
- Standard control system frames, models, and macros (i.e. transfer functions and logic blocks, etc.).
- Standard CT, VT, fuse, and relay models.
- Pre-defined model templates, including:
  - Battery System with frequency control (10 kV, 30 MVA).
  - Double Fed Induction Wind Turbine Generator (0.69 kV, 2 MW).
  - Fully Rated Converter Wind Turbine Generator (0.4 kV, 2 MW).
  - Variable Rotor Resistance Wind Turbine Generator (0.69 kV, 0.66 MW).

- Photovoltaic System (0.4 kV, 0.5 MVA)
- Standard DPL scripts, including scripts to:
  - Minimise the Net Present Value of project (Variation) costs by varying the project service date.
  - Conduct time-sweep load flow calculations.

Documentation about the various elements and types in the Global Library can be found directly in the description page of the elements, in the Appendix B: Standard Models in *PowerFactory* and in [Technical References Document \(Templates\)](#).

#### 4.5.1.1 Versioning in the Global Library

All the main objects (types, models etc.) in the Global Library are managed using versions, for example a Type object might be at v002.1, where the main version number (002 in this example) is incremented when a material 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.

A version page shows the version number and a Change Log to show the modification history.

The distinction between the two types of change is important because when the global library is updated due to the installation of a new version of *PowerFactory* by the user, all minor version changes will be implemented automatically, whereas main version updates will not be done automatically. Instead, users can decide whether to use the updated models, or continue using the older versions, which will be automatically retained in subfolders.

#### Reporting on versions used and new versions available

A user who has updated to a new version of *PowerFactory* may wish to know what new versions for models, types etc. are available.

The following actions can be taken to review the use of standard models etc. from the Global Library in a particular project:

- Right-click on the project and *Show external types*.
- Types etc. from the Global Library (and other external libraries) are listed in the output window.
- The Global Library objects can be opened up from the list in the output window and the user can then see whether later versions are available, and what has been changed in the later versions.
- 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.

#### 4.5.2 User Defined Global Library

Sometimes it is useful to share a library with another *PowerFactory* user in a Global Library. This requires the creation of a library folder, which can be done by the Administrator as shown in Figure 4.5.2. To do this, the Administrator must proceed as follows:

- Right-click on the “Database”
- Click on *New → Folder*
- Choose a library name and select “Library” as *Folder Type*
- Click **OK**

The new Library is now created at the same level of the hierarchy as the Global Library. Every user of this database has full read access, however the Types within this Library must be created and edited by the Administrator.

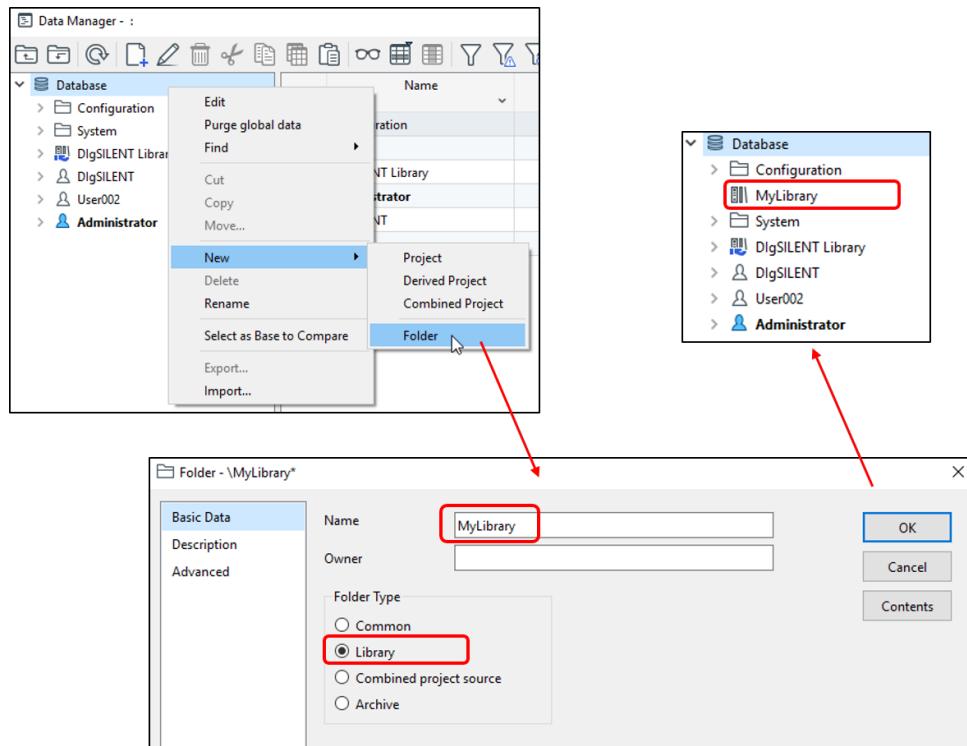


Figure 4.5.2: Creation of a user-defined Global Library

### 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 (Project Library).

### 4.5.4 Diagrams

Single line graphics are defined in *PowerFactory* by means of graphic folders of class *IntGrfNet* (Diagram icon). Each diagram corresponds to a *IntGrfNet* folder. They are stored in the *Diagrams* folder (Diagram icon) 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 *Graphics Board*. If a reference to a network diagram is stored in a study case's Graphics Board, when the study case is activated, the diagram is automatically opened. Diagrams can be easily added and deleted from the Graphics Boards.

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 Graphics Board.

Chapter 9 (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 red, to distinguish it from inactive Grids.

---

For details of how to define grids refer to Chapter 8.Basic Project Definition, Section 8.2 (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 8.1.2 (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:

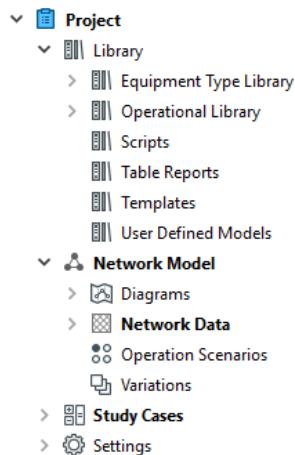


Figure 4.6.1: Basic Project Hierarchy

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.

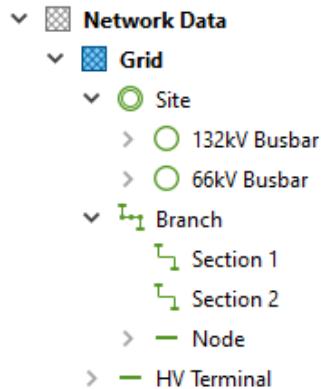


Figure 4.6.2: Example Project Structure

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 11: 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 Chapter 14: Project Library, Section 14.3: Operational Library).

For information about how to define substations refer to Chapter 11: 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 11: 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 11: Building Networks, sections 11.2 and 11.5.

## 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, together with the graphics board.

#### 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:

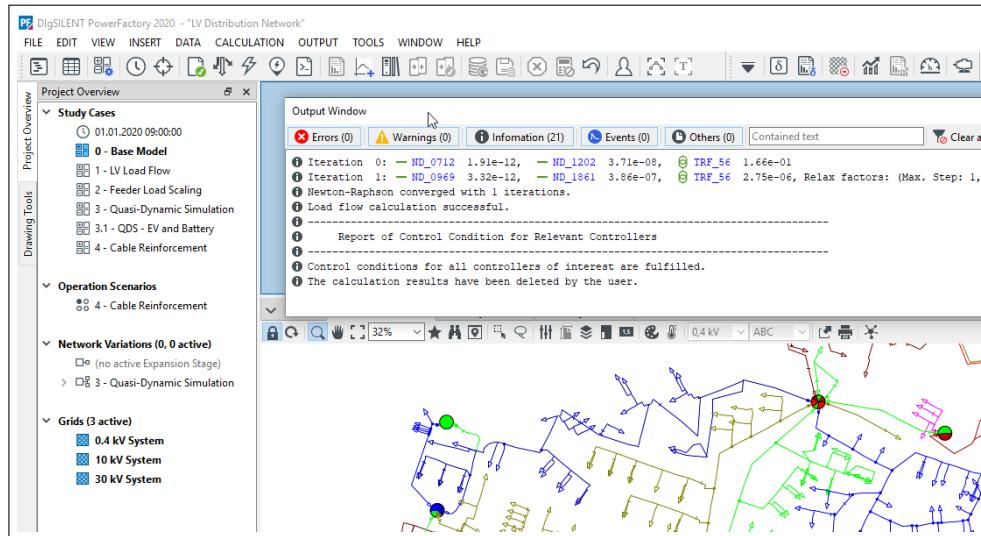


Figure 4.7.1: Docking a tool window

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 graphic board layout, split etc., as this is a study-case specific configuration.

#### Graphics board

The graphics board, sometimes called the graphical editor, uses a tabbed view concept, enabling a

number of graphics to be open at any one time. 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 either vertically or horizontally at any one 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 9.2.5.

#### 4.7.1.2 Key Features of the user interface

The main *PowerFactory* window is shown in Figure 4.7.2

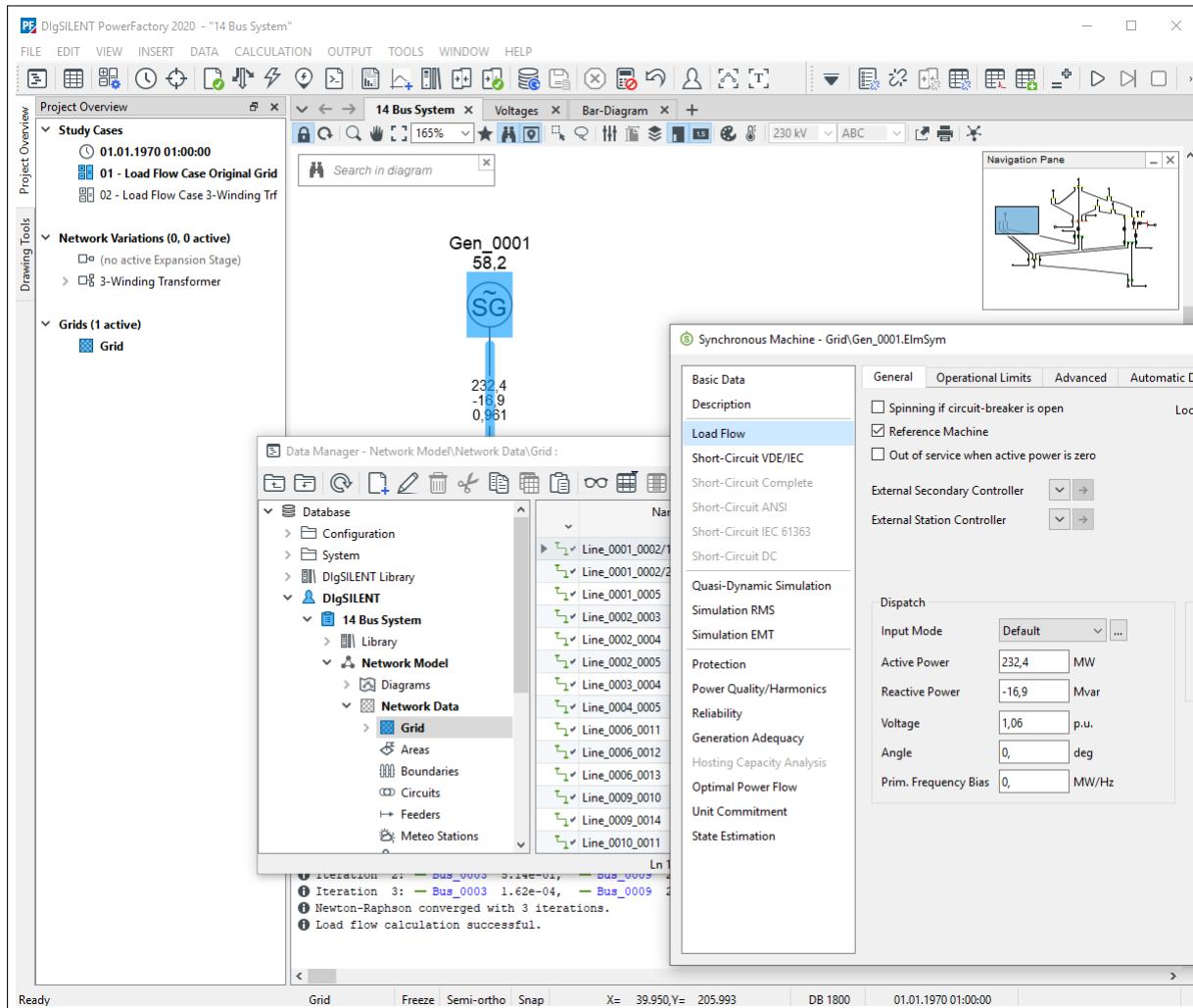


Figure 4.7.2: *PowerFactory* user interface

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 4.7.2 (Menu Bar).
3. The Main Toolbar includes commands and other icons. The Main Toolbar is discussed in further detail in Section 4.7.3 (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* sensitive 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 cannot be closed, but can be minimised. The output window is discussed in further detail in Section 4.7.4 (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 9.2.6 for more information.)

## 4.7.2 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.

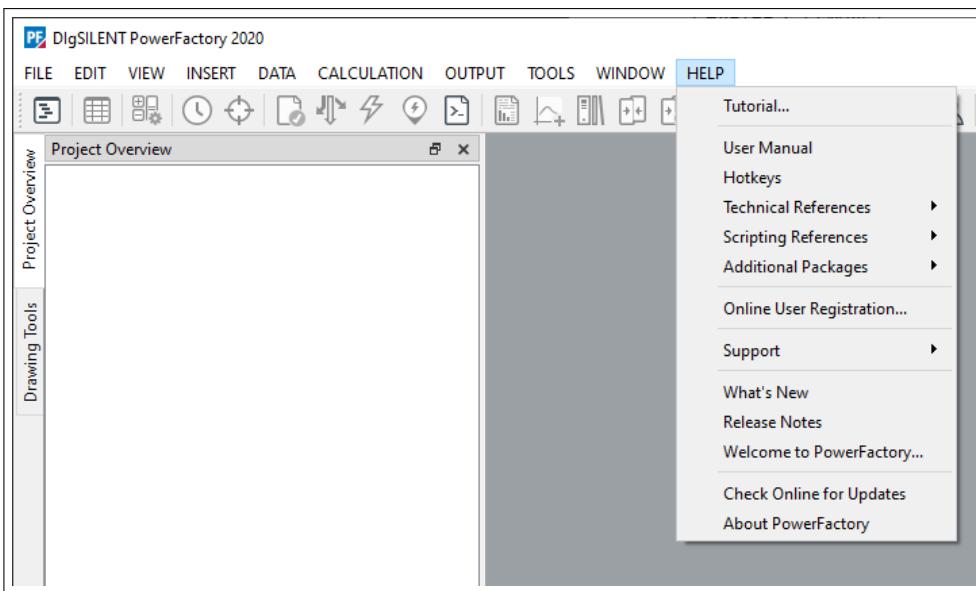


Figure 4.7.3: The help menu on the Menu bar

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.

### 4.7.3 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 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 9 (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. When the option “Use multiple Data Manager” is enabled in the user settings menu (*User Settings → Data/Network Model Manager*) the user will be able to open as many instances of the Data Manager as required. If “Use multiple Data Manager” is disabled, the first instance of the Data Manager will be re-opened. For more information on the Data Manager refer to Chapter 10.

## 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. For more information, see Chapter 12: 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 Calculation 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 26.6: 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 *Calculate* → *Multiple Faults* a Short-circuit event list will be created with these two busbars in it.

## Execute Scripts

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.

## Output Calculation Analysis

Presents calculation results in various formats. The output is printed to the output window and can be viewed, or copied for use in external reports. Several different reports, depending on the calculation, can be created. For more information about the output of results refer to Chapter 19: Reporting and Visualising Results, Section 19.4.2.

 **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 19.7.

 **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 19.4.1.

 **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 19.5.

 **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 19.5.

 **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.

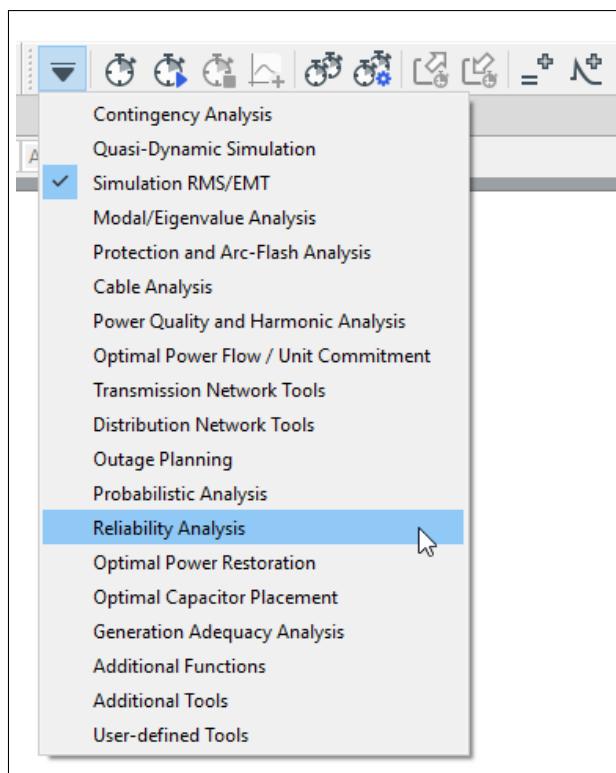


Figure 4.7.4: Change Toolbox selection

## 4.7.4 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.

### 4.7.4.1 Sizing and Positioning the Output Window

The default location of the output window is “docked” (fixed) at the bottom of the main window, but like the other tool windows it can be moved around or resized.

Two additional options for the output window are:

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

#### 4.7.4.2 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 sensitive menu by right-clicking the mouse in the output window pane.

-  Opens the User Settings on the *Output Window* page.
-  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.

#### 4.7.4.3 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 double-clicked with the left mouse button to open an edit dialog for the object. Alternatively, the object can be right-clicked and then *Edit*, *Edit and Browse Object*, or *Mark in Graphic* selected. This simplifies the task of locating objects in the single line graphic.

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

#### 4.7.4.4 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.

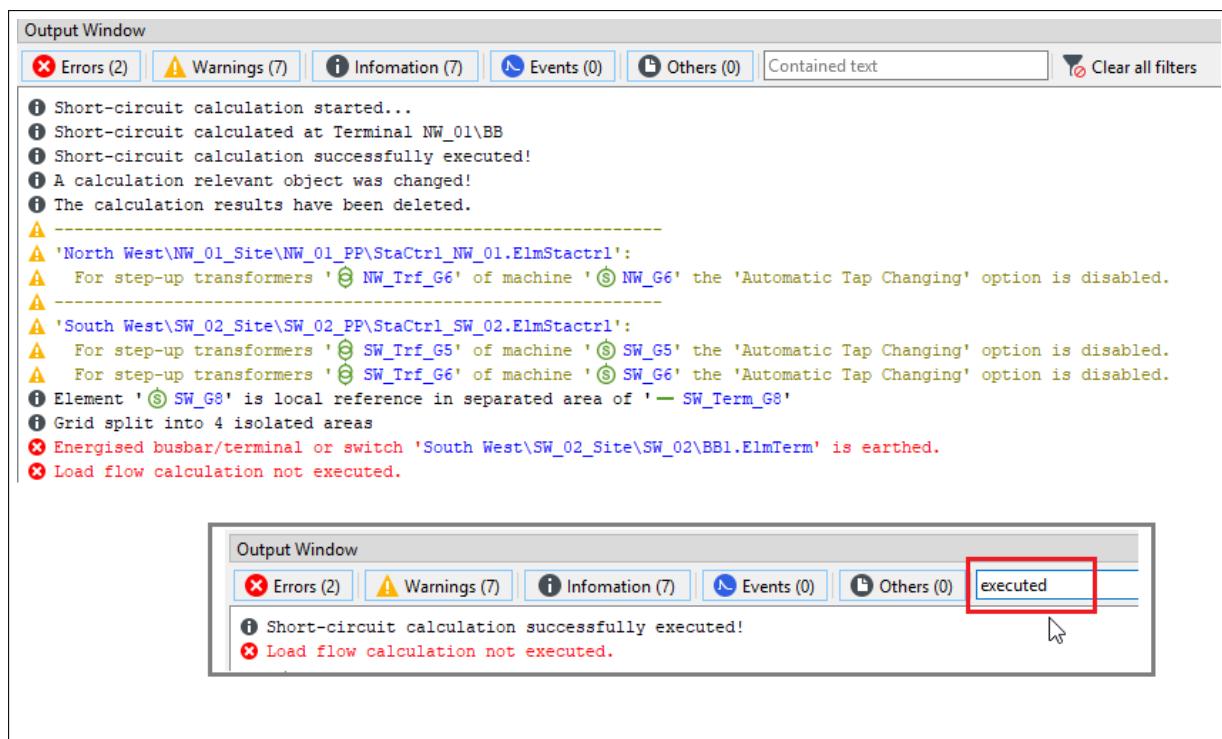


Figure 4.7.5: Output window

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.

#### 4.7.4.5 Output Window Graphical Results

Reports of calculation results may contain bar graphical information. The “voltage profiles” report after a load flow calculation, for instance, produces bar graphs of the per-unit voltages of busbars. These bars will be coloured blue, green or red if the option *Show Verification Report* in the Load Flow Calculation command has been enabled. They will be cross-hatched if the bars are too large to display.

Part of a bar graph output is shown in Figure 4.7.6 The following formatting is visible:

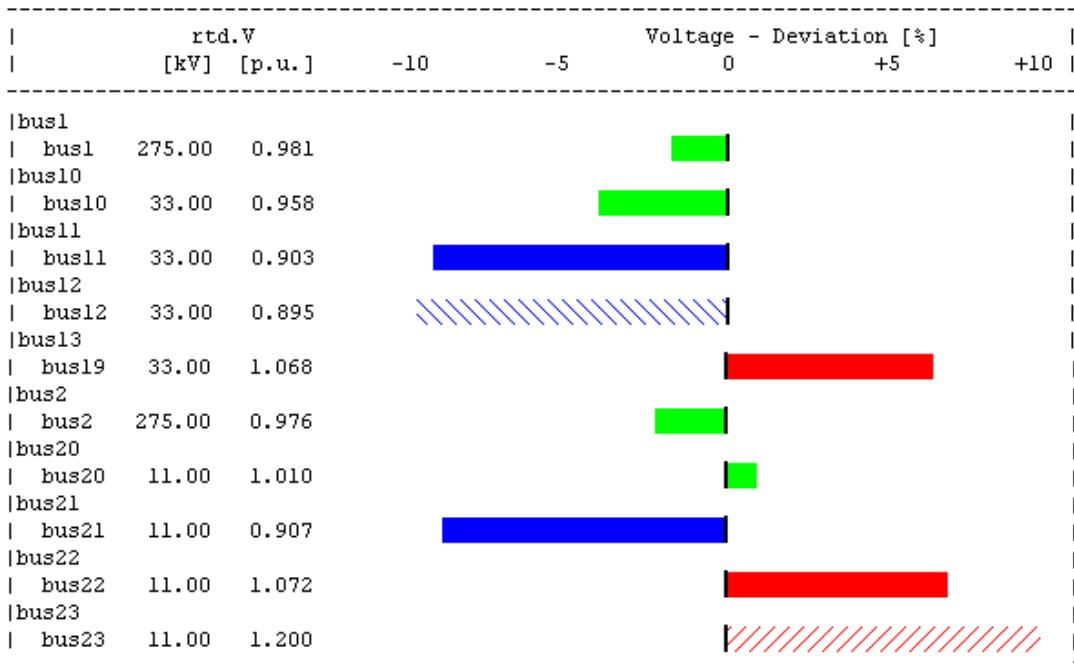


Figure 4.7.6: Output window bar diagram

- Green Solid Bar: Used when the value is in the tolerated range.
- Blue Solid Bar: Used when the value is below a limit.
- Red Solid Bar: Used when the value is above a limit.
- Cross-hatched Bar: Used when the value is outside the range.

#### 4.7.4.6 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.

## 4.8 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.

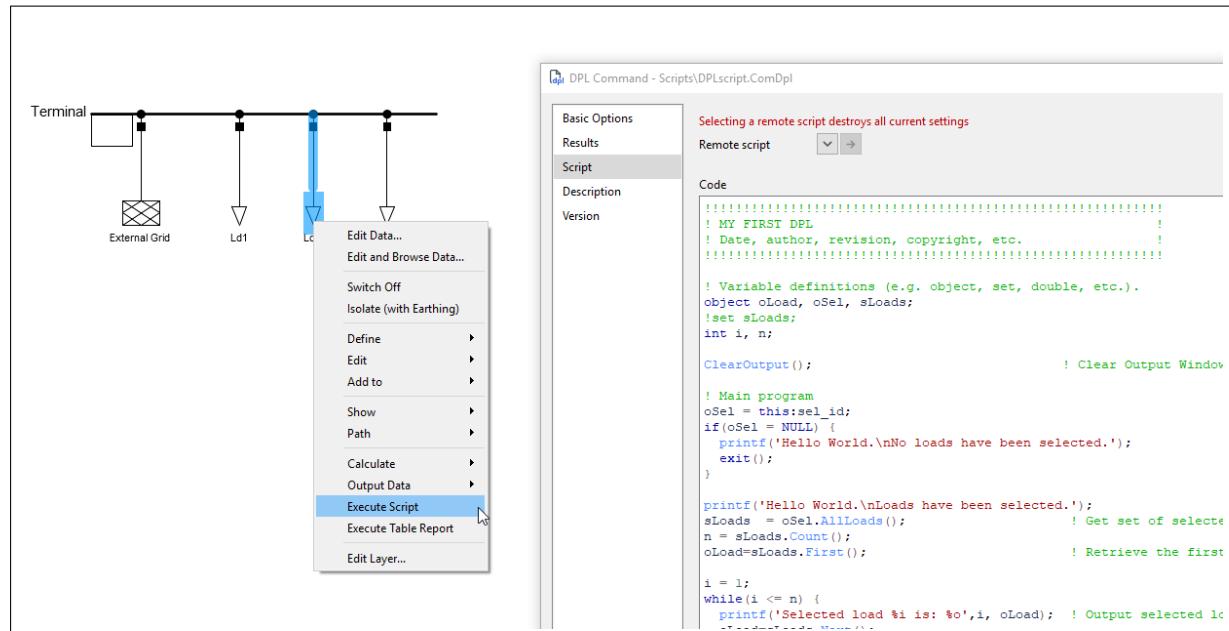


Figure 4.8.1: Example DPL Script

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

# Program Administration

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](#).

## 5.1 Program Installation and Configuration

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. If more user accounts are subsequently created, the user can switch to other users, or the Administrator account can be used to change the login policy so that a dialog is presented when the user starts to log in. See section [6.4.2](#) for more information.

In a multi-user-database installation (see Chapter [6](#): User Accounts, User Groups, and Profiles) new accounts and passwords are created by the administrator. The 'Administrator' account is created when installing *PowerFactory* and is used to create and manage users' accounts in a multi-user environment (see Chapter [6](#): User Accounts, User Groups, and Profiles). To log on as Administrator, the shortcut from the Windows Start Menu can be used. When already running a *PowerFactory* session, the user can select *Tools* → *Switch User* in the main menu to log-on as Administrator.

For further information about the role of the database administrator refer to Section [6.2](#): The Database Administrator. Many of the activities carried out by the Administrator are easily accessed using 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.

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. Administrator rights are necessary to perform changes to these settings.

## 5.2 PowerFactory Configuration

The configuration of the application is stored in an “ini” file located with the executable. These settings can be changed within *PowerFactory* via the Configuration (*SetConfig*) dialog, which is available via *Tools* → *Configuration*. Depending on where the 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.

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

- Oracle
- Microsoft SQL Server

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 Chapter [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.
- **Interpreter:** 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.

### Visual Studio

Here the *Version* and *Shell Extension* of Visual Studio can be set. Visual Studio is used for the compilation of DSL models.

### 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).

## 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 configuration

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 for Business account**

If Google Maps is to be used as the map provider, the “Google Maps for Business 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)**: The digex\* libraries contain the compiled dynamic models.

#### Advanced tab

Settings on the *Advanced* tab should only be changed under the guidance of the DIgSILENT 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 DIgSILENT PowerFactory Licence Manager, its dialog can be accessed via *Tools* → *Licence* → *Select Licence*...

---

**Note:** The DIgSILENT 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 *DlgSILENT* 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](#).

## 5.4 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.

#### 5.4.4 Import Workspace from 14.X or 15.0...

This option allows the migration of the database from an older *PowerFactory* version (e.g 14.X, 15.0...) to the newest version.

This can be selected from the main menu, under *Tools* → *Workspace* → *Import Workspace from 14.x or 15.0...*. After “Import Workspace from 14.x or 15.0...” has been selected, the user can choose the working directory. The database that is saved in selected working directory will be migrated.

Depending on the database size, a migration may take several hours. The user is given the choice between a complete migration or minimal migration, where the only database structure is migrated, and then the individual projects are migrated later on, as they are activated. The latter option shortens the time required for the initial database migration.

##### 5.4.4.1 Migration Types

**Complete:** the database structure and all projects will be altered and migrated immediately upon pressing the OK button.

**Minimal:** the database structure will be altered immediately, but the project migration will occur upon activation.

Minimal migration is recommended for the migration of large databases.

## 5.5 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.

### 5.5.1 Functionality in Offline Mode

#### 5.5.1.1 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 he/she is 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 5.5.1)

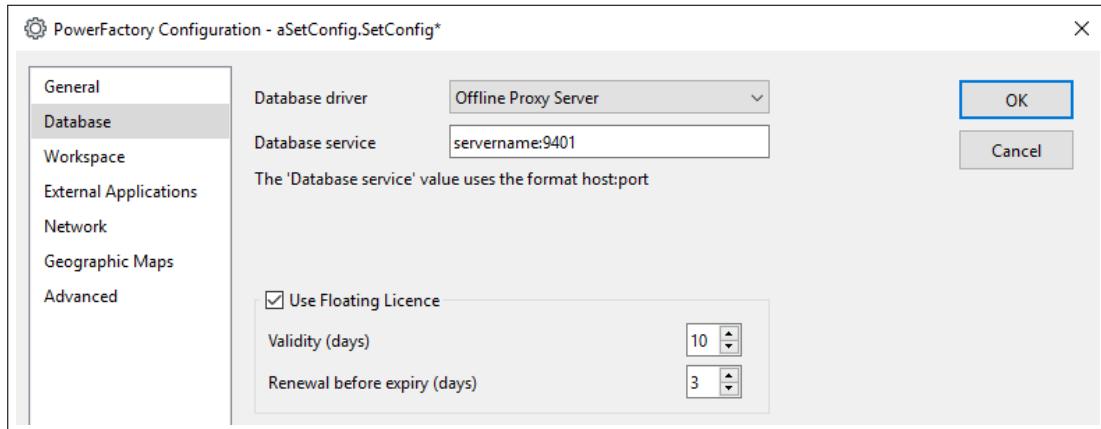


Figure 5.5.1: Log-on dialog, *Database* page

**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.

### 5.5.1.2 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.

### 5.5.1.3 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*

#### 5.5.1.4 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*.

#### 5.5.1.5 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*.

#### 5.5.1.6 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*.

#### 5.5.1.7 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*.

#### 5.5.1.8 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*

## 5.5.2 Functionality in Online Mode

### 5.5.2.1 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 .

## 5.5.3 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**

## 5.6 Housekeeping

### 5.6.1 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 *PowerFactory Advanced Installation and Configuration Manual*.

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

### 5.6.3 Configuring Housekeeping Tasks

The Housekeeping command (*SetHousekeeping*) is used to control which housekeeping tasks are enabled. It is recommended that the user move this object from Database\System\Configuration\Housekeeping to Database\Configuration\Housekeeping, in order to preserve the user's configuration throughout database upgrades.

The following sections discuss the different housekeeping tasks available in the Housekeeping dialog.

#### 5.6.4 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 vault directory.
- **Deferred archiving by Housekeeping job:** by selecting “Archive” from the context menu, the project will be immediately archived, but not placed in the vault directory. This will happen automatically depending on the Housekeeping settings.

**Important:** The vault directory can be defined under “Tools\Configuration\Database \Vault Directory”

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 vault 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.

#### 5.6.5 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 using a template project (under Configuration/Default in the Data Manager tree).

The settings for multiple projects can be changed in a data manager on the Storage tab, as shown below in Figure 5.6.1. A value of ‘1’ is equivalent to the Housekeeping option *Delete project* being selected.

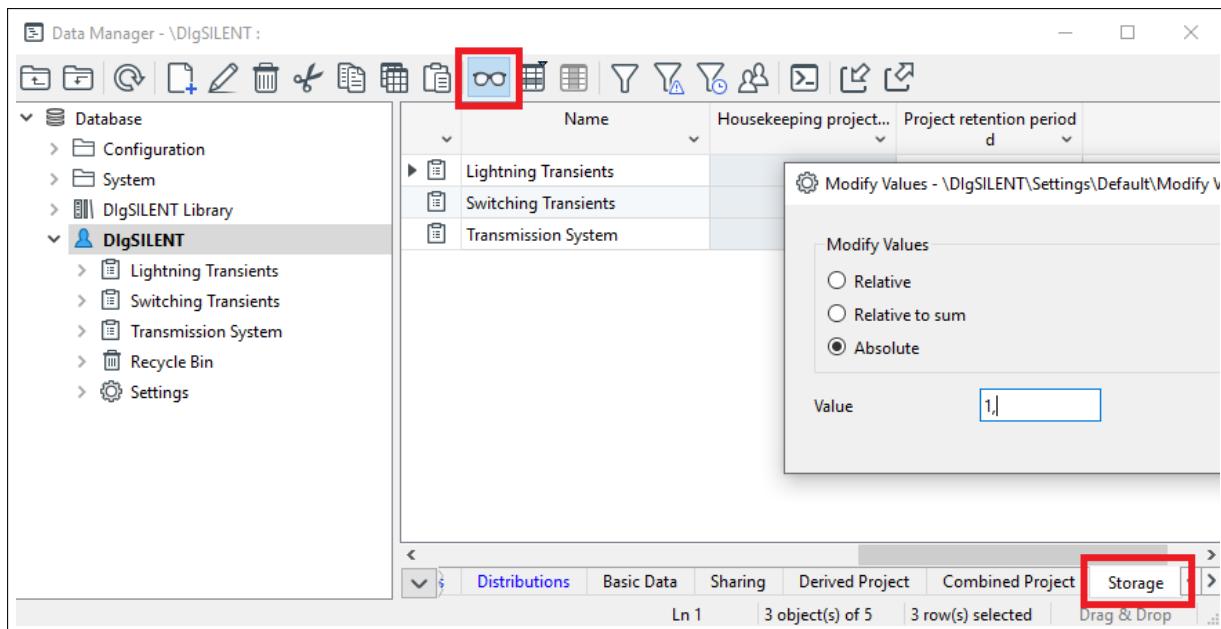


Figure 5.6.1: Setting parameters for multiple projects

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 5.6.2).

## 5.6.6 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 5.6.5 for details).

### 5.6.7 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.

### 5.6.8 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.

### 5.6.9 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 5.6.2.
3. Configure the Housekeeping dialog as described in Section 5.6.3.
4. If using the project deletion/archiving task, configure automatic deletion/archiving properties for new projects, as described in Section 5.6.5.
5. If using the project deletion/archiving task, configure automatic deletion/archiving properties for existing projects, as described in Section 5.6.5.
6. Regularly monitor the HOUSEKEEPING\_LOG table to verify the status of housekeeping runs, as described in Section 5.6.8.

# Chapter 6

# User Accounts, User Groups, and Profiles

This chapter provides details of how to create and manage user accounts, user groups, and profiles. The information in this chapter is particularly relevant for a multi-user database (i.e. *Team Edition*), and will not generally be of so much 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 default name for a *PowerFactory* user (unless using *Team Edition*) is the Windows user name, which is automatically created when *PowerFactory* is started for the first time.

## 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

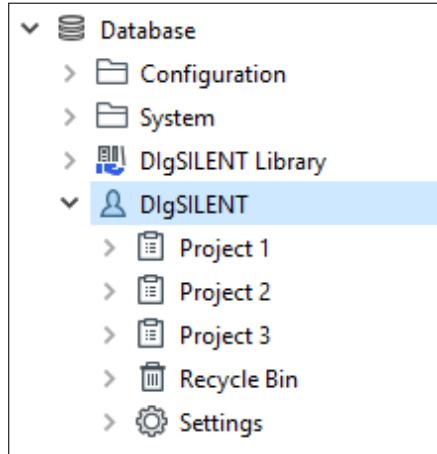


Figure 6.1.1: Basic database structure

## 6.2 The Database Administrator

A database administrator account is created with the *PowerFactory* installation. The main functions of the administrator are:

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

Under a multiuser database environment, 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 6.6 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 main library and the system folders. 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 (Administrator)*.

- Log into *PowerFactory* as a normal User and select via the Main menu *Tools* → *Switch User*. Select **Administrator** and enter the corresponding password.

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

## 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 [6.5](#) and [6.6](#).

### 6.3.2 Security and Privacy

The options available here are:

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

These are described in Section [6.4](#) below.

### 6.3.3 Calculation Settings

The options available here are:

- Parallelisation...This allows the administrator to configure the Parallel Computing Manager.
- 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 [39.3.6.2](#).

### 6.3.4 Housekeeping

This gives access to the dialog for setting up Housekeeping tasks. Details can be found in Section [5.6](#).

## 6.4 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](#).

### 6.4.1 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.

#### 6.4.1.1 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.

#### 6.4.1.2 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](#).

### 6.4.2 Login Policy Options

#### 6.4.2.1 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.

#### 6.4.2.2 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.

#### 6.4.3 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” includes mouse-clicks, keyboard actions or scripts or calculations running. As well as being a useful security measure, setting an Idle Session Timeout is useful in a multi-user environment because the licences are released back to the licence server.

#### 6.4.4 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.

#### 6.4.5 Privacy

This feature is available to allow database administrators 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.

### 6.5 Creating and Managing User Accounts

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. (see Chapter 5: Program Administration). In this case the program will automatically create and activate the new account, without administrator intervention. In order to create other *PowerFactory* users if required, the ‘User Manager’ object can be used as described below:

In multi-user database installations, the administrator creates new user accounts by means of a tool called the ‘User Manager’, which is found in the Configuration folder.

To create a new user:

- Log on as Administrator. You can do so by starting the *PowerFactory* Administrator shortcut in the Windows Start menu or by switching the user via *Tools* → *Switch User* in the main tool bar.
- The User Manager can be accessed from the Administration menu on the main toolbar: *Administration* → *User Management* → *User Manager...*
- Press the **Add User...** button.

The User edit dialog will be displayed. The settings are the following:

- *General page*
  - User Name: user Name that will be used for login to *PowerFactory* at startup
  - 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.
  - 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. If User A is in the list of permitted user, User A can access the user account without entering the user password.
- *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
  - User account is time-limited: this option will set the account to be time limited and therefore can be used for temporary users within the database.
  - Force Authentication server usage: setting this option also requires the definition of an authentication server within the *PowerFactory* configuration as explained in the manual.
- *Password Policy page*: On this page, the default Password policy (see section 6.4.2) 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 6.6 below.
- *Parallel Computing*: 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*: the *Unit Commitment* module (see Chapter 39) 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. To edit an existing user account:

- Right-click on the desired user and select *Edit* from the context sensitive menu. The User edit dialog will be displayed.

Any user can edit her/his own account by means of the User edit dialog. In this case only the full name and the password 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).

---

## 6.6 Creating 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](#)) and for controlling access to licence modules.

User groups are created by the administrator via the User Manager. To create a new user group:

- Log on as Administrator.
- The User Manager can be accessed from the Administration menu on the main toolbar: *Administration* → *User Management* → *User Manager...*
- 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 sensitive menu). Via the context sensitive 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 sensitive 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 sensitive 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).

## 6.7 User Interface Customisation (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 *Others* → *Settings* → *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.

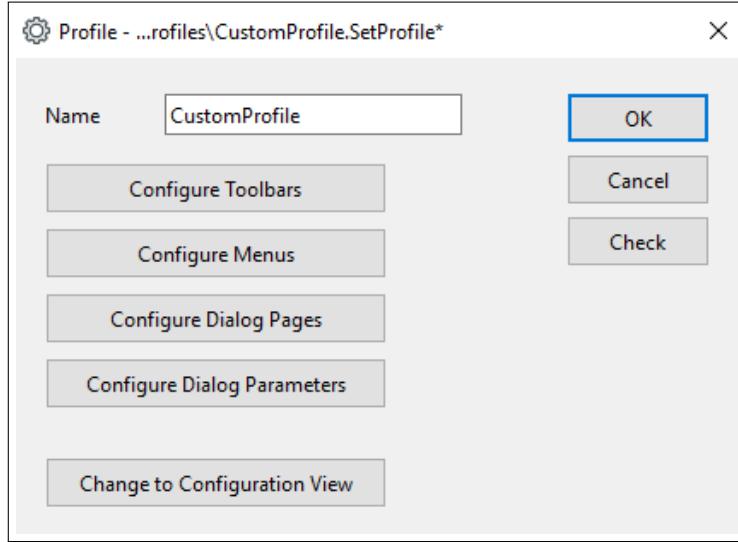


Figure 6.7.1: Profile dialog

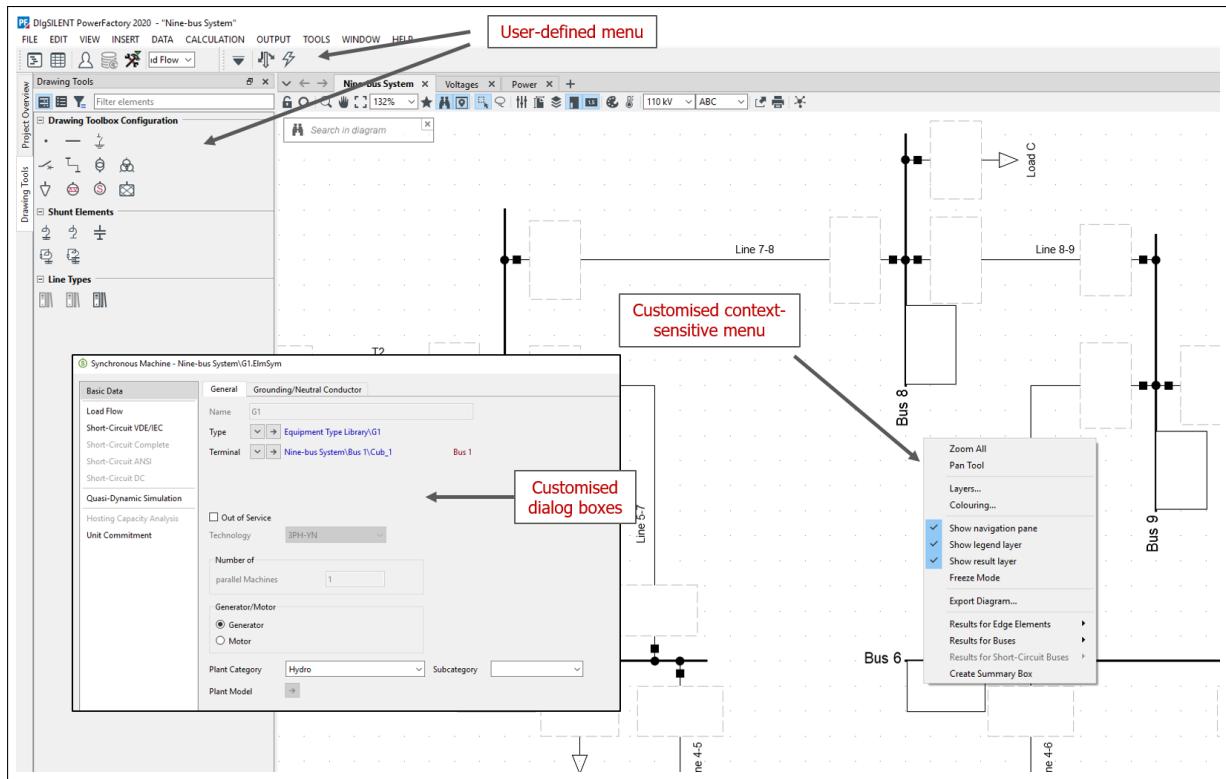


Figure 6.7.2: GUI Customisation using Profiles

## 6.7.1 Tool Configuration

### Definition of Icons

Icons can be defined in the *Configuration → Icons* folder by selecting the *New Object* icon and then

*Others* → *Other Elements* → *Icon (IntIcon)*. 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).

### Command Configuration

The User-defined Tools toolbar can be used to make commonly-used tools such as scripts and Add On Modules available to users. Changes and additions to the User-defined Tools toolbar can only be made by the administrator; from the top menu, *Tools* → *Tool Configuration...* is selected and 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.
- **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).

### Template Configuration

- **Template:** the name of the template. The name may be for a unique template, or include wildcards (such as \*.ElmLne) for selection of a group of templates. Templates should be in 'System/Library/Busbar Systems' folder, or in the 'Templates' folder of the active project.
- **Drawing mode:** the drawing mode can be set where there are multiple diagrammatic representations for a template (such as for a substation). Three options are available:
  - *Blank* will place the default (detailed) graphic of the template.
  - *Simplified* will place the simplified graphic of the template.
  - *Composite* will place a composite representation of the template.
- **Symbol name:** sets the representation of templates with a composite drawing mode (e.g. GeneralCompCirc or GeneralCompRect).
- **Icon:** previously created icons can be selected, which will be shown on the menu where the template is placed. If no icon is selected, a default icon will appear (a Template symbol or custom icon).
- **Description:** this description will be displayed when a user hovers the mouse pointer over the icon. If left blank, the template name will be displayed.

### 6.7.2 Configuration of Toolbars

The Main Toolbar and Drawing Toolbars can be customised using the Toolbar Configuration. The field *Toolboxes* may either refer to a *Toolbox Configuration* (*SetTboxconfig*) or a *Toolbox Group Configuration* (*SetTboxgroconfig*), which may in-turn refer to one or more *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 *Basic Commands* and *Custom Commands* groups of commands.

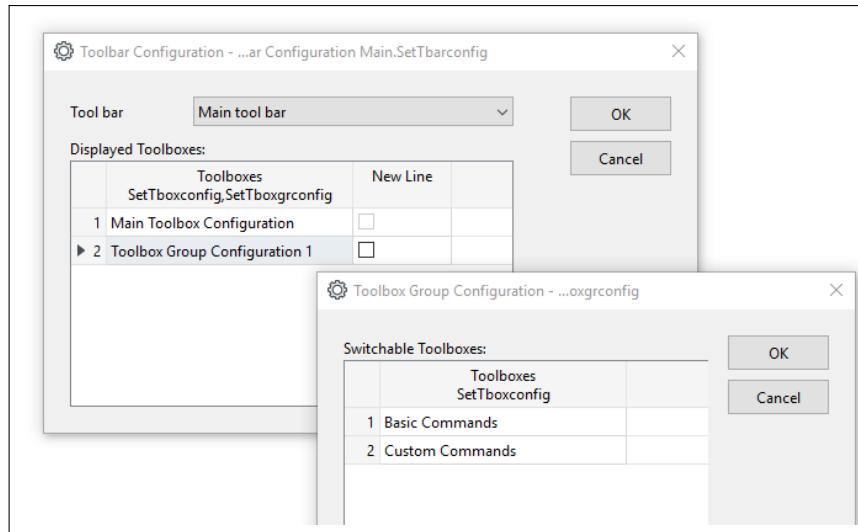


Figure 6.7.3: Toolbar Configuration

Each toolbox can be customised to display the desired icons, such as illustrated in Figure 6.7.4

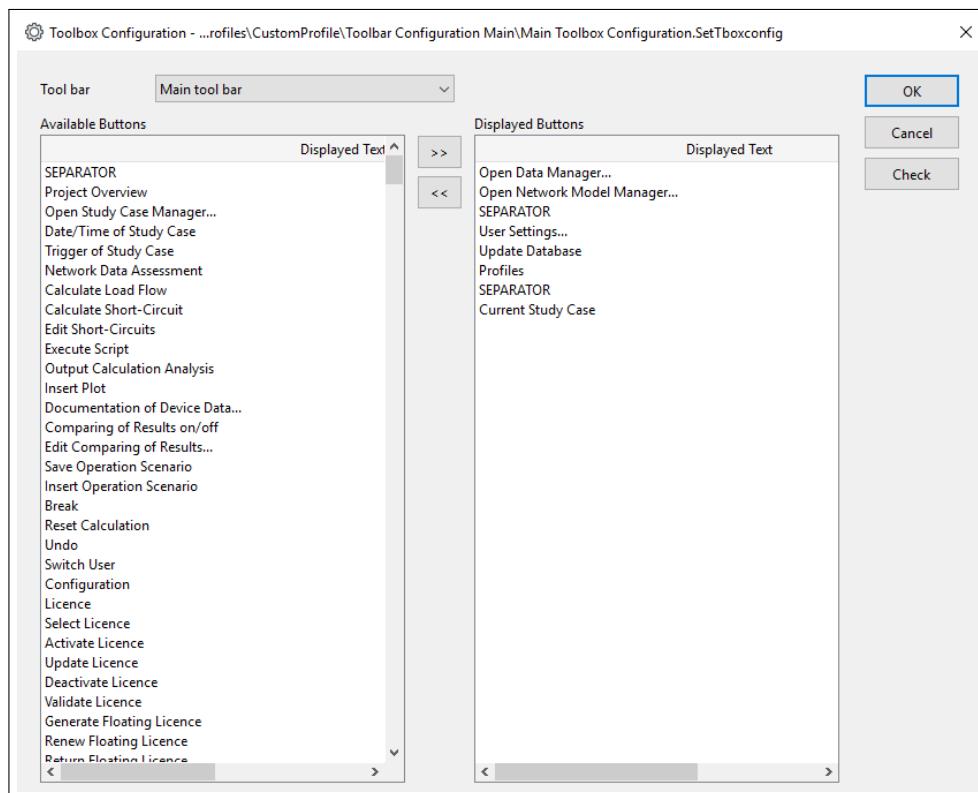


Figure 6.7.4: Toolbox Configuration

Prior to customising the displayed buttons and menu items etc, the user should first define any required custom Commands and Templates. A *Tool Configuration object* can be created in the *Configuration → Profiles* folder, or within a user-defined Profile, by selecting the *New Object* icon and then *Others → Settings→ Tool Configuration*. If created in the *Profiles* folder, the commands will be available from the "Standard" profile. Conversely, if the Tool Configuration object is created within a profile (*SetProfile*) the commands and templates will only be available for use in this profile. If there is a Tool Configuration within a user-defined profile, as well as in the *Profiles* folder, the Tool Configuration in the user-defined

profile will take precedence. Optionally, customised icons can be associated with the Commands and Templates.

### 6.7.3 Configuration of Menus

The *Main Menu*, *Data Manager*, *Graphic*, *Plots*, and *Output Window* menus can be customised from the *Menu Configuration* dialog. The *Change to Configuration View* button of the Profile dialog is used to display description identifiers for configurable items, such as illustrated in the context-sensitive menu shown in Figure 6.7.5. 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).

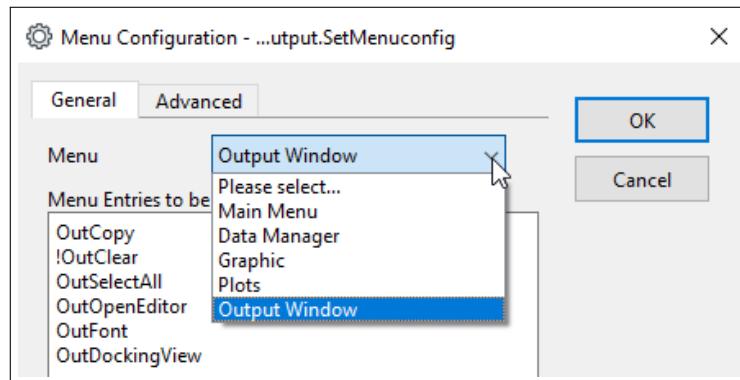


Figure 6.7.5: Menu Configuration

### 6.7.4 Configuration of Dialog Pages

The *Dialog Page Configuration* may be used to specify the Available and Unavailable dialog pages shown when editing elements, such as illustrated in Figure 6.7.6. Note that Users can further customise the displayed dialog pages from the *Functions* tab of their *User Settings*.

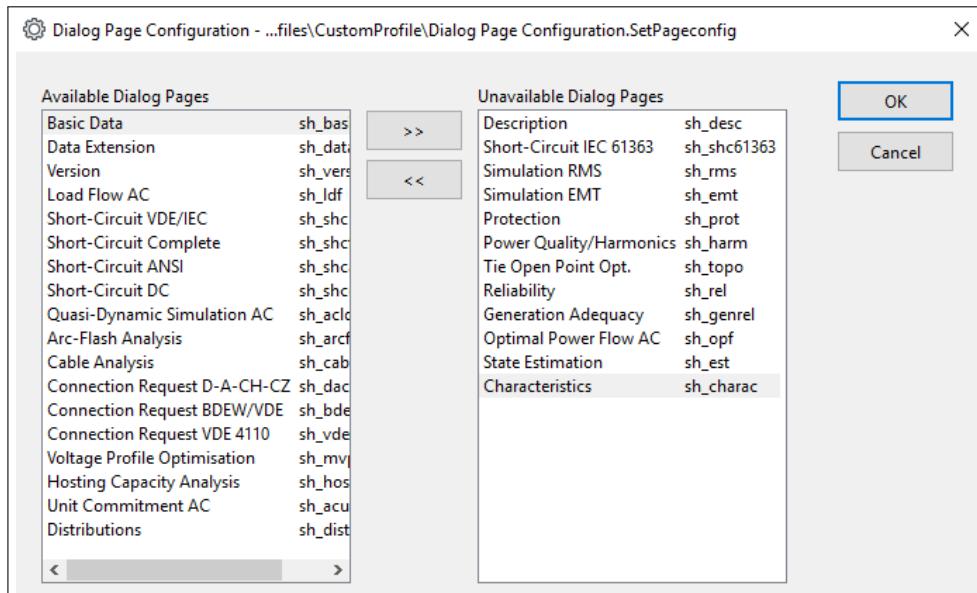


Figure 6.7.6: Dialog Page Configuration

### 6.7.5 Configuration of Dialog Parameters

The *Dialog Configuration* may be used to customise element dialog pages, such as illustrated for a Synchronous Machine element in Figure 6.7.7. “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.

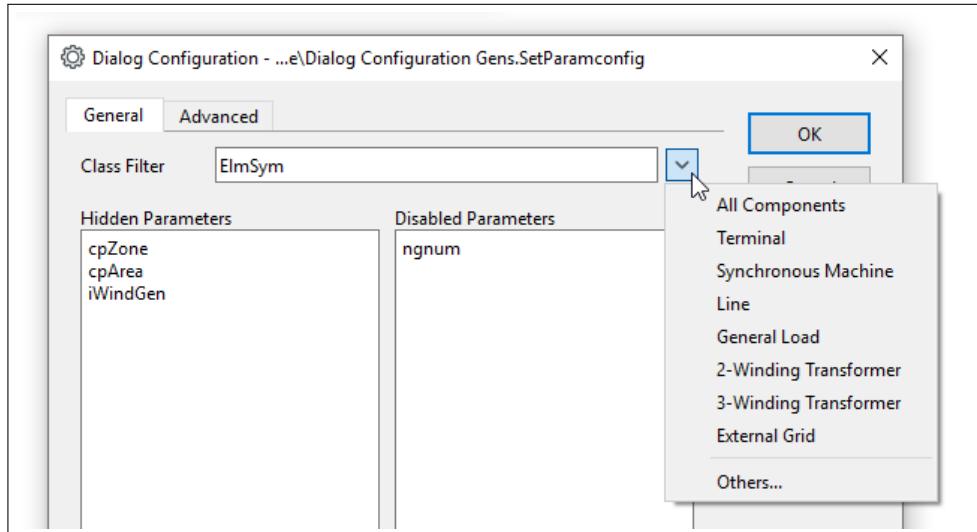


Figure 6.7.7: Dialog Configuration

**Note:** Configuration of Dialog parameters is an advanced feature of *PowerFactory*, and the user should be cautious not to hide or disable dependent parameters. Seek assistance from *DlgSILENT* support if required.

### 6.7.6 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 Data/Network Model Manager Settings

The Data/Network Model Manager settings include:

#### Browser

- **Save Data Automatically.** 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 Activity.** Pops up a confirmation dialog 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.
- **Use multiple Data Manager.** When enabled, more than one Data Manager dialog can be opened at a time. When disabled only one Data Manager may be opened at a time and pressing the *New Data Manager* button will pop up the minimised Data Manager.

#### Operation Scenario

If *Save active Operation Scenario automatically* is enabled, the period for automatic saving must be defined.

#### 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.

### Folders for Global Library

The default global type folder is the *Database/Library/Types* folder. This default folder contains many predefined object types, but objects within this folder may not be changed by the user (read-only access). This option allows the user to specify a different "Global Type Folder", possibly a company specific and defined type library.

For information about the *PowerFactory* Data Manager refer to Chapter 10.

## 7.2 Window Layout

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

### 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.

## 7.3 Graphic Windows Settings

### 7.3.1 General tab

#### Open graphics board 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 → Graphic Board*.

#### 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.

#### Use custom background colour

If the option is enabled, the user can define the background colour for graphics, selecting from a drop-down menu.

### General Options

- **Diagram window margin:** The graphical pages are by default shown against a grey background, with a small margin. The size of that margin can be adjusted here.
- **Show Grid only if step size will be least:** Grid points smaller than the selected size will not be shown.
- **Show Text only if height will be least:** Text smaller than the selected size will not be shown.

## 7.3.2 Advanced tab

### Cursor

Defines the cursor shape:

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

### Default Font for New Graphic

When the *Use custom font*. is enabled, a customised font can be defined.

### 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.

### Allow resizing of branch objects

If the option is enabled, the user can left click an edge element within the single line graphic and then resize it.

### Show “Edit Graphic Object” in context sensitive 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

### 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 milisecond:** the date and time of the system up to the milisecond 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 the button **Font...**

#### Page Setup

If the user wishes to print out the contents of the output window, the button **Page Setup...** offers a range of settings which can be used to configure the output.

#### 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) for details.

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

## 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.
- **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.

### Font... and Colours...

Here the user can customise the text used inside the editor.

## 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. 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.

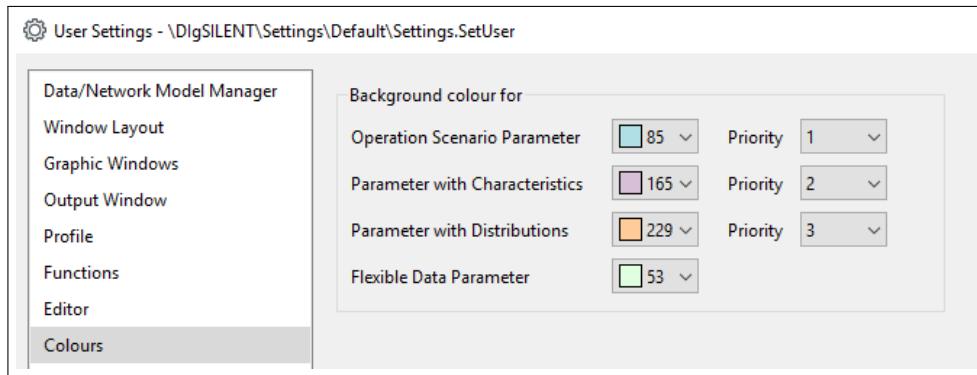


Figure 7.8.1: Data colouring options

## 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). Please 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 5.5). 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 22.4.

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 50s. 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

### Localisation

- **Decimal Symbol.** Selects the symbol selected to be used for the decimal point.
- **Use operating system Format for Date and Time.** The operating system date and time settings are used when this is checked.

### 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.

### System Stage Profile

This setting relates to the old system stage system used before the current Variations were introduced. The options define the extent to which the user can create or modify the “old” stages.

### 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 straightforwardly.

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

This option is only active if housekeeping is enabled.

### 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 ‘Example’ dialog at startup

This option can be deselected if the user does not want to see the Example dialog at each log-on. It can also be deselected directly from the dialog itself.

### 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 “Don’t ask again” option which can be 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.

# **Part III**

# **Handling**

# Chapter 8

## 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.

### 8.1 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 field at the bottom of the Element Selection window (*IntPrj*) (after selecting option *Others* in the *Elements* field). Note that names in *PowerFactory* are case-sensitive.
- 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-sensitive 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 8.1.1. 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-sensitive 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.

If the user wishes to change the default structure of the project, it can be modified from the Administrator account. The default structure is defined in a project held the folder: System, Configuration, Default.

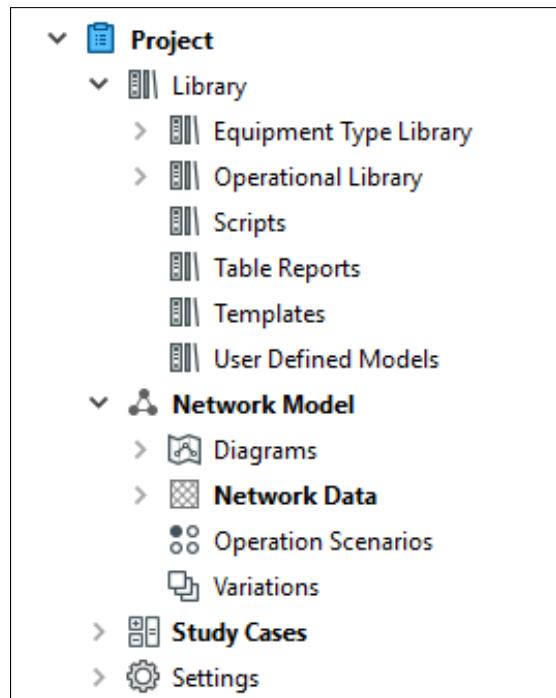


Figure 8.1.1: Default project structure

### 8.1.1 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-sensitive menu.

The *Basic Data* page contains basic project settings and allows the creation of new study cases and grids:

- Pressing the → button in the *Project Settings* field opens the Project Settings dialog (*SetProj*). See section [8.1.2](#) for more information regarding the settings of the project.
- Pressing the **New Grid** button will create a new grid and will open the grid edit dialog. A second dialog will ask for the study case to which the new grid folder should be added. For additional information about creating a new grid refer to Section [8.2](#) (Creating New Grids)
- The **New Study Case** button will create a new study case and will open its dialog. The new study case will not be activated automatically. For further information about creating study cases refer to Chapter [13](#): Study Cases, Section [13.2](#) (Creating and Using Study Cases).
- 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** in the *Changed Settings* field of the dialog. The settings can only be changed when a project is inactive.
- 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 on the network.
- The name of the active study case is shown in the lower part of the dialog window under *Active Study Case*. Its dialog can be opened by pressing the → button.
- Pressing the **Contents** button on the dialog will open a new data browser displaying all the 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 *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 *Migration* page provides information about the migration status of the project. The migration priority of the project is also set in this page, this priority is used when using the *Minimal* migration option described in section [5.4.4.1](#).

The basic project structure was changed from *PowerFactory* version 2020, the *Variations* folder and *Operation Scenarios* folder being moved to sit 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, for example. Instead, an option to “Migrate Variations and Operational Scenario folder” is provided on the *Migration* page.

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

## 8.1.2 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.

### 8.1.2.1 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.

### 8.1.2.2 Input

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

**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).

**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).

### 8.1.2.3 Advanced Calculation Parameters

On the *Advanced Calculation Parameters* page, additional parameters used during the calculation are defined.

**Base Apparent Power:** this is the value used during the calculation; the base power of each element is defined by its nominal value.

**Min. voltage for HV voltage level:** this describes the voltages threshold for network elements to be considered as high voltage elements. This information is used in various commands when defining acceptable load flow errors.

**Min. voltage for MV voltage level:** this describes the voltages 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.

**Minimum Resistance, Minimum 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.

**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 in the table above
  - **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 selected.

**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.

**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 control 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.

### 8.1.2.4 Single Line Graphics

#### Geographic data

- **Coordinate system:** the *geographic coordinate system* 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).
- **WGS-84 bounds:** these fields show the bounds of the selected system in WGS-84 coordinates.

#### When connecting component to busbar

- **Create Circuit-Breakers:** 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.
- **Create Load-Break-Switch or Circuit-Breaker dep. on nominal voltage:** if this option is selected instead, there is the option to have Load-Break switches for lower voltages, with the threshold customised by the user.

#### Variations

- **Show inactive elements from other variations:** by default, inactive elements (for example elements created in a variation stage which is not yet active) are shown on graphics which 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 48.4.6.

### 8.1.2.5 Miscellaneous

#### 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” option, earths will be applied.
- **Isolate opens circuit breakers only:** when equipment is isolated using a planned outage or “right-click, isolate”, 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 42), 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)”.

### 8.1.2.6 External Data

**External Data Directory:** this is a folder on the user's hard disk, in which data files related to the *PowerFactory* project, such as images or Python scripts, are stored. The filepath has to be absolute,

but can contain the placeholders shown below on the dialog box:

- “\$(InstallationDir)” for the installation directory of *PowerFactory*;
- “\$(WorkspaceDir)” for the workspace directory.

Once defined, the external Data Directory can be used to link files to *PowerFactory* objects by the following abbreviation:

- “\$(ExtDataDir)”

**Check existence of referenced files:** if this button is clicked, *PowerFactory* checks to see that all external files referenced in the project exist, and reports those not found.

**Display reminder after file selection:** for certain file reference parameters, it is expected that the file will be in the External Data area and, if this check box is enabled, a dialog box will come up if a different location is specified. Note that it is permitted to select files outside the External Data directory; the dialog is just a reminder that the External Data directory is recommended.

#### 8.1.2.7 Plots

**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).

#### 8.1.2.8 Data Verification

##### Nominal Voltage Check

**Max. allowed differences over Lines/Switches:** 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 (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 nominal voltage of a transformer or other element is not too low compared with the terminal to which it is attached. With differences above these thresholds, a load flow will fail with an error message; for smaller differences (but >10%) a warning is given.

#### 8.1.2.9 Substation Types

This list of substation types, configurable by the user, can be used in geographic diagrams to distinguish graphically between different types of substation 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.

### 8.1.3 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-sensitive 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 [8.1.1, Storage page](#).

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

### 8.1.4 Exporting and Importing Projects

Projects (or any folder in the database) can be exported using the \*.pdf (*PowerFactory Data*) file format, or by exception for older applications by using the \*.dz format. It is recommended to use the PFD format (\*.pdf) whenever possible when exporting projects: the consumption of memory resources is significantly lower than with the old file format (\*.dz) and functions such as historic data, time stamps and former versions are not supported by the old \*.dz format.

A new project export method, the Snapshot Export, has been made available from Version 2017. This method, described in section [8.1.4.2](#), creates a file in a \*.dzs format.

#### 8.1.4.1 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-sensitive menu (only available for inactive projects). The dialog is shown in Figure [8.1.2](#).

- **Objects to export:** this table shows all top-level objects that will be exported within the \*.pdf file.
- **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 master projects, the corresponding Version for the derived project must be selected. See Section [21.3.1](#) for further details.
- **Export data in retention period:** if enabled, data changes from within the retention period will be exported. See Section [8.1.1](#) 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 an 'adequate' project. In this case no master project is required. It should be noted that this project can no longer be reused as a derived project.
- **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.
- **PFD file:** the path where the \*.pdf file will be saved.

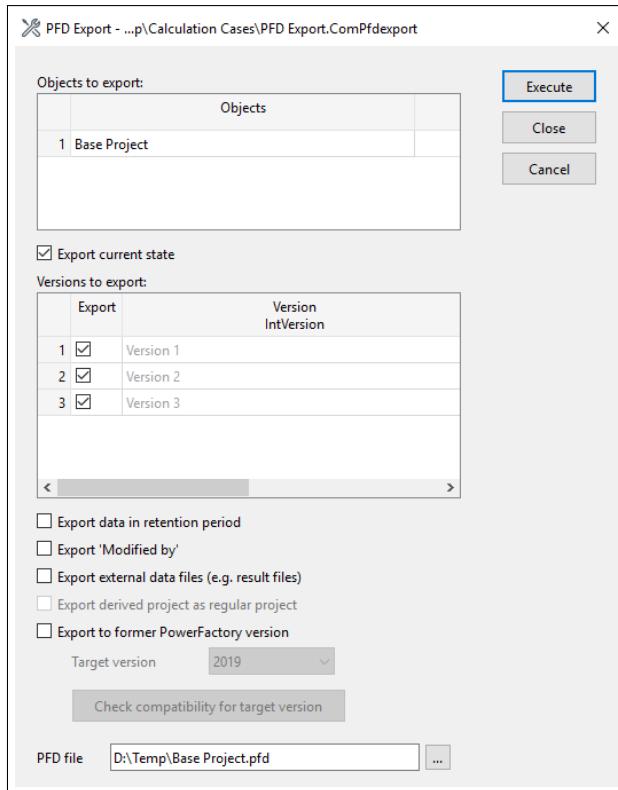


Figure 8.1.2: Export dialog

#### 8.1.4.2 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 .pfd or .dz 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.

#### 8.1.4.3 Importing a Project

Projects can be imported by selecting *File* → *Import...* → *Data...* from the main menu or by clicking on the  icon in the Data Manager. The user can select the type of file to import from the *Files of type* field in the Open dialog which pops up. Alternatively, projects can be imported by selecting *Import...* on the project context-sensitive menu (only available for inactive projects).

The import and export of information in other data formats is described in Chapter [24](#).

#### 8.1.5 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 by selecting the option *Check for external References* via the project context-sensitive menu.

If external references are found, these can be packed before exporting by selecting the option *Pack external References* in the project context-sensitive menu.

The user can define the source of the External References (i.e. Global Library, Configuration folder, etc). A new folder, called “External” containing all external references will be created inside the project.

#### 8.1.6 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 “Other Elements (Int\*)” and then entering “IntDocument” in the Element field.

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.

## 8.2 Creating New Grids

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

1. Open the edit dialog of the project and press the **New Grid** button.
2. Select *Insert* → *Grid...* on the main menu.

3. Right-click the Network Data folder (in the active project) in a Data Manager window and select *New → Grid* from the context-sensitive menu.

The dialog to create a new grid will then pop up. There the grid name, the nominal frequency and a grid owner (optional) may be specified. A second dialog will appear after the **Ok** button has been pressed. In this dialog the study case that the grid will be linked to must be selected. Three options are given:

1. **add this Grid/System Stage to active Study Case:** only available when a study case is active.
2. **activate a new Study Case and add this Grid/System Stage:** creates and activates a new study case for the new grid.
3. **activate an existing Study Case and add this Grid/System Stage:** add the new grid folder to an existing, but not yet active study case.

After the **Ok** button in the second dialog has been pressed, the new grid is created in the Network Data folder and a reference in the Summary Grid object of the selected study case is created. Normally, the second option (from the list above) is preferred because this creates a new study case which is dedicated to the new grid only. This means that the new grid may be tested separately using a load flow or other calculation. To analyse the combination of two or more grids, new study cases may be created later, or the existing ones may be altered.

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

## 8.3 Project Overview

The Project Overview is illustrated in Figure 8.3.1. It is a dockable window, 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 facilitates easy interaction with the project data. The window is docked by default, but can be undocked by the user and displayed as a floating window that can be placed both inside and outside of the main application window.

To dock an undocked Project Overview window, it should be moved all the way to the left border of the main window, until the mouse reaches the border.

If required, the window can be closed by the user. To close or reopen the window, toggle the option *Window → Project Overview* from the main menu.

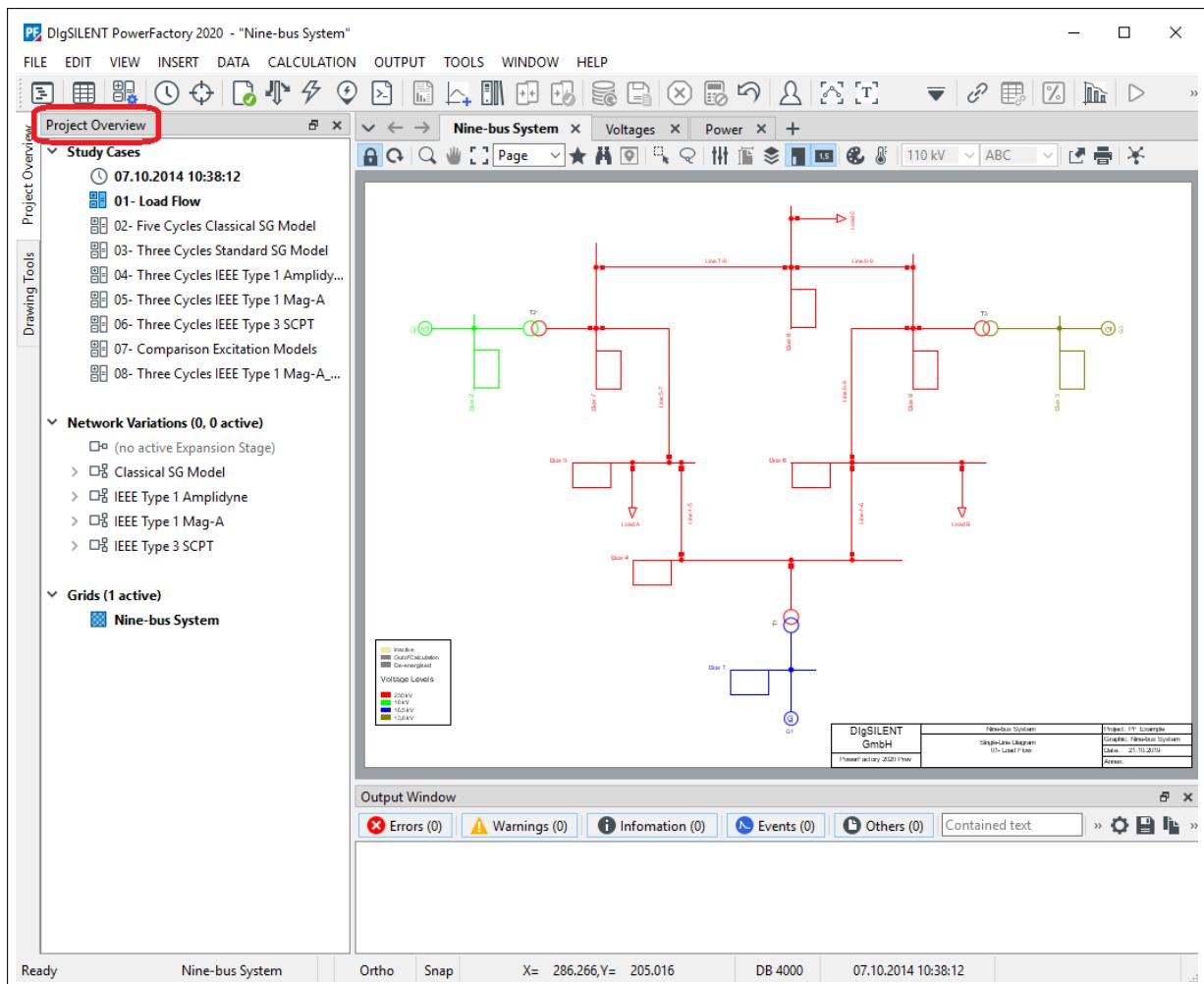


Figure 8.3.1: Project overview

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 or System Stages
  - List of inactive Grids or System Stages
- Trigger

- Active triggers

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

A context-sensitive menu can be accessed by right-clicking on each of the tree entries. The following actions are available for each of the entries:

- Change active item(s): Activate, Deactivate, change active
- Show all available items
- Edit (open dialog)
- Edit and Browse
- Delete
- Save (for Operation Scenario only)

# Chapter 9

# Network Graphics

## 9.1 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 (described in Section 30.2: Dynamic Models)
- Plots (described in Section 19.7: Plots)

Diagrams are organised in Graphic Boards for visualisation (see Section 9.2.2 for more information).

## 9.2 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.

### 9.2.1 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 logic (controller) circuits and relays.
4. Plot Pages for designing graphs, 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 controller types (common and composite types 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 *Show Graphic*. The *Show Graphic* 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.

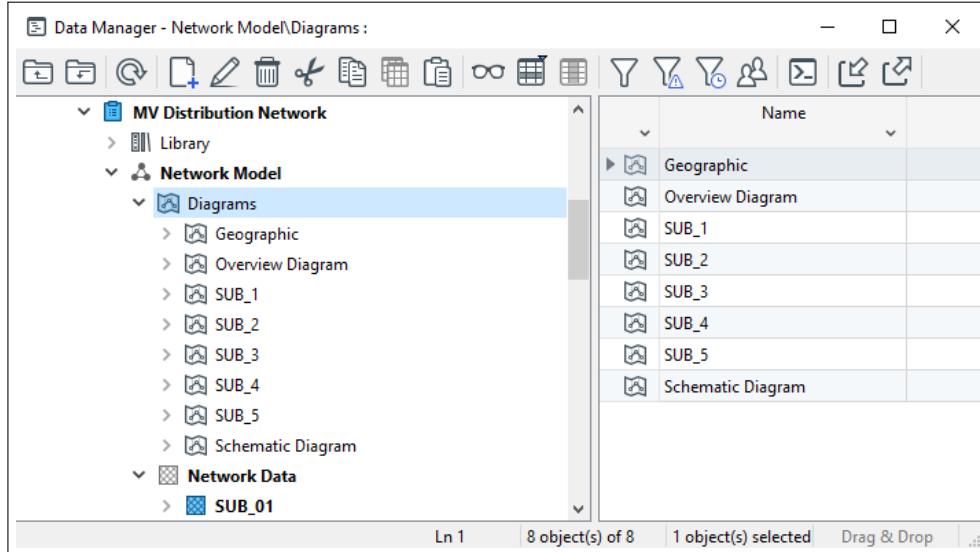


Figure 9.2.1: The Diagrams folder inside the Data Manager

## 9.2.2 Active Graphics, Graphics Board and Study Cases

The graphics that are displayed in an active project are determined by the active study case. The study case folder contains a folder called the *Graphics Board* folder (*SetDesktop*) in which references to the graphics to be displayed are contained. The Graphics Board folder is automatically created and maintained and should generally not be edited by the user.

Consider the project shown in figure 9.2.2. There are several diagrams in the *Diagrams* folder, but the graphics board 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 graphics board. See section 9.2.5 for details.

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

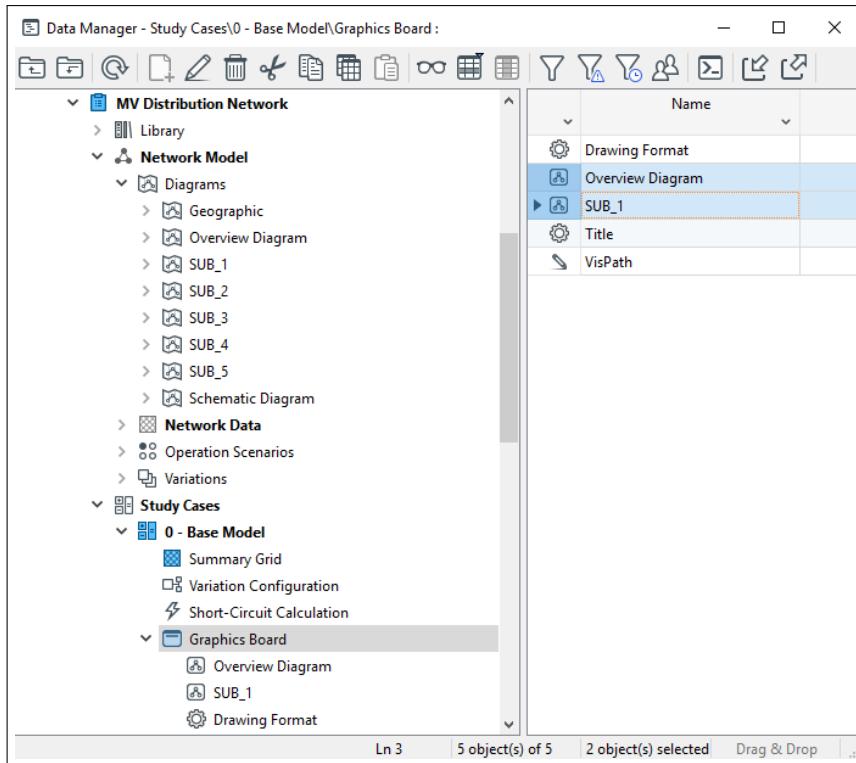


Figure 9.2.2: Relationship between the study case, graphics board and single line diagrams

### 9.2.3 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 11.

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 Single Line graphic 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 Single Line 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 right-clicking the graphic → *Graphic Options* or using the *Graphic Options* 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 menu → *Delete Graphical 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 Element*) a warning message will confirm the action. This warning may be switched off in the User Settings dialog, General page, *Always confirm deletion of Grid Data*.

### 9.2.4 Creating New Graphic Windows

A new graphic window can be created by using *Insert* on the main menu. Four options relating to graphics are offered:

- **Single Line Diagram:** Creates a Single Line Diagram graphic in the target folder. Before the graphic is inserted, the user is prompted to select the relevant grid.
- **Geographic Diagram:** Creates a Geographic Diagram of the network.
- **Block / Frame Diagram:** Creates a new (empty) Block Diagram Graphic object, which is then displayed.
- **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.

Graphic pages can only be shown as pages in a so-called graphics board, so this will automatically be created if required. Creating a new page while in a graphics board can be done by clicking on the  icon on the graphics board toolbar.

### 9.2.5 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, and new graphics can be added using the  icon.

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 whether the screen is split horizontally or vertically, and also on how many tabs are already open in a group. The options offered to the user also depend on the type of graphic, but this is the complete list of 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 rightward/downward to new group*: For split-screen working, start a new group (See section 4.7 for more details)
- *Rename page...* displays a dialog to rename the graphic.
- *Close page* closes the graphic page, removing it from the graphic board 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.

### 9.2.6 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 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 9.2.3 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.

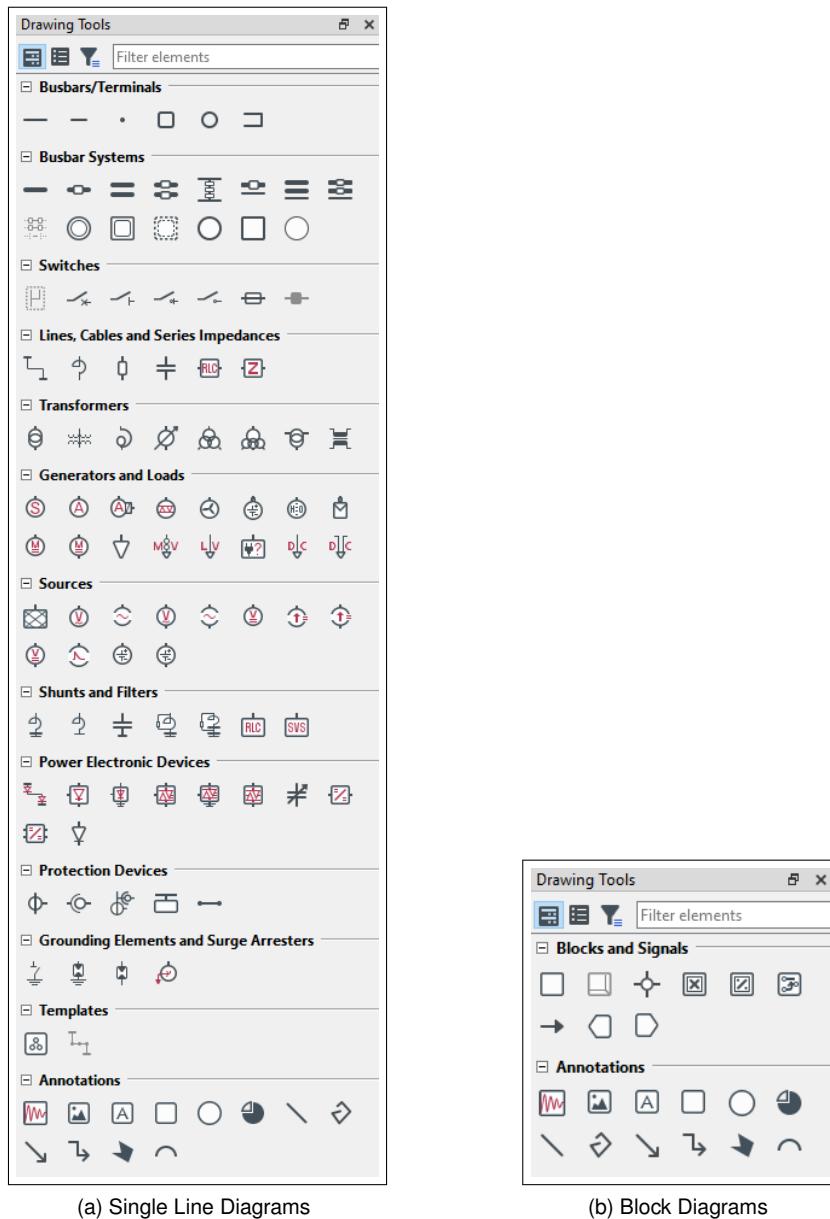


Figure 9.2.3: Drawing Toolbox examples

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.

#### 9.2.6.1 Annotations

As well as tools for adding elements to a graphic, the drawing toolbox also includes a number of annotation options. By default, these are placed into a separate Annotation layer. See section 9.6 for more information on annotations.

#### 9.2.7 Active Grid Folder (Target Folder)

On the status bar of *PowerFactory* (figure 9.2.4), 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 9.2.4: The Status Bar

## 9.3 Graphic Commands, Options, and Settings

In this section the commands, options and settings that are available in *PowerFactory* to configure and use the graphic windows are introduced. The sub-sections of this chapter are divided as illustrated in figure 9.3.1. These commands are also available from the main menu under *View*.

Further commands available from the context-sensitive menus of elements are also listed towards the end of this section (9.3.13).

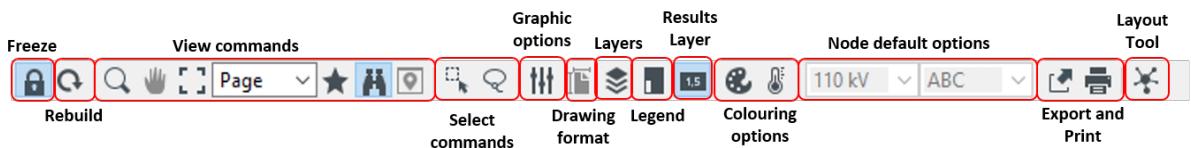


Figure 9.3.1: Categories of graphic commands, options, and settings

### 9.3.1 Freeze Mode

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.

### 9.3.2 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.

### 9.3.3 View commands

#### 9.3.3.1 Zoom In

Press the *Zoom In*  to change the cursor 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.

---

#### 9.3.3.2 Zoom All

 zooms to the page extends.

#### 9.3.3.3 Zoom Level

Zooms to a custom or pre-defined level.

#### 9.3.3.4 Hand Tool

Use the hand tool  to pan the single line diagram (when not at the page extends). The hand 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.

#### 9.3.3.5 View Bookmarks

The *View Bookmarks*  allow 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.

### 9.3.3.6 Search in Diagram

The  icon is used to search for elements in the graphic, as described in section 9.9 below.

### 9.3.3.7 Navigation Pane

The  icon is used to open or close the navigation pane, described in section 9.8 below.

## 9.3.4 Select commands

### 9.3.4.1 Rectangular Selection

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

### 9.3.4.2 Free-form Selection

 is used to select a custom area of the diagram.

### 9.3.4.3 Mark All Elements

 marks (selects) all objects in the current diagram. This is helpful for moving the whole drawing to another place or copying the whole drawing into the clipboard. In block diagrams the surrounding block will not be marked. The keyboard short cut **Ctrl+A** may also be used to perform this action.

---

**Note:** If the user finds that not all required graphic objects are selected, particularly for example annotations, check the relevant graphic layer via the Layers dialog: it could be that the *Elements are selectable* option has been de-selected. See section 9.3.6 for more about working with layers.

---

## 9.3.5 Graphic Options

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

### 9.3.5.1 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 9.3.3.5 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 the line → *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:** by default, changes of the active Colouring Scheme take effect on every diagram (Default). 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 or create a new one and alter it to the wished scheme. Close the object browser and choose the new colouring scheme out of the drop down list.

The Advanced-tab offers the following settings:

- **Allow navigation pane to be shown:** if checked, the *Navigation Pane* can be activated by *Window* → *Navigation Pane* or the context sensitive menu in the diagram *Navigation Pane*.
- **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 *Additional Attributes* and *Coordinates Space* pages should generally only be configured with the assistance of *DlgSILENT* 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 Graphic by right-clicking and selecting *Measure Length of Lines*. In geographic diagrams, this option is activated by default.

### 9.3.5.2 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 Tools

- **Snap:** snaps the mouse onto the drawing raster.
- **Grid:** shows the drawing raster using small points.
- **Ortho-Type:** defines if and how non-orthogonal lines are permitted:

- **Ortho Off:** connections will be drawn exactly as their line points were set.
- **Ortho:** allows only right-angle connections between objects.
- **Semi Ortho:** the first segment of a connection that leads away from a busbar or terminal will always be drawn orthogonally.

#### 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.

---

#### 9.3.5.3 Text Boxes page

- **Object names**
  - **Show frame for nodes/branches:** the text boxes of the according elements will show a frame.
  - **Background:** specifies the transparency of object names 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.
- **Results**
  - **Show frame for nodes/branches:** the result boxes of the according elements will show a frame.
  - **Background:** specifies the transparency of result boxes (as object names)
  - **Always show result boxes of detailed couplers:** self-explanatory.
  - **Space saving representation of result boxes on connection lines:** self-explanatory.
- **Show line from general text boxes to referenced objects:** may be disabled to unclutter the graphic.
- **Reset text boxes completely:** textboxes and result boxes have reference points (the point on the box at which the box will 'attach' to its element) that may be changed by the user. If this option is:
  - Enabled: the default reference will be used.
  - Disabled: the user defined reference will be used.
- **Show tooltip on network elements:** enables or disables the balloon help dialogs.

#### 9.3.5.4 Switches page

- **Switch state symbol at connection end:** selects the switch representation (see figure [9.3.2](#)):
  - **Permanent box:** shows a solid black square for a closed and an frame line for an open switch (left picture).
  - **Old style switch:** shows the switches as the more conventional switch symbol (right picture).

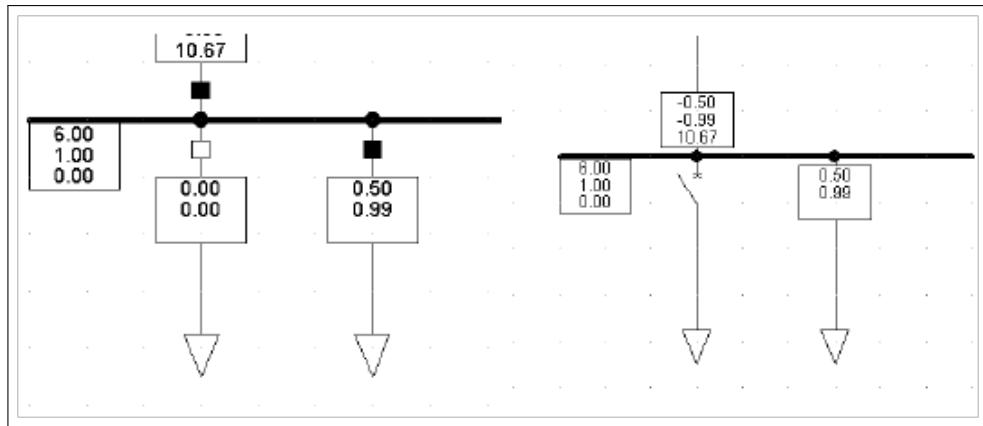


Figure 9.3.2: Cubicle representations

- **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.

#### 9.3.5.5 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.
- **Scale level threshold for visibility of:** in extensive networks with a high scale level, edge elements (except lines), switch state boxes at line ends, text labels and annotation objects are hidden at a specified scale level to improve the clarity of the diagram.
- **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 (9.3.5.1).
- **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.
- **GPS projection:** defines the projection used in the geographic diagram. This is only active when local maps are selected.

### 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.

### 9.3.6 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 *Graphical Layers* dialog, accessed through the graphic toolbar ((toolbar icon)), by right-clicking on an empty spot of the graphic area → *Layers*, or selecting *View → Layers* from the main menu.

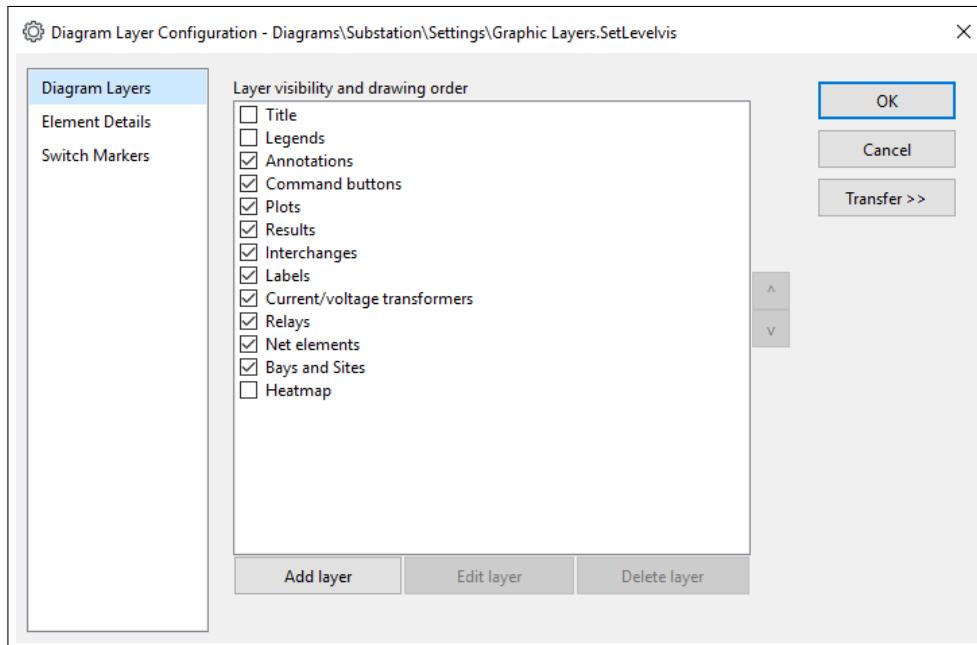


Figure 9.3.3: Graphical layers dialog (*SetLevelvis*)

As shown in Figure 9.3.3, the Graphic Layers 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.

**Note:** Prior to *PowerFactory 2019*, all annotation graphical objects were held in a dedicated Annotation Layer. Now, 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 9.6 for more details.

### 9.3.6.1 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 9.3.3 above). 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	Text/Box Format	SL/GEO/B
<b>Legends</b>	Results boxes legend and colour legend	Text/Box Format	SL/GEO
<b>Net elements</b>	Symbols for the elements of the grid	Text/Box Format	SL/GEO/B
<b>Labels</b>	Boxes with names and additional data description, if configured	Text/Box Format	SL/GEO/B
<b>Results</b>	Boxes with calculation results. See <b>Note</b> below	Text/Box 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>Bays and Sites</b>	Bay representation and Site frames	Line colour, width and style	SL
<b>Device data</b>	Additional Text explanation given in the device symbol	Text/Box Format	SL/GEO/B
<b>Background image</b>	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, DXF, BMP, JPEG, PNG, GIF, TIF)	SL/B
<b>Geographic map</b>	Geographical map used as the background to allow easier drawing of the diagram	Map Provider (see section 9.10), map type and graphic settings	GEO
<b>Load/generation distribution</b>	Shows circles for load and generation around substations	Selection of S, P or Q, colour-settings	GEO
<b>Additional text</b>	Additional text labels for elements	Text/Box Format	SL/GEO

Layer	Content	Configuration Options	Diagram Type: SL Single Line, GEO Geographic, B Block
<b>Relays</b>	Graphically represented relays	Text/Box Format	SL/GEO
<b>CTs and VTs</b>	Contains the drawn current and voltage transformers	Text/Box Format	SL/GEO
<b>Plots</b>	Plots placed in the diagrams	Text/Box Format	SL/GEO/B
<b>Commands buttons</b>	Commands buttons used mostly for DPL scripts	None	SL/GEO
<b>Block Definition</b>	Definition each block is based on	Text/Box Format	B
<b>Connection numbers</b>	Index of each possible block connection point	Text/Box Format	B
<b>Connection names</b>	Name of each unused connection of a block	Text/Box Format	B
<b>Signals</b>	Name of the signal transmitted	Text/Box Format	B

Table 9.3.1: Predefined diagram layers in the layers dialog

**Note:** The Results layer can be easily toggled on and off without going into the Layers command by using the  icon on the graphics toolbar.

### 9.3.6.2 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.

Layer	Content	Configuration Options	Diagram Type: SL Single Line, GEO Geographic, B Block
<b>Connection points</b>	Dots at the connections between edges and buses/terminals and signal connections to blocks	Text/Box Format	SL/GEO/B
<b>Connection Arrows</b>	Double-Arrow at connections where the end point is not represented in the current diagram.	Text/Box Format	SL/GEO
<b>Phases</b>	Number of phases of a line/cable, shown as parallel lines	Text/Box Format	SL/GEO
<b>Sections and Line Loads</b>	Symbols at lines consisting of sections and/or where line loads are connected	Text/Box Format	SL/GEO
<b>Vector groups</b>	Vector group for rotating machines and transformers	Text/Box Format	SL/GEO
<b>Tap positions</b>	Positions of taps for shunts and transformers	Text/Box Format	SL/GEO

Layer	Content	Configuration Options	Diagram Type: SL Single Line, GEO Geographic, B Block
<b>Power Flow direction arrows</b>	Arrows that can be configured for active and reactive power flow representation	Active/Reactive Power for direct/inverse/ homopolar system	SL/GEO

Table 9.3.2: Element Details in the Layers dialog

### 9.3.6.3 Switch Markers

On the Switch Markers page, markers specific to switch operation or status can be selected and configured:

<b>Remotely controlled substations</b>	Remote Controlled Substations	Colour	SL/GEO
<b>Tie open points</b>	Tie open points in the network	Colour/Size/Width	SL/GEO
<b>Normally open switches</b>	Normally open switches (Reliability Analysis)	Colour/Size/Width	SL/GEO
<b>Open standby switches</b>	Indicating lines energised but open at one end	Colour/Size/Width	SL/GEO

Table 9.3.3: Switch Markers in the Layers dialog

### 9.3.6.4 Working with 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 sensitive 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 [9.6](#).

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 9.3.1 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.

### 9.3.7 Colouring Options

#### 9.3.7.1 Diagram Colouring

The single line and geographic diagrams have an automatic colour representation mode. The *Diagram Colouring* icon  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.

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
  - 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
  - Probabilistic Analysis
  - 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
  - Meteo Stations
  - Operators
  - Owners
  - Paths
  - Routes
  - Zones
- Variations / System Stages
  - Modifications in Recording Expansion Stage
  - Modifications in Variations / System Stages
  - 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.

### 9.3.7.2 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 9.3.7.1.

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 9.3.7.1) 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 9.3.4 shows an example of a Heatmap, which is coloured according to loading, over- and under-voltage.

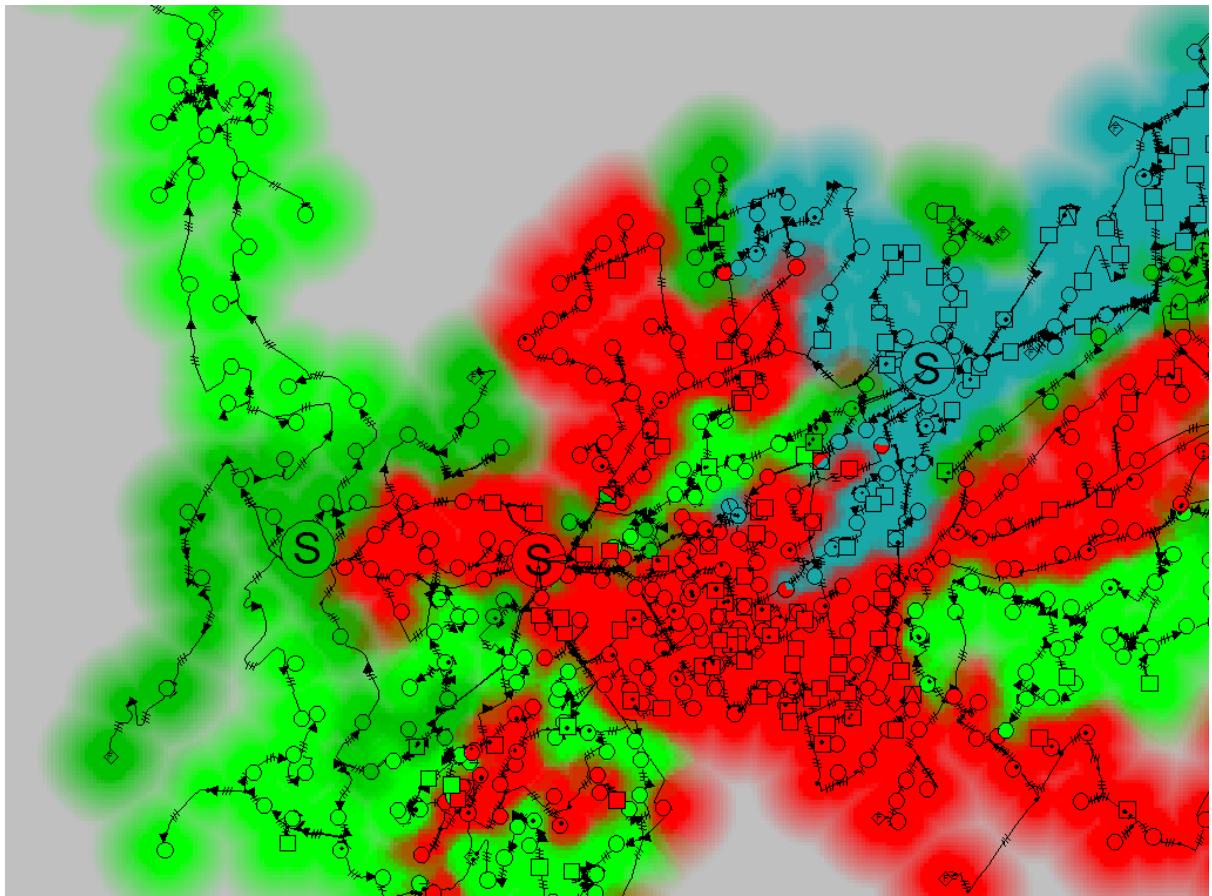


Figure 9.3.4: Example of a heatmap coloured according to loading, over- and under-voltage

## 9.3.8 Graphic Legends

### 9.3.8.1 Show Title Block

The title block can be turned on and off from the Layers dialog (see [9.3.6](#)). The title block is placed in the lower right corner of the drawing area by default.

The contents and size of the title mask can be changed by right-clicking the title block and selecting the *Edit Data* option from the context sensitive menu. The Select Title dialog that pops up is used to scale the size of the title block by setting the size of the block in percent of the default size. The font used will be scaled accordingly. To edit the text in the title block press the edit button ( $\rightarrow$ ) for the 'Title Text' field.

All text fields have a fixed format in the title block. The data and time fields may be chosen as automatic or user defined. Most text fields are limited to a certain number of characters. When opening a new graphic the title will appear by default.

### 9.3.8.2 Show Legend Blocks

The legend blocks can be turned on and off from the Layers dialog (see [9.3.6](#)), or from the single line diagram toolbar ().

The results legend block describes the contents of result boxes (for information about result boxes see [9.5](#)).

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 Data* 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.

### 9.3.9 Node Default Options

Figure 9.3.5 shows the commands available for setting node default options. These are discussed in further detail in this section.

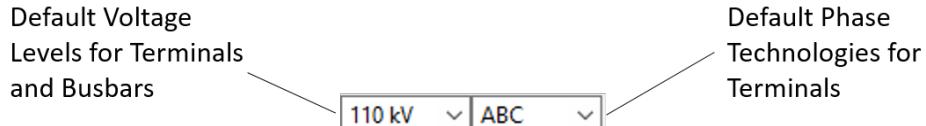


Figure 9.3.5: Node default options

#### **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.).

### 9.3.10 Page, Print and Export Options

#### 9.3.10.1 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).

#### 9.3.10.2 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.

### 9.3.10.3 Exporting Graphics

Graphics can be exported in a wide range of formats, using the *Export Diagram* function. This is accessed either from the main toolbar, using *File* → *Export Diagram...*, or by using the icon on the graphic 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 user to select graphic pages to be exported.

### 9.3.11 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 [11.6](#).

### 9.3.12 Insert New Graphic

Pressing the button opens the *New* command dialog described in section [9.2.4](#).

---

**Note:** The Page Tab menu is opened by right-clicking a page tab, shown just below the single line diagram. Existing graphics can be opened by selecting *Show Graphic* of the context sensitive menu of the graphic object in the subfolder Network Model/Diagrams or by choosing it from the list, which opens after selecting *Insert Page* → *Open Existing Page* from the context sensitive menu of the page tab.

---

#### 9.3.12.1 Other Page Commands

Other page commands accessed through the page tab are as follows:

**Remove Page:** this function will remove the selected graphic from the Graphics Board. The graphic itself will not be deleted and can be re-inserted to the current or any other Graphics Board at any time.

**Rename Page:** this function can be used to change the name of the selected graphic.

**Move/Copy Page(s):** this function can be used to move a page/s to modify the order of graphics. Also accessed through:

- Mouse Click: Left-click and select a single page (optionally press control and select multiple pages) and drag the page/s to change the order graphics are displayed.
- Data Manager: (Advanced) Modify the order field of Graphics Pages listed within the Study Case Graphics Board. To reflect the changes, the study case should be deactivated and then re-activated.

### 9.3.13 Element Options

In addition to the options available from the graphics toolbar, there are many edit options which are available for elements in the graphic, using the context-sensitive menu (i.e. right-click).

**Edit and Browse Data:** this option lets the user edit the data of the selected object. The object will be selected in its project folder within the list of the other objects and can be double-clicked to open its edit dialog. See chapter 10 (Data Manager) for more information.

**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.

---

**Note:** To delete graphical objects only, right click the selected element/s and select *Delete Graphical Object only*.

---

**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 Graphic Only* should be chosen from the right-click menu. Similar results are obtained when using the “Draw Existing Net Elements” tool (see Section 11.6: Drawing Diagrams with Existing Network Elements).

---

**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 (undo), by right-clicking and selecting ‘Undo’, or by pressing **Ctrl+Z**.

---

**Rotate:** rotates symbols clockwise, counter-clockwise, or 180 degrees. It is generally preferable to disconnect an element before rotating it.

**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*)

**Connect:** right-click and select *Connect Element* 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*)

**Redraw:** right-click and select *Redraw Element* to redraw a selected element.

**Move:** 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 (move). 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.

**Edit Line Points:** right-click and select *Edit Line Points* will show the black squares ('line points') that define the shape of the connection. Each of these squares can be moved by left clicking and dragging them to a new position (see figure 9.3.6). New squares can be inserted by left clicking the connection in between squares. Line points are deleted by right-clicking them and selecting the *Delete Vertex* option from the case sensitive 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.

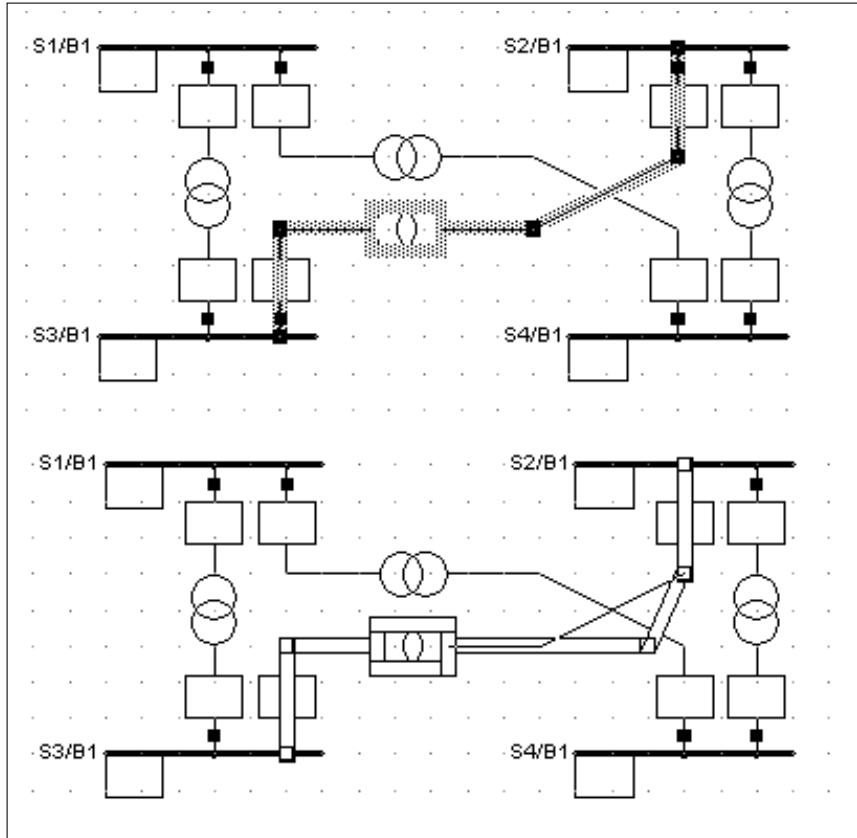


Figure 9.3.6: Editing line points

## 9.4 Editing and Changing Symbols of Elements

You can edit or change the symbols, which are used to represent the elements in the single line graphic. Right click with on a symbol of an element in the single line graphic, and select *Change Symbol* from the context sensitive menu in order to use a different symbol for the element.

*PowerFactory* supports user-defined symbols as Windows-Metafile (\*.wmf) and Bitmap (\*.bmp) files.

For additional information refer to Appendix E (Element Symbol Definition).

## 9.5 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 9.5.1 illustrates an example of this.

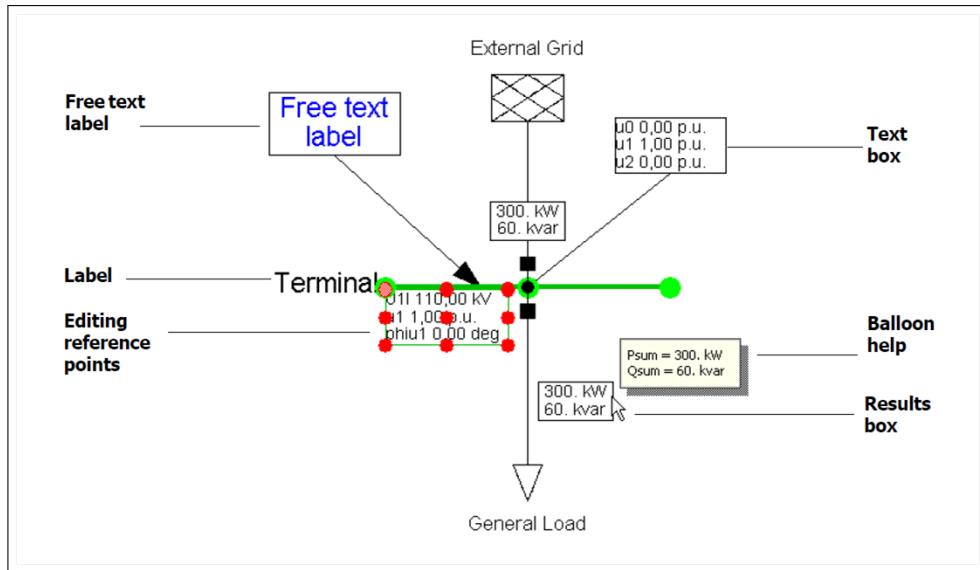


Figure 9.5.1: Results boxes, text boxes, and labels available in *PowerFactory*

### 9.5.1 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 9.5.1. 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 9.5.1 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 reference point, and drag it to a red 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.

If the option “Reset textboxes completely” is set in the graphical settings, then the default reference and docking points will be selected again, and the result box is moved back to the default position accordingly.

### 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 sensitive 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 [9.5.2](#).

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 sensitive menu (right-clicking the desired result box). The available options include:

- Shift to layer (see [9.3.6](#)).
- 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 [7.1](#): 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**.

### 9.5.2 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 [9.5.1](#). 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 Textboxes...* from the *Result Boxes* page of the Graphic Option dialog ([9.3.5](#)) is disabled.

### 9.5.3 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 [9.5.1](#)). 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 sensitive menu.

### 9.5.4 Free Text Labels

Free Text Labels (see figure [9.5.1](#)) 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 ‘Create Free Text Label’.

## 9.6 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.

## 9.7 Annotation of Protection Device

Adding a protection device into the single line diagram is described in section [33.2.2](#).

## 9.8 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. This is illustrated in figure [9.8.1](#).

The navigation pane can be turned on or off using the  icon in the graphics toolbar. The frame within the navigation pane can be moved around in order to see different parts of the network.

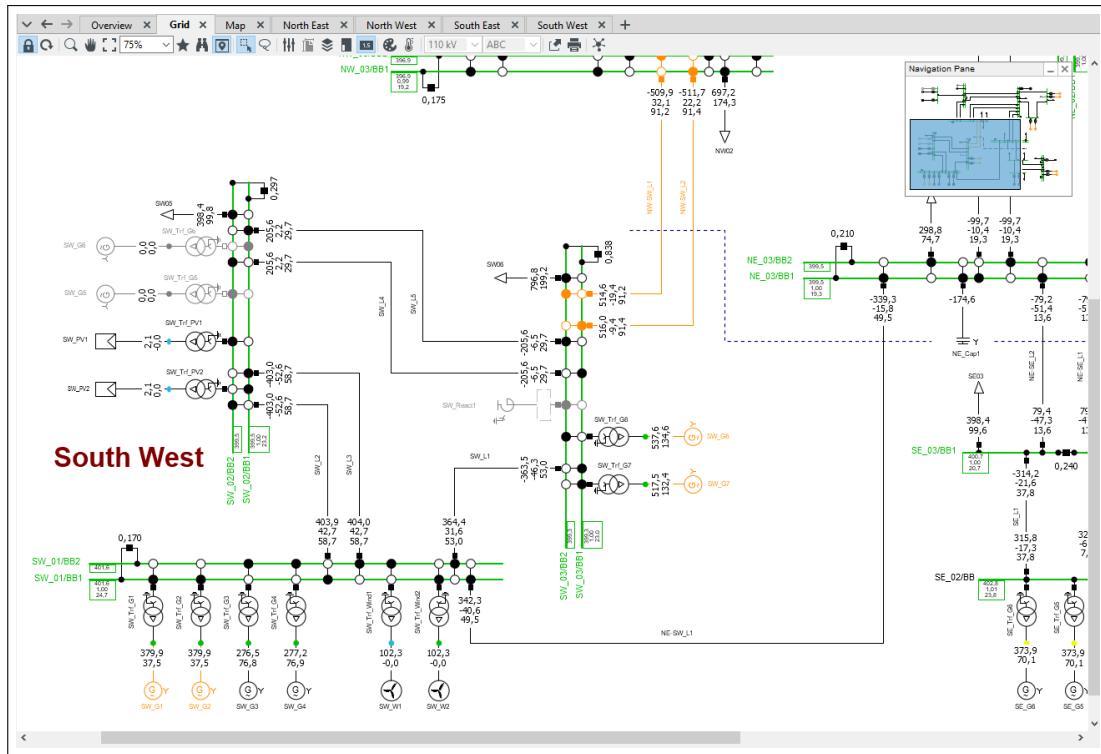


Figure 9.8.1: Navigation pane

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 *Graphic Options* icon ((toolbar icon)), then the dialog, go to the *Advanced* tab within the *Basic Attributes* and disable the option “Allow Navigation Pane”.

## 9.9 Graphic search facility

It is possible to search for network elements within a graphic, using the icon in the graphic toolbar. This is illustrated in figure 9.8.1 above: the search facility automatically lists possible objects as the user types.

On geographic diagrams (see section 9.10 below), 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.

## 9.10 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 the active project or active grid and selecting *Show Graphic* → *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 11.6.1.2).

The settings for the geographic diagram are defined in the Graphic Options, *Geographic Diagram* page (see section 9.3.5.5).

An additional layer called *Load/Generation Distribution* is available for GPS coordinates to illustrate the magnitude of network load and generation (apparent power), as illustrated in figure 9.10.1. Note that the displayed size of circles does not change as the user zooms in and out of the diagram. Colour and *Scaling Factor* settings can be modified on the *Configuration* page of *Layers* (Layers icon), as described in section 9.3.6 (Layers).

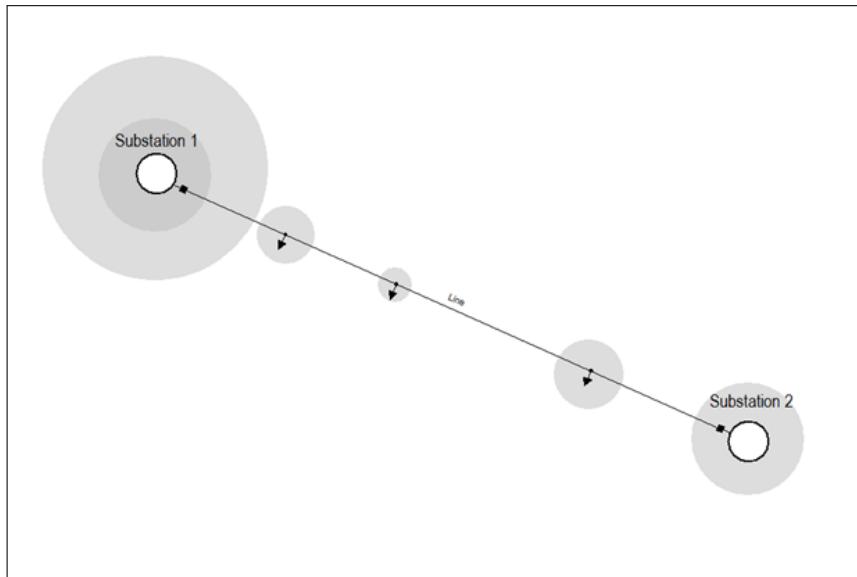


Figure 9.10.1: Load/Generation Distribution example

Maps can be used as background images and can be specified on the *Configuration* page of *Layers* (Layers icon). 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*.

<sup>1</sup>requires Google Maps for Business account: <http://www.google.com/enterprise/mapsearch>

### 9.10.1 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

These parameters are valid in the range -100 % and +100 % and can be used to highlight either the map or the network elements.

Figures 9.10.2 and 9.10.3 illustrate small distribution grids where OpenStreetMap, and Esri ArcGIS® satellite maps, respectively, are used as the background image providers.

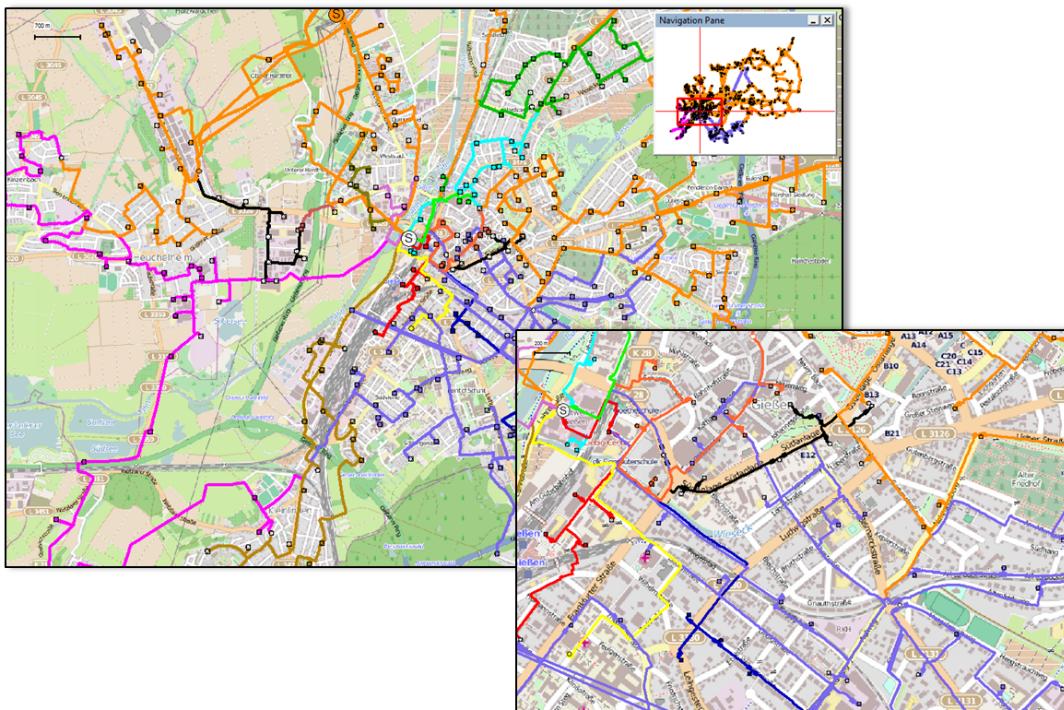


Figure 9.10.2: Network example with OpenStreetMap data

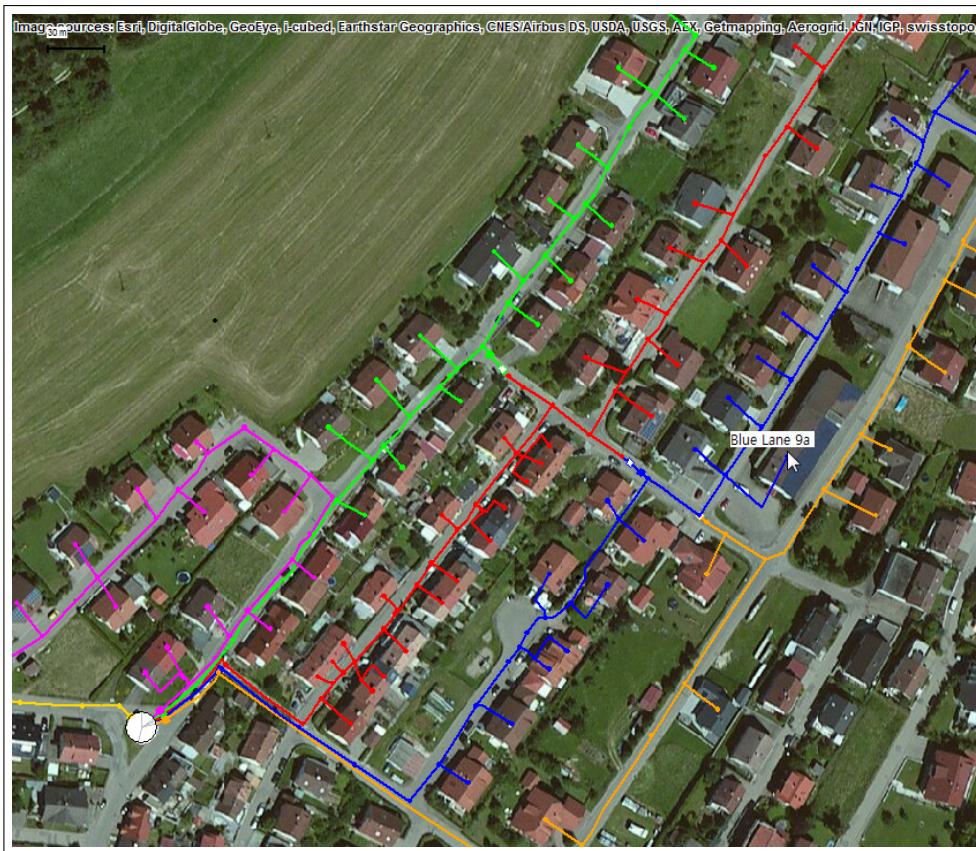


Figure 9.10.3: Network example with satellite background map (ESRI ArcGIS©)

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 9.10.4

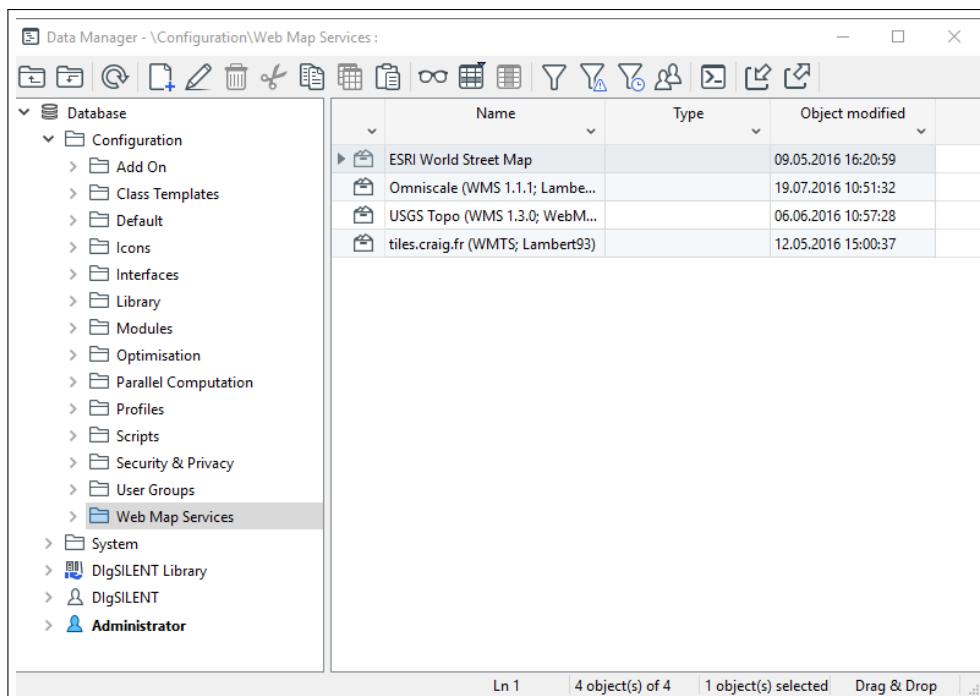


Figure 9.10.4: Web Map Services folder location

If the “Web Map Services” folder doesn’t exist, it can be created by right clicking on the *Configuration* folder and selecting *New → Folder*. The folder should be a *System (DlgsILENT)* folder with the key “MapServices” as shown in figure 9.10.5.

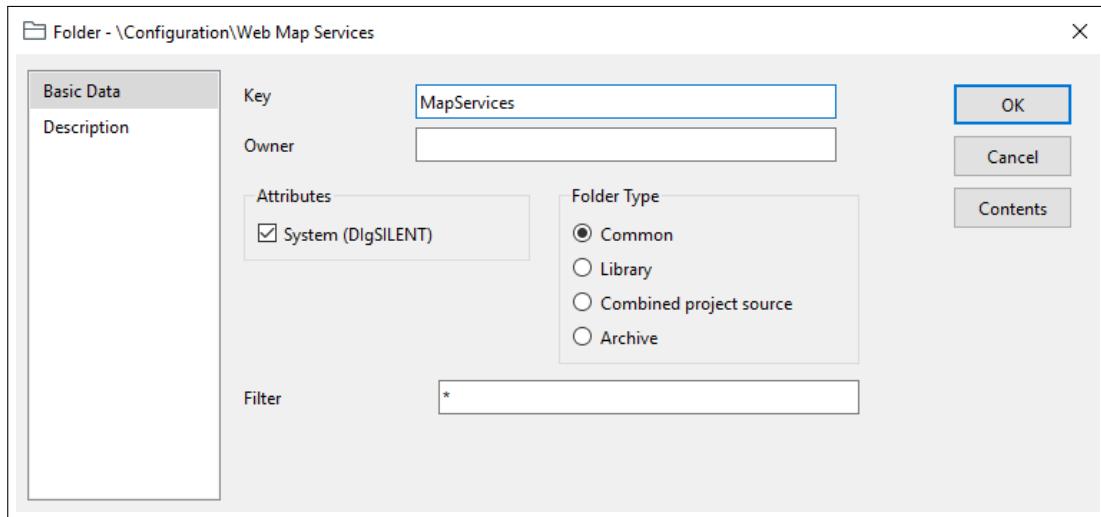


Figure 9.10.5: Web Map Services folder

A new Web Map Service can be created by clicking on the button *New* () in the Data Manager and selecting *Others → Other Elements (Int\*) → 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 9.10.6

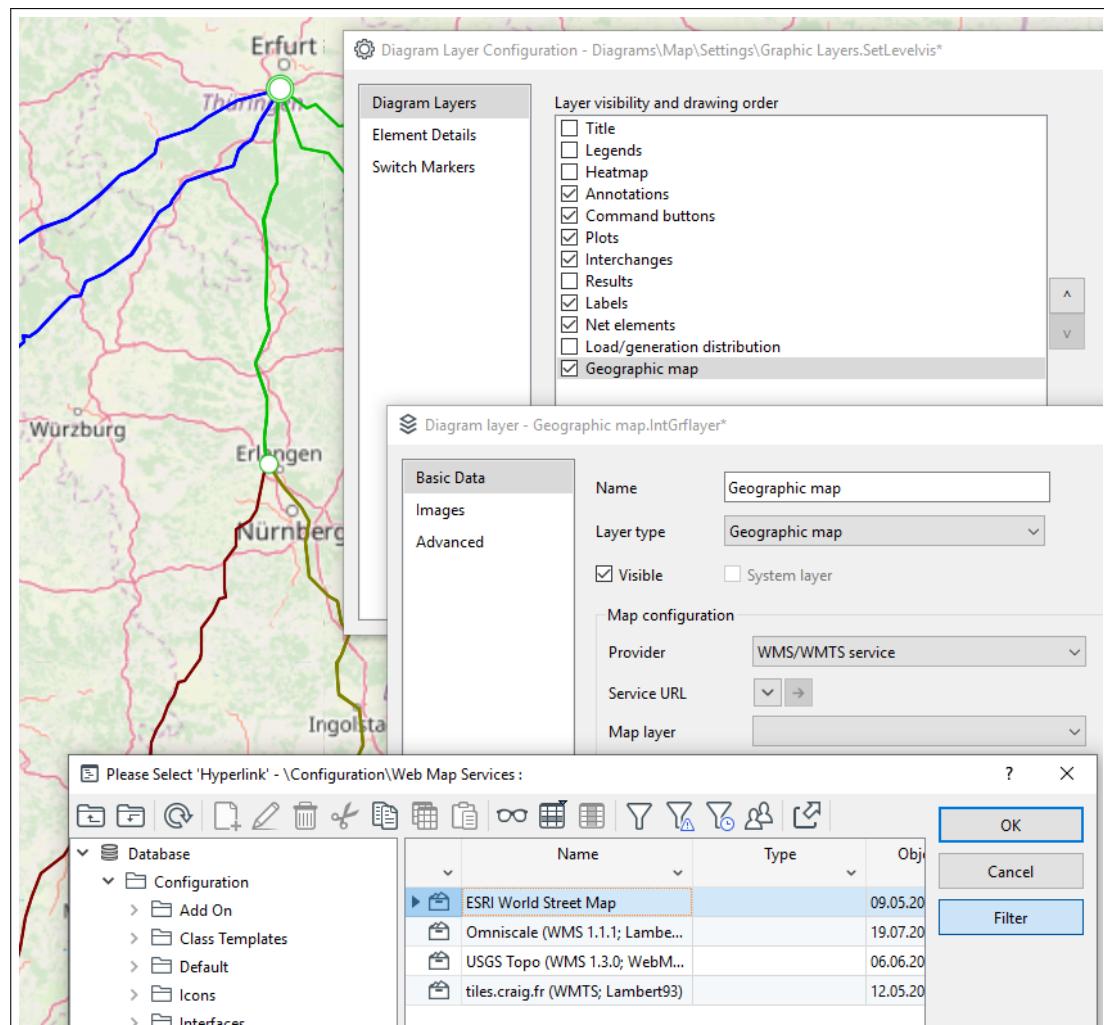


Figure 9.10.6: Background layer configuration

## 9.10.2 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.

# Chapter 10

## Data Manager

### 10.1 Introduction

To manage/ browse the data in *PowerFactory*, a Data Manager is provided. The objective of this chapter is to provide detailed information on how this Data Management tool. Before starting, users should ensure that they are familiar with Chapter 4 (*PowerFactory* Overview).

### 10.2 Using the Data Manager

The Data Manager provides the user with all the features required to manage and maintain all the data from the projects. It gives both an overview over the complete data base as well as detailed information about the parameters of single power system elements or other objects. New case studies can be defined, new elements can be added, system stages can be created, activated or deleted, parameters can be changed, copied, etc. All of these actions can be instituted and controlled from a single data base window.

The Data Manager uses a tree representation of the whole database, in combination with a versatile data browser. To initially open a Data Manager window press the  icon from the main toolbar. The settings of this window can be edited using the 'User Settings' dialog (Section 10.2.5: Data Manager Settings).

The Data Manager window has the following parts (see Figure 10.2.1):

- The title bar, which shows the name and path of the folder currently selected in the database [1].
- The Data Manager local tool bar [2].
- In the left upper area the database window, which shows a symbolic tree representation of the complete database [3].
- In the left lower area the input window. It may be used by more experienced users to enter commands directly, instead of using the interactive command buttons/dialogs. By default it is not shown. For further information see Section 10.7 (The Input Window in the Data Manager) [4].  
The input window is opened and closed by the clicking on the *Input Window* button ().
- On the right side is the database browser that shows the contents of the currently selected folder [5].
- Below the database browser and the input window is the status bar, which shows the current status and settings of the Data Manager (for further information see Section 10.2.5).

There are some special features of the database browser which can be accessed at any time when the content of a folder is shown:

- Balloon text: this is not only available for the buttons in the tool bar and the active parts of the status bar or the browser window, but also for the data fields [a].
- Active Title buttons of each column; click on any title button to sort the items in the column; first click- items are sorted in ascending order; second click - items are sorted in descending order [b].
- Object buttons showing the object standard icon in the first column of the database browser: each object is represented by a button (here a line object is shown). One click selects the object and a double-click presents the edit dialog for the object [c].

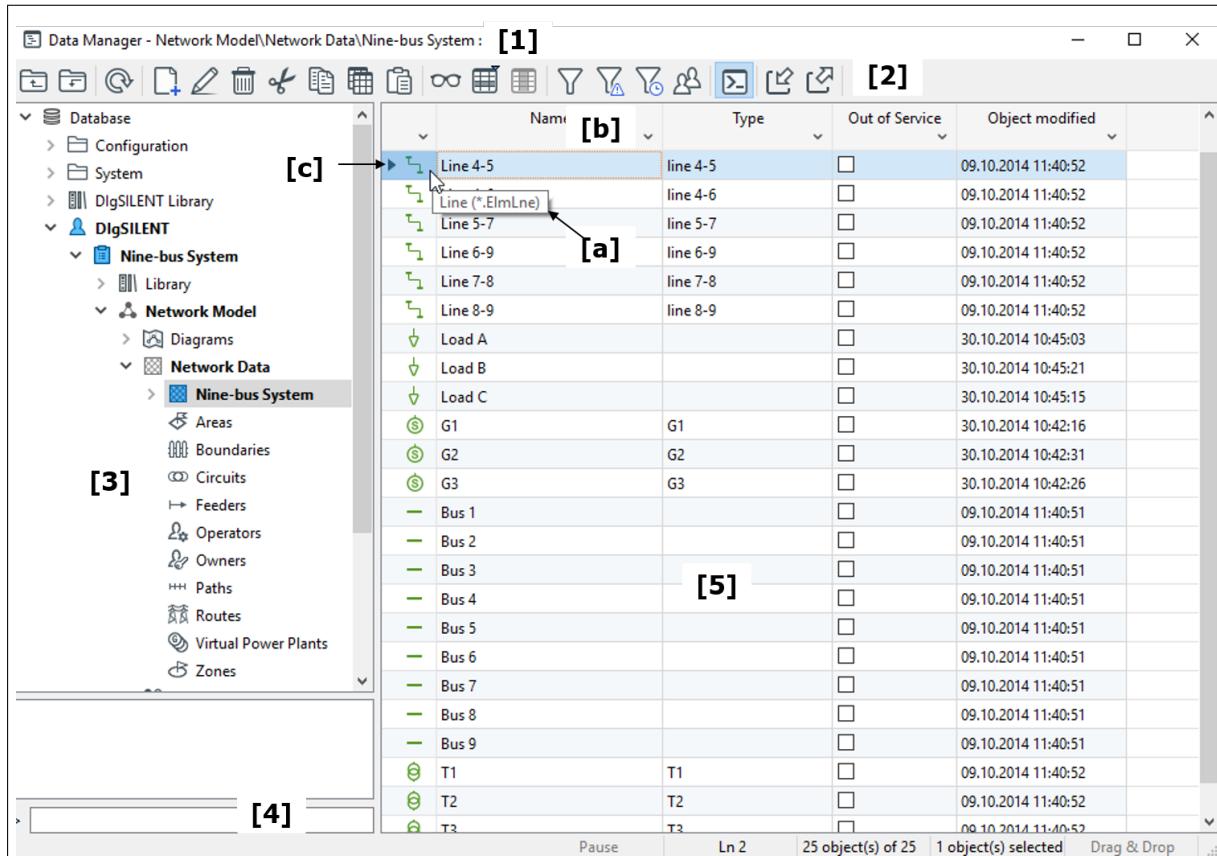


Figure 10.2.1: The Data Manager window

*PowerFactory* makes extensive use of the right mouse button. Each object or folder may be 'right-clicked' to pop up a context sensitive menu. For the same object the menu presented will differ depending on whether the object is selected in the left or right hand side of the Data Manager (this is known as a 'context sensitive' menu). Generally, the left hand side of the Data Manager will show object folders only. That is, objects that contain other objects inside them. The right hand side of the Data Manager shows object folders as well as individual objects.

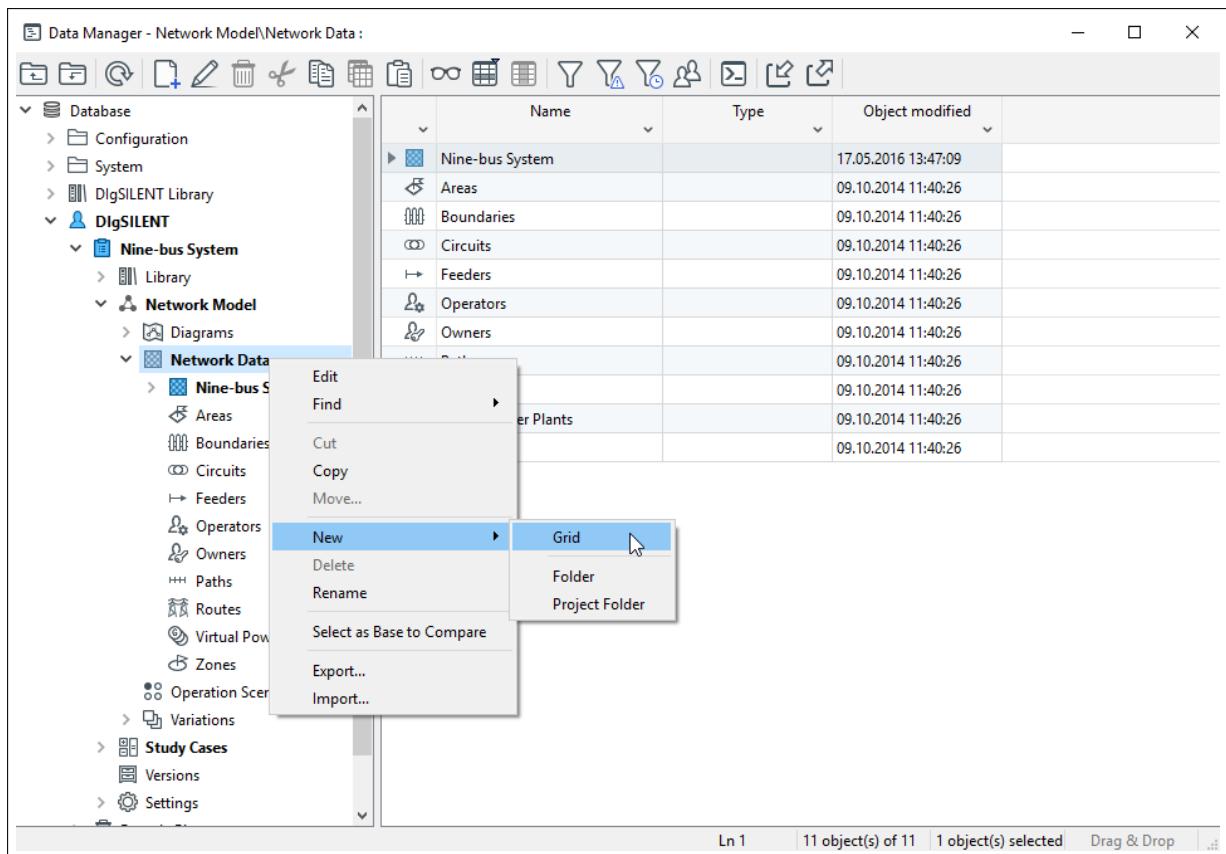


Figure 10.2.2: Context sensitive menus in the Data Manager

Using the right mouse button to access menus is usually the most effective means of accessing features or commands. Figure 10.2.2 shows an illustration of a context-sensitive right mouse button menu.

The symbolic tree representation of the complete database shown in the database window may not show all parts of the database. The user settings offer options for displaying hidden folders, or for displaying parts that represent complete stations. Set these options as required (Section 10.2.5: Data Manager Settings).

### 10.2.1 Navigating the Database Tree

There are several ways to “walk” up and down the database tree:

- Use the mouse: all folders that have a “+” sign next to them may be expanded by double-clicking on the folder, or by single clicking the “+” sign.
- 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 (see “c” in Figure 10.2.1) 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 *Up Level* (📁) and *Down Level* (📁) on the Data Manager tool bar can be used to move up and down the database tree.

## 10.2.2 Adding New Items

Generally, new network components are added to the database via the graphical user interface (see Section 11.2: Defining Network Models with the Graphical Editor), such as when a line is drawn between two nodes creating, not only the graphical object on the graphics board, 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 sensitive 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 Graphic, 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 commanded 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  icon must be pressed (*new object* icon). This is found the Data Manager toolbar and presents the dialog shown in Figure 10.2.3.

To simplify the selection of the new objects, a filter is used to sort the object list. This filter determines what sort of list will appear in the drop-down list of the ‘Element’ field. If “Branch Net Elements” is first selected, the selection of, for instance, a 2-winding transformer is accomplished by then scrolling down the element list.

The Element field is a normal edit field. It is therefore possible to type the identity name of the new element, like *ElmTr3* for a three-winding transformer, or *TypLine* for a line type directly into the field.

The possible list of new objects is therefore context sensitive and depends on the type or class of the originally selected folder.

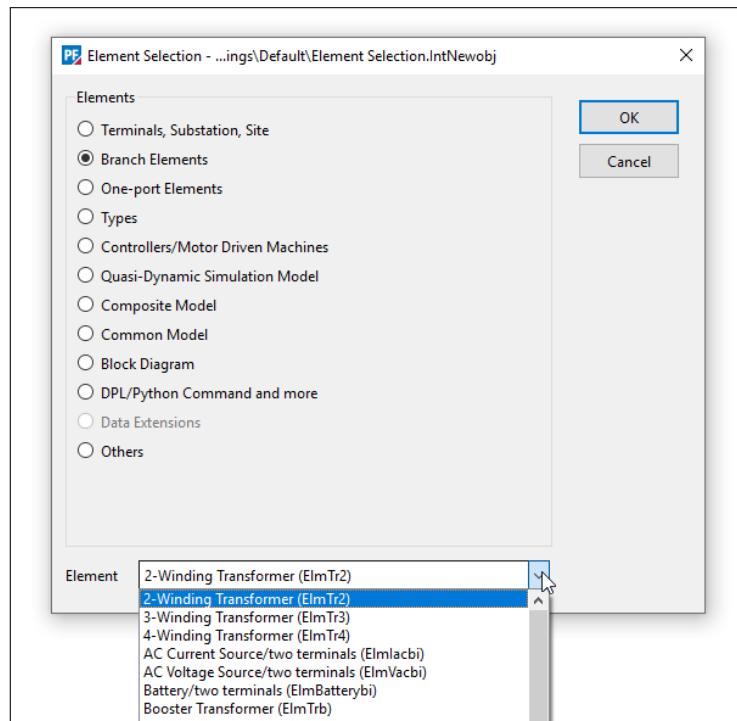


Figure 10.2.3: The element selection dialog

After the selection for a new object has been confirmed, the “Element Selection” dialog will close, the new object will be inserted into the database and the edit dialog for the new object will pop up. 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 sensitive 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 the right mouse button.

Each folder may be set to be read-only, or to be a *PowerFactory* system folder. 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: electric elements, command objects, settings, projects, etc.
- Type folders are used as ‘libraries’ for type objects.
- System folders, which are read only folders

The use of read-only folders is clear: they protect the data. In addition, folders containing data that is not normally accessed may be hidden. Selecting the kind of folders that the user/administrator wants to be hidden is done in the user settings dialog see Chapter 7 (User Settings).

### 10.2.3 Deleting an Item

A folder or object which is selected may be deleted by pressing the **Delete** key on the keyboard, or by clicking the  icon on the toolbar of the Data Manager.

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 often causes anomalies in the database consistency. Of course, *PowerFactory* knows at any moment which objects are used by which others and could prevent the user from creating an inconsistency by refusing to delete an object that is used by others.

### 10.2.4 Cut, Copy, Paste and Move Objects

#### Cut, Copy and Paste

Cutting, copying and pasting may be achieved in four different manners:

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 gray. 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 sensitive menu. All references to objects that are being moved will be updated. Cancelling a

cut-and-paste operation is performed by pressing the **Ctrl-C** key after the **Ctrl-X** key has been pressed.

3. By using the context sensitive menu. This menu offers a *Cut*, a *Copy* and a *Move* item. The move item will pop up a small second database tree in which the target folder can be selected. When the selected objects have been Cut or Copied, the context sensitive menu will then show a *Paste*, *Paste Shortcut* and a *Paste Data* item.
  - **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: off' message on the Data Manager's status bar. When the Drag & Drop option is on, it is possible to copy or move single objects by selecting them and dragging them to another folder. Dragging is done by holding down the left mouse button after an object has been selected and keeping it down while moving the cursor to the 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. To enable the 'Drag & Drop' option double click the 'Drag & Drop' message at the bottom of the Data Manager window.

---

### 10.2.5 The Data Manager Status Bar

The status bar shows the current status and settings of the Data Manager. Some of the messages are in fact buttons which may be clicked to change the settings.

The status bar contains the following messages.

- "Pause: on/off" (visible only in case of an opened input window) 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. The message is a button: double-clicking it will toggle the setting.
- "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 message will toggle the setting.

### 10.2.6 Additional Features

Most of the Data Manager functionality is available through the context sensitive menus (right mouse button).

The following items can also be found in the context sensitive menus:

**Show Reference List (*Output... → Reference List*)** Produces the list of objects that have links, or references (plus the location of the linked object), to the selected object. The list is printed to the

output window. In this manner for example, a list of elements that all use the same type can be produced. The listed object names can be double- or right-clicked in the output window to open their edit dialog.

**Select All** Selects all objects in the database browser.

**Mark in Graphic** Marks the highlighted object(s) in the single line graphic. This feature can be used to identify an object.

**Show → Station** Opens a detailed graphic (displaying all the connections and switches) of the terminal to which the selected component is connected. If the component, is connected to more than one terminal, as might be in the case of lines or other objects, a list of possible terminals is shown first.

**Goto Busbar** 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** Opens the folder in the database browser that holds the element that is connected to the currently selected element. In the case of more than one connected element, which is normally the case for busbars, a list of connected elements is shown first.

**Calculate** 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 performed 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.

Other useful features:

- Relevant objects for calculations are tagged with a check-mark sign (this will only be shown following a calculation). Editing one of these objects will reset the calculation results.

## 10.3 Searching for Objects in the Data Manager

There are three main methods of searching for objects in the data base: Sorting, searching by name and filtering.

### 10.3.1 Sorting Objects

Objects can be sorted according to various criteria, such as object class, name, rated voltage,..., etc. Sorting according to object class is done using the *Open Network Model Manager...* icon on the toolbar (☰). A browser window will open, in which the user may select a particular class of calculation-relevant object (e.g. synchronous machine, terminal, general load, but not graphics, user settings etc.).

Further sorting can be done according to the data listed in a table- either in the Data Manager or in a browser obtained using the procedure described above. This is done by clicking on the column title. For example, clicking on the column title 'Name' in a data browser sorts the data alphanumerically (A-Z and 1-9). Pressing it again sorts the data Z-A, and 9-1.

Tabulated data can be sorted by multiple criteria. This is done by clicking on various column titles in a sequence. For example, terminals can be sorted alphanumerically first by name, then by rated voltage and finally by actual voltage by pressing on the titles corresponding to these properties in reverse-sequence (actual voltage... rated voltage... name). A more detailed example follows:

Suppose that you have executed a load flow calculation and that, for each rated voltage level in the network, you want to find the terminal with the highest voltage. These terminals could be identified easily in a table of terminals, sorted first by rated voltage and then by calculated voltage. Proceed as follows:

- Perform the load flow calculation.
- Select the *ElmTerm* — class in the Network Model Manager .
- Include, in the *Flexible Data* page tab, the terminal voltage and nominal voltage (see [10.6](#)).
- In the table (*Flexible Data* page tab), click on the title 'u, Magnitude p.u' to sort all terminals from highest to lowest calculated voltage.
- Then click on the title 'Nom.L-L Volt kV' to sort by nominal voltage level.
- Now you will have all terminals first sorted by voltage level and then by rated terminal voltage.

### 10.3.2 Searching by Name

Searching for an object by name is done either in the right-hand pane of the Data Manager or in a data browser. To understand the procedure below, notice that the first column contains the symbols of the objects in the table. Clicking on such a symbol selects all columns of that row, i.e. for that object. The procedure is as follows:

- Select an object in the table by clicking on any object symbol in the table (if one object was already selected then select a different one).
- Now start typing the object name, which is case sensitive. Notice how the selection jumps as you type, For example, typing 'T' moves the selection to the first object whose name starts with T, etc.
- Continue typing until the selection matches the object that you are looking for

### 10.3.3 Using Filters for Search

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. transmission lines with a length in the range 1 km to 2.2 km, or synchronous machines with a rating greater than 500 MW etc.

The function is available in both the Data Manager and a data browser. 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 *Changed Settings \ Filters* folder.

The options in the General Filter dialog window are now explained:

**Name:** Name of filter.

**Object filter:** This field defines either the complete or a part of the search criteria, 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 id defined within the Data Manager. It allows the user to specify the folder in the database that will be searched.

**Check boxes:**

- *Include Subfolders* will search the root folder specified as well as the subfolders in the root folder. 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).
- *Area 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 **Changed 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 criterion (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 10.3.1). 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 transmission line, and the effect of such a filter criterion is to limit the data in the browser to transmission 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 sets, which are described in Section 19.3 Variable Selection.

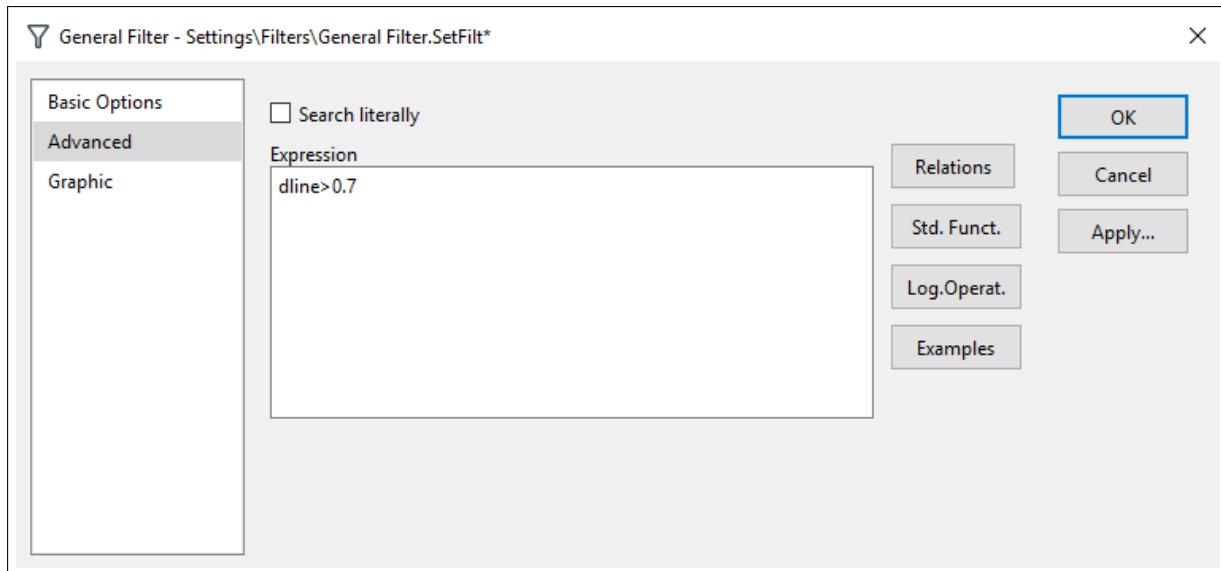


Figure 10.3.1: Filter dialog - Advanced page

**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 9: Network Graphics (Single Line Diagrams). 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 Project\Changed Settings\Filters folder in the project and are available for use directly, using the right mouse menu. If a search is to be performed in a particular grid simply proceed as follows: right-click the *grid folder* → *Find* → *Local Filters* → *Filter Name* (e.g. Lines longer than 700 m). Remember to press the **Apply** button to perform the search. If you unchecked the *Show Filter Settings before Application* box under *User Settings* → *General* then the filter will be applied as soon as it is selected from the menu. This is useful when you have already defined several filters for regular use.

---

## 10.4 Auto-Filter functions in Data Manager and browser windows

Columns within the Data Manager, browser windows or the Network Model Manager can be filtered. To add a filter for a column proceed as follows:

- Click on the down arrow in the column header of the table. A window will open.
- A list of all entries that differ from each other within that column, will be shown.
- Option: *Selection*
  - Select/deselect desired/undesired entries from that list. Only objects, which contain marked entries will later be shown in the table. All other ones will be hidden.
  - Sometimes the list of entries can be very long. To reduce this list, the user may enter a text into the search field. The list will be filtered with every typed character.
- Option: *Custom...*
  - In addition to the possibility of selecting existing entries, also 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 filter can be combined via an AND or an OR relation. For each of both filters one of the following criterions has to be chosen:
    - \* None
    - \* Equals
    - \* Is not equal to
    - \* Contains
    - \* Does not contain
    - \* Starts with
    - \* Does not start with
    - \* Ends with
    - \* Does not end with
    - \* Special
  - After one criterion has been selected (except: None), a drop-down box will appear, from which one of the entries may be chosen or an own text may be entered.

Filtered columns are indicated by a blue column heading and a filter symbol in the lower right corner of the column header. By holding the mouse cursor still over the heading of a filtered column, a balloon help appears and shows the applied filter settings.

The auto filters in the Data Manager and browser windows are temporary. They will be lost, if for example another path is chosen.

## 10.5 Editing Data Objects in the Data Manager

The Data Manager offers several ways to edit power system components and other objects stored in the database, regardless they appear graphically or not.

The basic method is to double-click the object icons in the database browser. This will open the same edit dialog window obtained, when double clicking the graphical representation of an element in the graphic window.

An open edit dialog will disable the Data Manager window from which it was opened. The edit dialog has to be closed first in order to open another edit dialog.

However, it is possible to activate more than one Data Manager (by pressing the  icon on the main toolbar) and to open an edit dialog from each of these Data Managers. This can be useful for comparing objects and parameters.

Using the edit dialogs has one major drawback: it separates the edited object from the rest of the database, making it impossible to copy data from one object to the other, or to look at other object parameter values while editing.

*PowerFactory* brings the big picture back in sight by offering full scale editing capabilities in the Data Managers browser window itself. The browser window in fact acts like a spreadsheet, where the user can edit and browse the data at the same time. The browser window has two modes in which objects can be edited,

- Object mode
- Detail Mode

which are described in the following sections.

### 10.5.1 Editing in Object Mode

In the general case the icon, the name, the type and the modification date (with its author) of the objects are shown in the 'object' mode. Certain objects, for example network components, show additional fields like the "Out of Service" field.

The title buttons are used to sort the entries in the browser. The visible data fields can be double-clicked to edit their contents, or the **F2** button can be pressed. The object will show a triangle in its icon when it is being edited.

After the data field has been changed, move to the other fields of the same object using the arrow-keys or by clicking on these data fields, and alter them too.

The new contents of a data field are confirmed by pressing the **Return** key, or by moving to another field within the same object. The triangle in the icon will change to a small star to show that the object has been altered. The object itself however has not been updated. Updating the changes is done by pressing **Return** again, or by moving to another object in the browser. By default, *PowerFactory* will ask to confirm the changes. See Section 10.2.5 (Data Manager Settings) to disable these conformation messages.

### 10.5.2 Editing in "Detail" Mode

If the  icon on the browse window of the Data Manager is pressed, the browser changes to 'detail' mode. It will display only the objects from the same class as the one which was selected when the button was pressed.

In 'detail' mode, the browser shows all data fields for the selected calculation function data set, which can be selected by clicking on a tab shown at the bottom of the table view. If a page tab is out of reach, then the page tab scrollers will bring it within the browser window again.

The list of objects may be sorted by any column by pressing the title field button. The widths of the data fields can be adjusted by pointing the mouse on the separation line between two title fields and dragging the field border by holding a mouse button down.

As with the browser in 'object' mode, the data fields can be edited by double-clicking them. In the example the active power settings are being edited, but from the star in the object icon it is clear that another field of the same object has been edited too, but not confirmed, because this star would otherwise be a triangle.

It is possible to change a parameter field for more than one object simultaneously. This is, for instance, useful to raise a certain limit for a range of objects, in order to get a better load-flow result i.e. by alleviating line overloads. An example is shown in Figure 10.5.1 where the derating factor for a range of lines is changed at once.

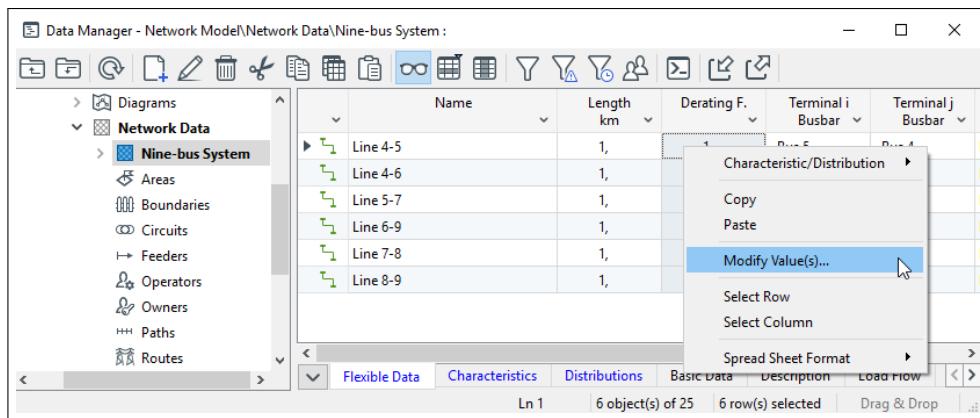


Figure 10.5.1: Modify values dialog

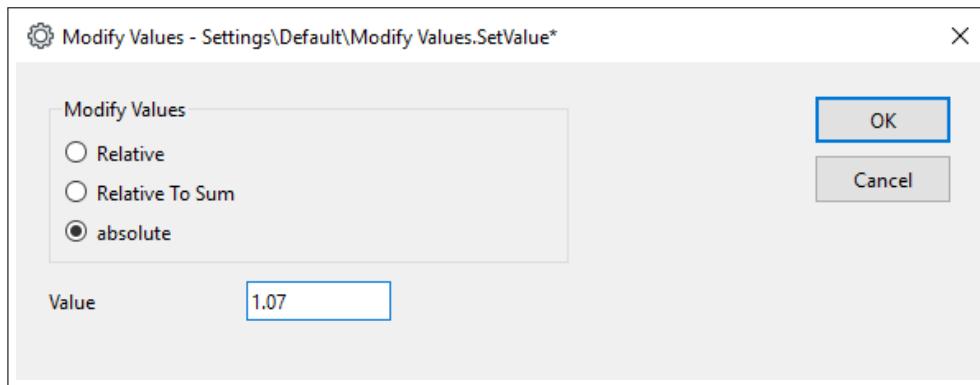


Figure 10.5.2: Modify values dialog

The parameter fields which have to be changed have to be multi-selected first. Right-clicking the selection will pop up a case sensitive menu from which the *Modify Value(s)* option opens the *SetValue* dialog, see Figure 10.5.2.

This dialog can be used to:

- increase or decrease them by multiplication with a scale factor ("Relative").
- increase or decrease them by multiplication with a scale factor with respect to the sum of values selected ("Relative to Sum").

- Set all the selected parameter fields to a new fixed (“absolute”) value.

It is not possible to simultaneously alter parameter fields from more than one column, i.e. to change nominal currents and nominal frequencies simultaneous, even if they would happen to take the same value or would have to be raised with the same percentage.

### 10.5.3 Copy and Paste while Editing

One of the great advantages of editing data fields in the Data Manager’s browser window is the possibility to copy data from one object to another. This is done by selecting one or more objects or object fields, copying this selection to the clipboard, and pasting the data back in another place.

To copy one or more objects,

1. Open the Data Manager and select the grid folder where you find the objects to be copied.
2. Select the objects.
3. Press **Ctrl-C** to copy or use the  icon on the Data Manager toolbox.
4. Press **Ctrl-V** to paste or use the  icon on the Data Manager toolbox. The objects will be copied with all the data. Their names will automatically be altered to unique names.

Copying data fields from one object to another is done just like for any spreadsheet software you may be familiar with. To copy one or more data fields,

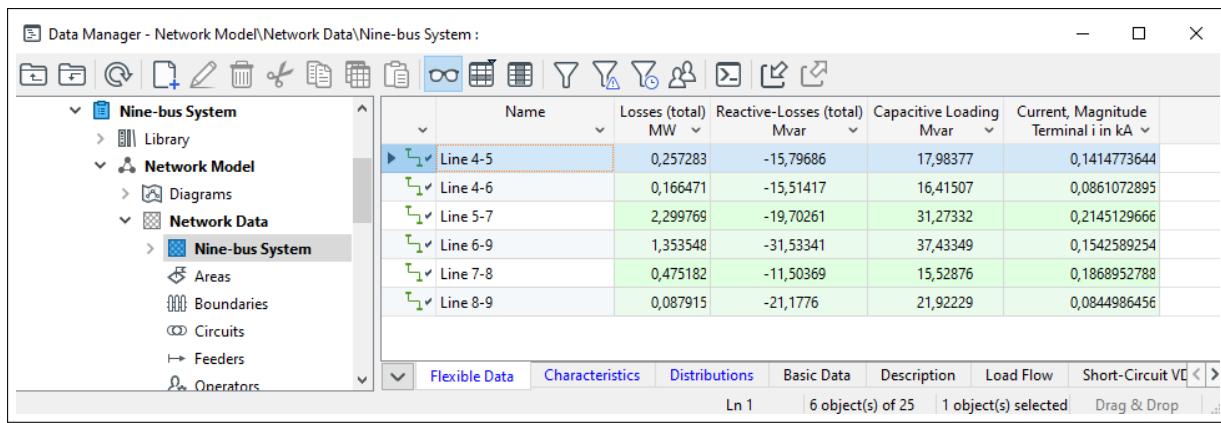
1. Select them by clicking them once. Select more data fields by holding down the **Ctrl** key.
2. Copy the fields to the clipboard by pressing **Ctrl-C** or the  icon.
3. Select one or more target objects data fields. If more than one field was copied, make sure that the target field is the same as the first copied data field.
4. Press **Ctrl-V** or the  icon. The contents of the data fields will be copied to the target objects.

## 10.6 The Flexible Data Page

The data browser (this will be seen in the Data Manager when the ‘Detail Mode’ has been engaged) has page tabs for all calculation functions. These tabs are used to view or edit object parameters which are categorised according to a calculation function and have a fixed format.

The ‘Flexible Data’ tab, 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 page depends on the calculation performed: 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. Figure 10.6.1 shows an example of the flexible data page tab.



The screenshot shows the Data Manager interface with the following details:

- Left Panel:** Shows a tree structure under "Nine-bus System". The "Network Data" node is expanded, and its "Nine-bus System" child node is also expanded, showing "Areas", "Boundaries", "Circuits", "Feeders", and "Operators".
- Central Area:** A table titled "Flexible Data" displays data for various lines. The columns are: Name, Losses (total) MW, Reactive-Losses (total) Mvar, Capacitive Loading Mvar, and Current, Magnitude Terminal i in kA.
- Table Data:**

Name	Losses (total) MW	Reactive-Losses (total) Mvar	Capacitive Loading Mvar	Current, Magnitude Terminal i in kA
Line 4-5	0,257283	-15,79686	17,98377	0,1414773644
Line 4-6	0,166471	-15,51417	16,41507	0,0861072895
Line 5-7	2,299769	-19,70261	31,27332	0,2145129666
Line 6-9	1,353548	-31,53341	37,43349	0,1542589254
Line 7-8	0,475182	-11,50369	15,52876	0,1868952788
Line 8-9	0,087915	-21,1776	21,92229	0,0844986456
- Bottom Navigation:** Buttons for "Flexible Data", "Characteristics", "Distributions", "Basic Data", "Description", "Load Flow", and "Short-Circuit VD". Status bars show "Ln 1", "6 object(s) of 25", "1 object(s) selected", and "Drag & Drop".

Figure 10.6.1: The Flexible Data page tab

### 10.6.1 Customising the Flexible Data Page

The displayed variables are organised in 'Variables Sets' that are, in turn, organised according to the calculation functions. For example, an object class *ElmTr2* (two-winding transformer) has a variable set for symmetrical load flow calculation, a variable set for short-circuit calculation etc. There may also be more than one variable set for any calculation function. For example, the object *ElmTr2* may have two variable sets for symmetrical load flow calculation.

The Flexible Page Selector allows the user to specify the variable set to use, or to define new variable sets. Furthermore, the Flexible Page Selector allows the user to access and edit the variable sets, i.e. to specify which variables to display in the Flexible Data page.

The 'Flexible Page Selector' dialog is shown in Figure 10.6.2. This dialog is opened by pressing the (grid icon) on the Data Manager toolbar. The Flexible Page Selector has a menu with all the different calculation functions. It opens in the page corresponding to the most recent calculation.

The selection of variables within Variable Sets is presented in detail in Section 19.3 Variable Selection.

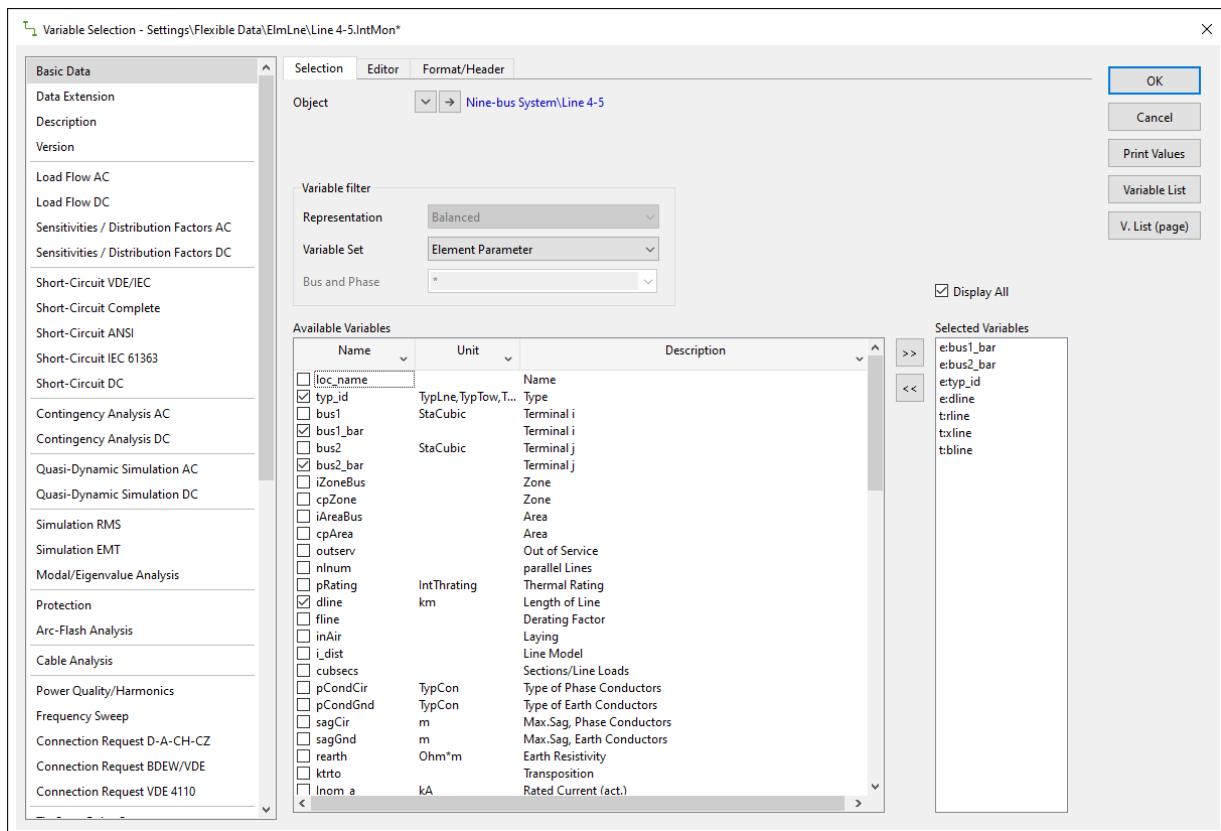


Figure 10.6.2: The Flexible Page Selector

The Format/Header tab (Figure 10.6.3) allows the user to customise the formats or column headers of the Flexible Data page.

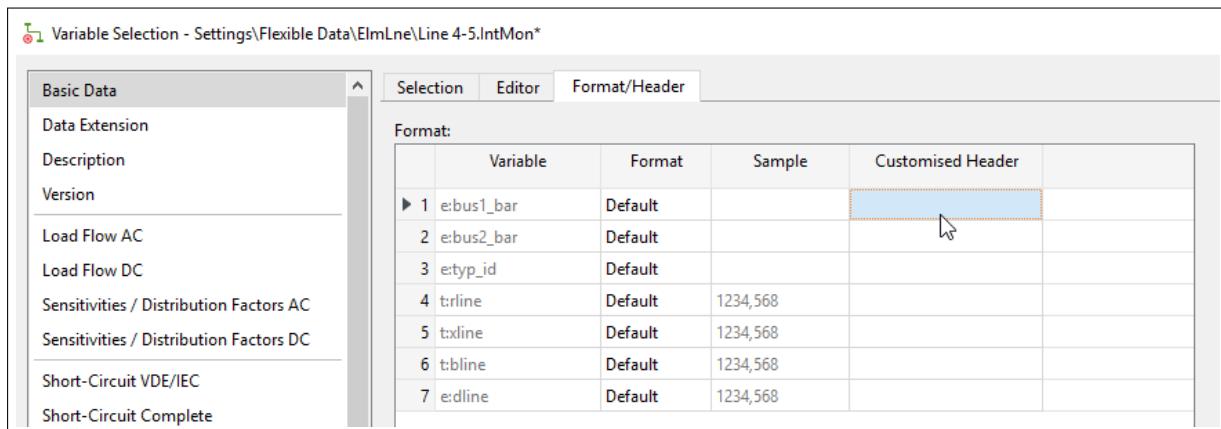


Figure 10.6.3: Customising flexible data page formats and column headers

**Note:** Variable Sets are objects of class *IntMon*, within *PowerFactory* they have multiple uses. This section only presents their use in conjunction with Flexible Data. For further information refer to Section 19.3 Variable Selection.

The number format per column in the Flexible Data Page can also be modified by right clicking on the column header of the variable and selecting *Edit Number Format* .... A new window will appear and the user may define the number representation.

The order of the columns (except: *Name*, *In Folder* and *Grid*) on the Flexible Data page can be changed. Therefor, the header of a column has to be clicked and while holding the left mouse button pressed, the column can be moved to the desired position. To illustrate, where the column will be placed, when the mouse button is released, an arrow between the actual and the possible new position of the column is shown during this process.

## 10.7 The Input Window in the Data Manager

The input window is for the more experienced users of DLgSILENT *PowerFactory*. It is closed by default. 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.

### 10.7.1 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.

## 10.8 Save and Restore Parts of the Database

A selected part of the database can be written to a “.pdf” file with the button **Export Data...** . This will bring a ‘File Save’ dialog where a filename must be specified.

Alternatively, the folder or object that is to be exported can be right-clicked in the database tree, after which the option *Export...* is selected.

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, inclusive 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** an user-account. Exporting the user-account on a regular basis is a practical way to backup your data.

It is even possible to export data from another user account, or even to export another user-account completely. However, only the shared, visible, data will be exported.

The exported data file can be imported into the database again in any desired folder by pressing the **Import Data...**  button. This will bring a ‘File Open’ dialog where the “.pdf” data-file can be selected.

The “.pdf”-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 shown also, which is the original folder of the saved data. If this is not desired, another target folder can be selected by pressing the **Drop Down** button. This button will bring a small version of the database tree. A new target folder can be selected from this tree.

### 10.8.1 Notes

By exporting a folder from the database, only the information in that folder and all its subfolders will be stored. If the exported objects use information (e.g. power system types like line or transformer types) that is saved somewhere else, then that information will not be stored. Make sure that the used power system types and all other referenced information is exported too.

When importing a file that contains objects which use data outside the import-file, a search for that data is started.

For instance, assume a project is exported. One of the line-models uses a type from a library outside the project. When exporting, the path and name of this type is written in the export-file, but the type itself is not exported, as it does not reside in the exported project.

At importing, the stored path and name of the ‘external’ type is used to find the type again and to restore the link. However, if the ‘external’ type is not found, then it will be created, using the stored path and name. Of course, the created object has default data, as the original data was not exported. Additionally, an error message is written to the output window.

Suppose that you are working with a large library, which is stored in a special user-account to make it read-only. The library is made accessible by sharing it to all users.

When export the projects, the objects from the external library are not exported. However, a colleague which has access to the same library may still import your projects without problems. The external objects used in your projects will be found in the same location, and the links to these objects will be correctly restored.

## 10.9 Spreadsheet Format Data Import/Export

The *PowerFactory* data browser in the Data Manager's window 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 data browser offers 'Spreadsheet Format' import and export facilities.

### 10.9.1 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 performed 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.
- 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-sensitive 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 10.9.1.
- The exported data can now be pasted into a spreadsheet program.

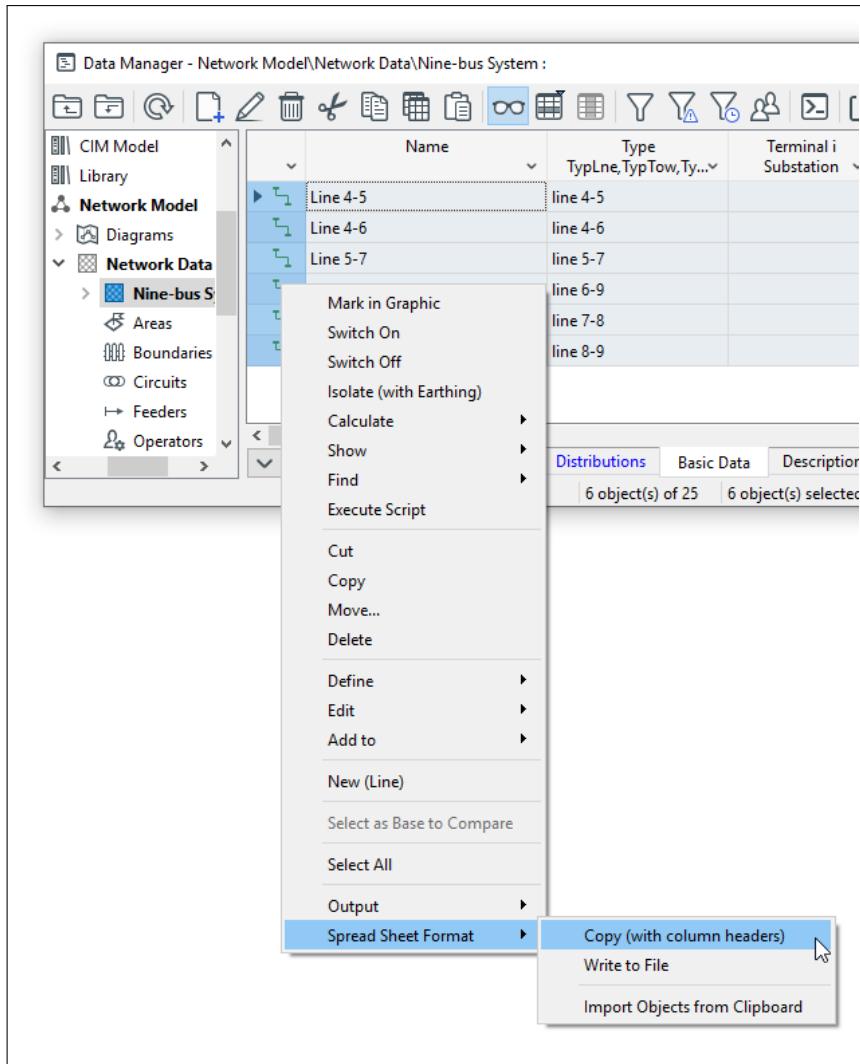


Figure 10.9.1: Exporting a range of data

## 10.9.2 Import from Spreadsheet Programs (e. g. MS EXCEL)

There are two methods available for importing data from a spreadsheet program. The first method uses a direct import of 'anonymous' numerical data, i. e. of the values stored in the cells of the table. This method is used to change parameter of existing objects by importing columns of parameter values.

The second method can be used to create new objects (or replace whole objects) by importing all the data from a spreadsheet.

Any range of parameter values can be copied from a spreadsheet program and imported into the Data Manager. The import is performed by overwriting existing parameter values by 'anonymous' values. The term 'anonymous' expresses the fact that the imported data has no parameter description. The size of the imported value range and the required data are tested. Importing invalid values (i.e. a power factor of 1.56) will result in an error message.

### Spreadsheet Import of Values

The import of values (anonymous variables), i. e. cells of a table, is explained by the following example.

In Figure 10.9.2, a range of active and reactive power values is copied in a spreadsheet program. In Figure 10.9.3, this range is pasted to the corresponding fields of 6 load objects by right-clicking the

upper left most field which 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.

A	B	C	D	E	F
1 Name	P	Q	Calibri	11	A A
2 load A	100	50	B I		
3 load B	50	30			
4 load C	75	35			
5					
6					
7					

Figure 10.9.2: Copying a range of spreadsheet data

Name	Terminal StaCubic	Terminal Busbar	Act.Pow. MW	React.Pow. Mvar	Ap
Load A		Bus 5	125,	50,	12
Load B		Bus 6	90,	30,	9
Load C		Bus 8	100,	35,	10

Figure 10.9.3: 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, which is 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 10.9.4.

	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 10.9.4: Excel required format

Figure 10.9.5 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	Line Types								
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	2-Winding Transformer Types								
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 10.9.5: Example of valid spreadsheet data

The import of the spreadsheet data into *PowerFactory* is performed as follows.

- Select the header line and one or more objects lines.
- Copy the selection. See Figure 10.9.6 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*. See Figure 10.9.7 for example.

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 10.9.6: Selecting object data in spreadsheet

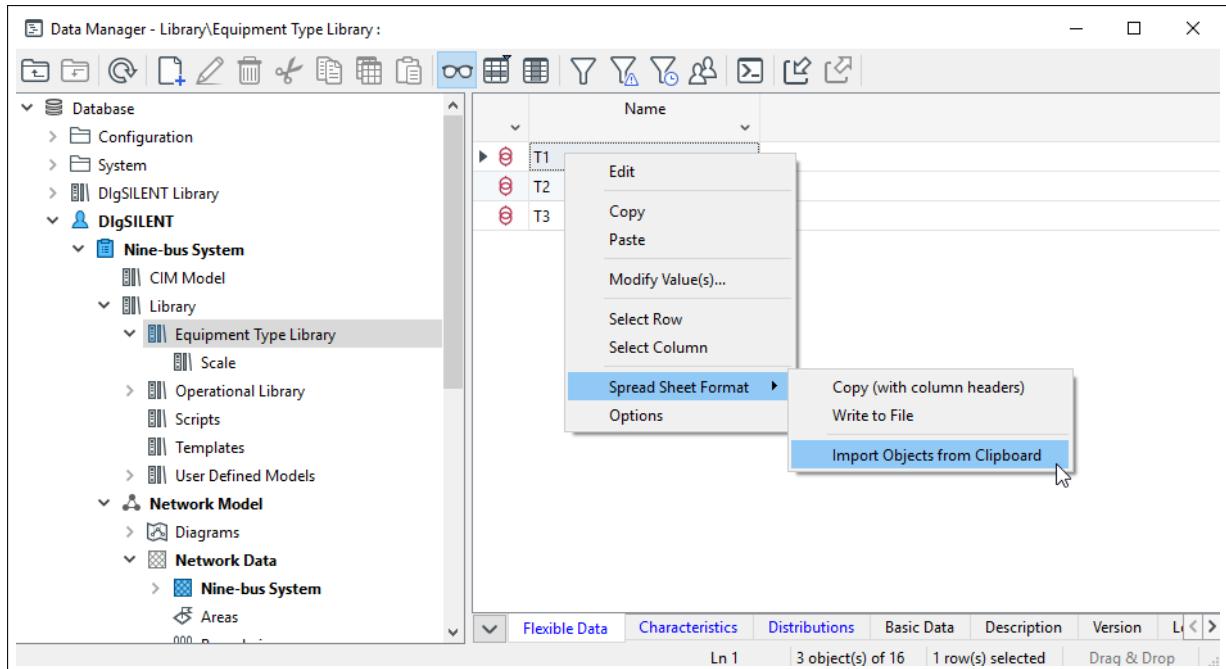


Figure 10.9.7: Importing objects from clipboard

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 10.9.8, none of the imported objects existed in the database an all were created new therefore. The example shows the database in detail mode.

	Name	rtd.Pow. MVA	Nominal Frequency Hz	HV-rtd.Volt. kV	LV-Rtd.Volt. kV
▶	T1	250,	60,	230,	16,5
▶	T2	200,	60,	230,	18,
▶	T3	150,	60,	230,	13,8

Figure 10.9.8: 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 then its data will be overwritten by the imported data

Because new objects are only created when they do not exist already, and only the imported parameters are overwritten when the object did exists already, the import is always a save action.

#### Remarks

##### Object Names

Object names may not contain any of the characters

\*? = " , \ ~ |

##### Default Data

When an imported object is created newly, 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*.

# Chapter 11

## Building Networks

### 11.1 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 [9](#) and [10](#).

### 11.2 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.

#### 11.2.1 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 is 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.

---

## 11.2.2 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.

## 11.2.3 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 “Graphic Options” 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* → *Flip At Node*.

---

Once drawn, an element can be rotated by right-click and selecting from the *Rotate* commands.

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 element and 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 11.3 (Lines and Cables).

#### 11.2.4 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.

### 11.2.5 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 sensitive 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 Data*. If multiple objects are selected, *right-click* → *Edit Data* will bring up a data browser.

The option *Edit and Browse Data* 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 Edit and Browse. This will open a new database browser when required, and will focus on the selected object.

### 11.2.6 Interconnecting Power Subsystems

Interconnections between two different graphics can be done in one of two ways:

- 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 Graphic Only*) into the second diagram. Both graphical objects are then associated with the same element; no new element is created.
- 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 11.2.1.

- Select a line drawing tool from the toolbox and create the first connection as normal by left clicking a node (see figure 11.2.1a).
- 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 11.2.1b).
- Once in the second graphic left click to place the line symbol (see figure 11.2.1c) and then left click on the second node.

The interconnected leg is shown by a  $\gg$  symbol.

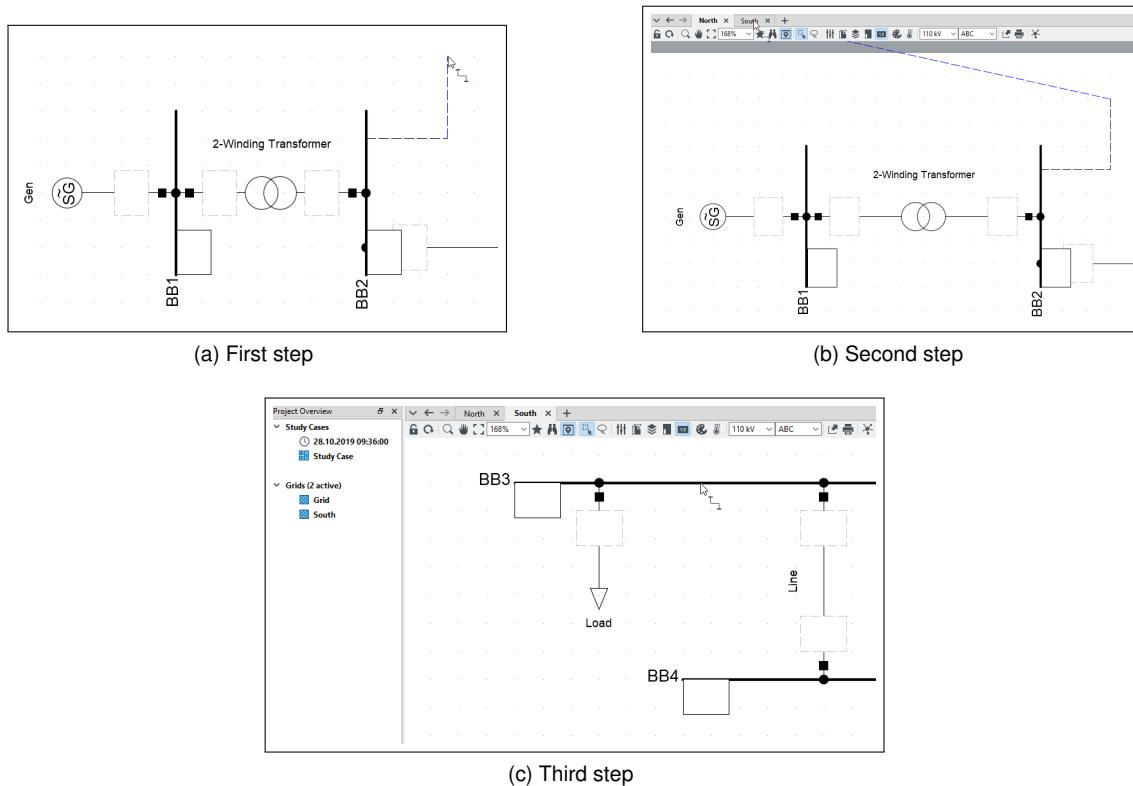


Figure 11.2.1: Interconnecting Power Subsystems

---

**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 *Show Detailed Graphic 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.

---

## 11.2.7 Substations

Substations and Secondary Substations from existing templates are created using the network diagrams. The substations are represented in these diagrams by means of composite node symbols.

### 11.2.7.1 Creating a New Substation in an Overview Diagram

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 [8.1.2.3](#). More information about Bays is given in section [11.2.7.3](#).

Overview diagrams are diagrams without detailed graphical information of the substations. Substations and Secondary Substations are illustrated as "Composite Nodes", which can be coloured to show the connectivity of the connected elements ("Beach Ball"). Substations and Secondary Substations from pre-defined templates (or templates previously defined by the user) are created using the network diagrams. The substations are represented in these diagrams by means of composite node symbols.

To draw a substation from an existing template in an overview diagram:

- In the Drawing Toolbox, click on the symbol of the composite node (○ or □ for substations or ○ for secondary substations).
- Select the required substation template from the list.
- Click on the overview diagram to place the symbol. The substation is automatically created in the active grid folder.
- Right click on the substation, select *Edit Substation* and rename the substation appropriately.
- Close the window with the templates.
- Press **Esc** or right click on the mouse to get the cursor back.
- Resize the substation symbol as required, by clicking on it and dragging the corners or sides.

For further information on templates refer to Chapter [14](#): Project Library, Section [14.4](#) (Templates Library).

#### To show the connectivity inside a composite node:

Press the  button to open the colouring dialog. Select the 'Function' for which the colouring mode is relevant (for example, select the *Basic Data* page). Select *Other* → *Topology* → *Station Connectivity*.

There are two ways to open the graphic page of a substation. The first is to double-click on the corresponding composite node in the overview diagram. The second is to go to the graphic object of the substation in the Data Manager, right-click and select *Show Graphic*.

### 11.2.7.2 Creating a New Substation in a Simplified Diagram

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 8.1.2.3. More information about Bays is given in Section 11.2.7.3.

Simplified diagrams are substation graphics which are more detailed than in the overview diagram but less detailed than the detailed substation diagram. Figure 11.2.2 shows the different possible representations of a substation.

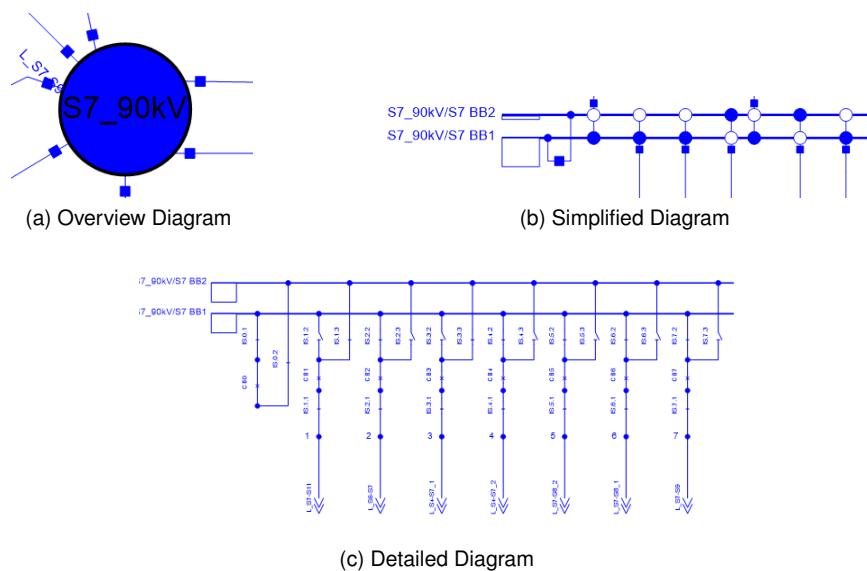


Figure 11.2.2: Substation examples

To create a substation from an existing template in a simplified diagram:

- Click on the *General Templates* symbol (  - Click on the *Library Busbar Systems* folder and select the desired substation template from the list.
  - Click on the single line diagram to place the symbol. The substation is automatically created in the active grid folder.
  - Right click the substation, select *Edit Substation*, and rename the substation appropriately.
  - Close the window with the templates.
  - Press **Esc** or right click on the mouse to get the cursor back.
  - Resize the substation symbol as required.

A detailed diagram of the newly created substation can be opened by selecting the option *Show Detailed Graphic of Substation* from the context sensitive menu.

For further information on templates refer to Chapter 14: Project Library, Section 14.4 (Templates Library).

The connectivity can be shown as explained in section 11.2.7.1.

### 11.2.7.3 Substation 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 but is also used by the Load Flow Calculation option *Calculate max. current at busbars*: see section [48.4.6](#).

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.

### 11.2.7.4 Substation 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 [44](#))

#### 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-sensitive menu choosing *Edit a Substation*, or by clicking on an empty place in the *substation graphic*, and from the context-sensitive 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 [11.2.3](#). Press afterwards **OK**.
- The new switching rule is automatically stored inside the substation element.

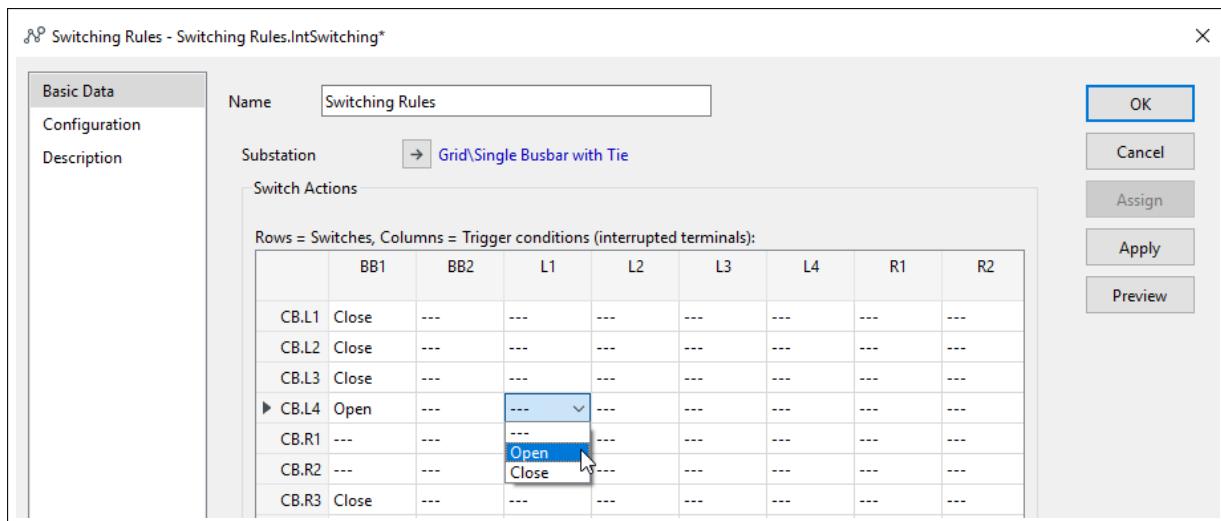


Figure 11.2.3: Switching Rule Dialog

### 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-sensitive 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.

## 11.2.8 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.

### 11.2.8.1 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 ( )
- Click on the overview diagram to place the symbol. The site is automatically created in the active grid folder.
- Press **Esc** or right click on the mouse to get the cursor back.
- Resize the site symbol as required.
- Right click on the site and select *Edit Site* to open the edit dialog of the element.

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 11.2.7.2.

If the site already exists it is possible to use the *Diagram Layout Tool* to generate its detailed representation automatically. See section 11.6 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 11.2.7.1.

### 11.2.8.2 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.

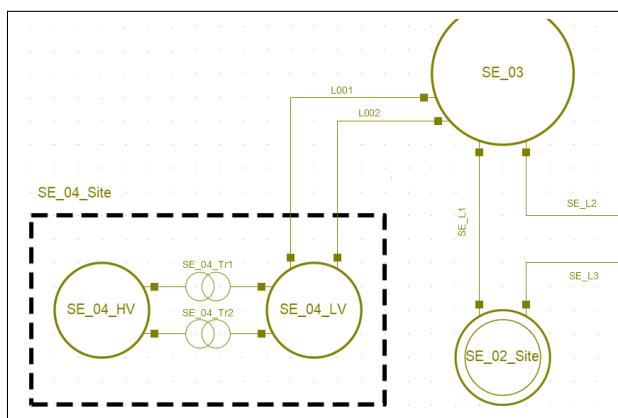


Figure 11.2.4: Substations within a site frame in an overview graphic

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.

### 11.2.9 Composite Branches

New composite branches (*ElmBranch*) can be created in the Data Manager using the procedure described in Chapter 10, Section 11.5.4 (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 11.2.6 (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 templates can be found in Chapter 14 (Project Library).

### 11.2.10 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 (→) 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 [9.3.7.1](#) (Diagram Colouring).

---

## 11.3 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 [11.3.1](#):

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 (*ElmLnesec*). The *ElmLne* object is again connected via cubicle objects between two terminal objects, but here the line section (*ElmLnesec*) 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.

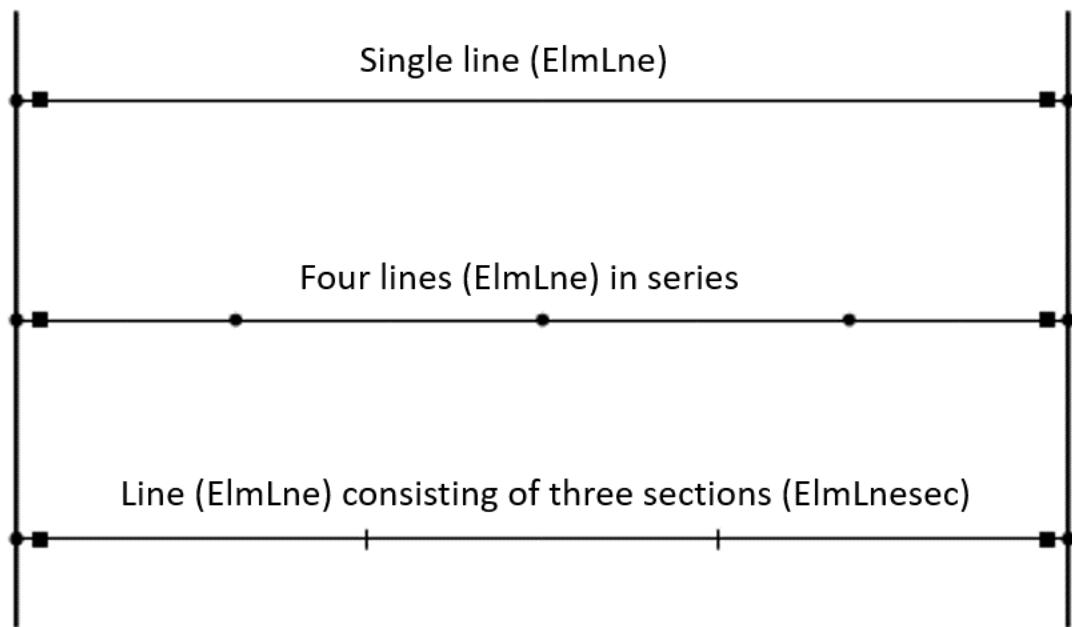


Figure 11.3.1: Basic line configurations

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 [11.2.9](#) and [11.5.4](#).

### 11.3.1 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.

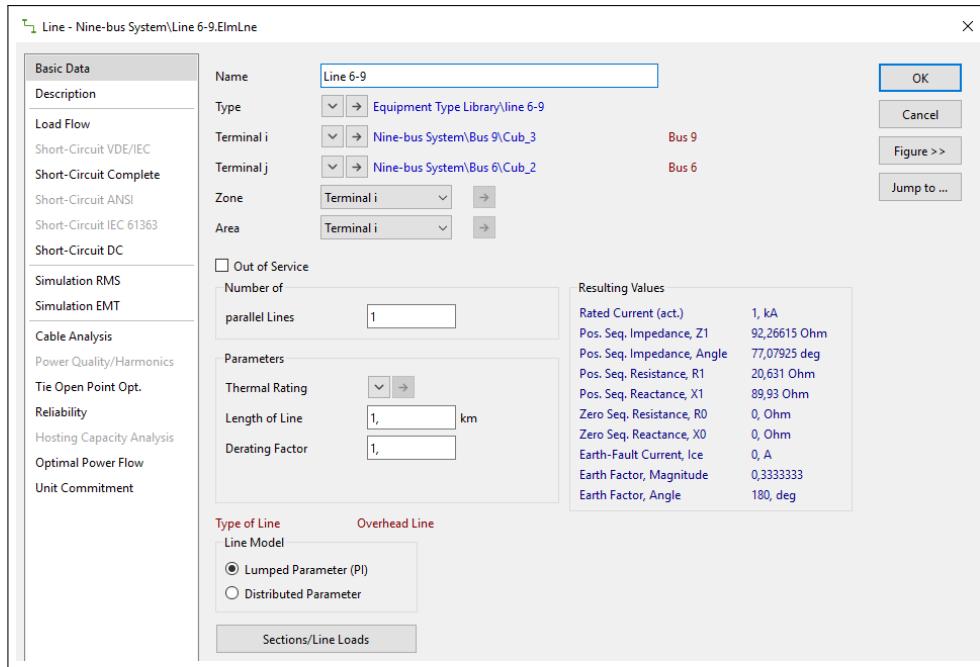


Figure 11.3.2: Editing a transmission line

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 11.2.10 (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](#).

## 11.3.2 Defining Line Sections

To divide a line into sections:

- Press the **Sections/Line Loads** 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.

### 11.3.3 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 lines drawn in the single line diagram, right-clicking and select *Define → Line Couplings* from the context sensitive menu.
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 11.3.3, the number of circuits and earth wires should be defined. Then the conductor types should be selected, by double clicking on the *TypCon* field.

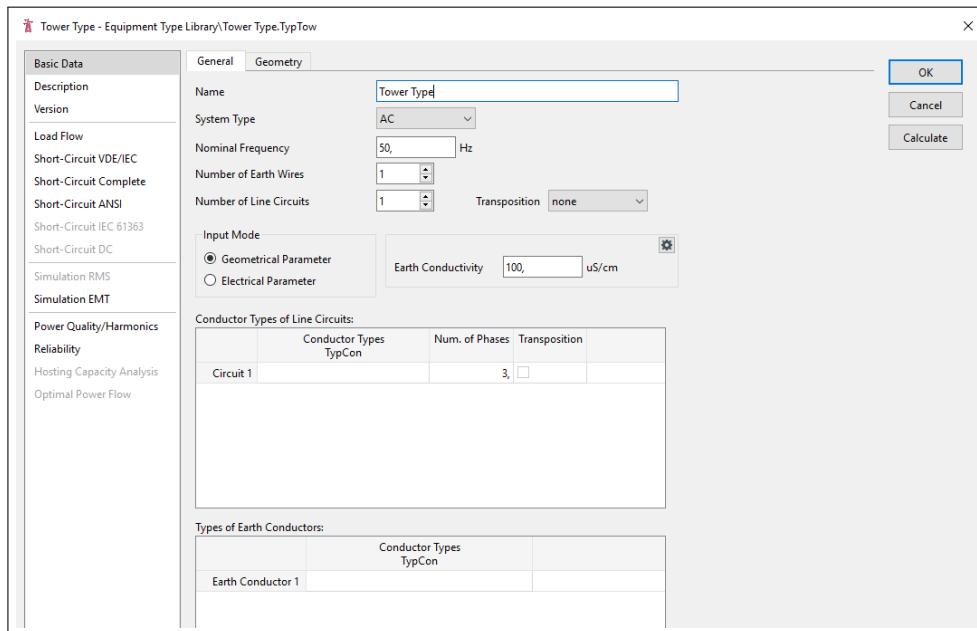


Figure 11.3.3: Tower Type (*TypTow*)

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*).

- On the edit dialog of the conductor type, shown in figure 11.3.4, the nominal voltage, number of subconductors, model and measurements should be defined.

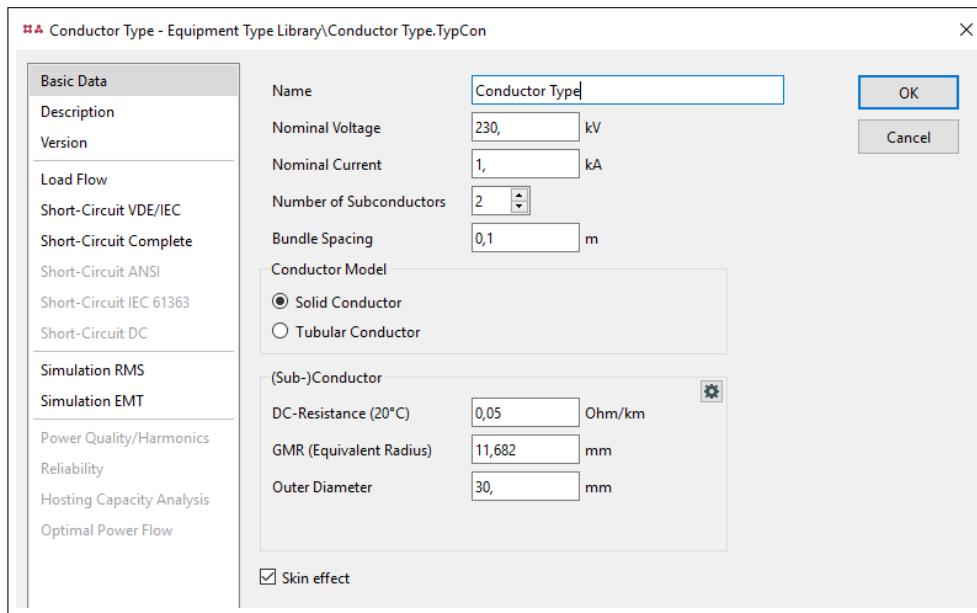
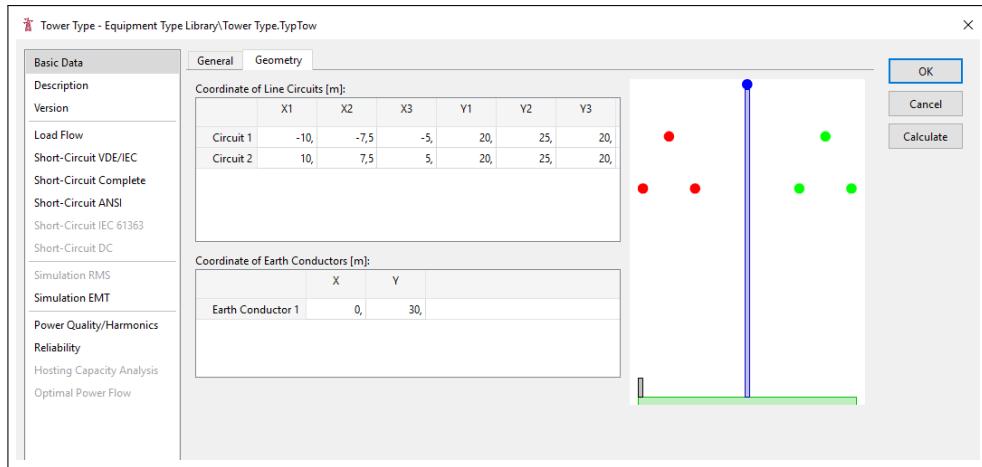


Figure 11.3.4: Conductor Type (*TypCon*)

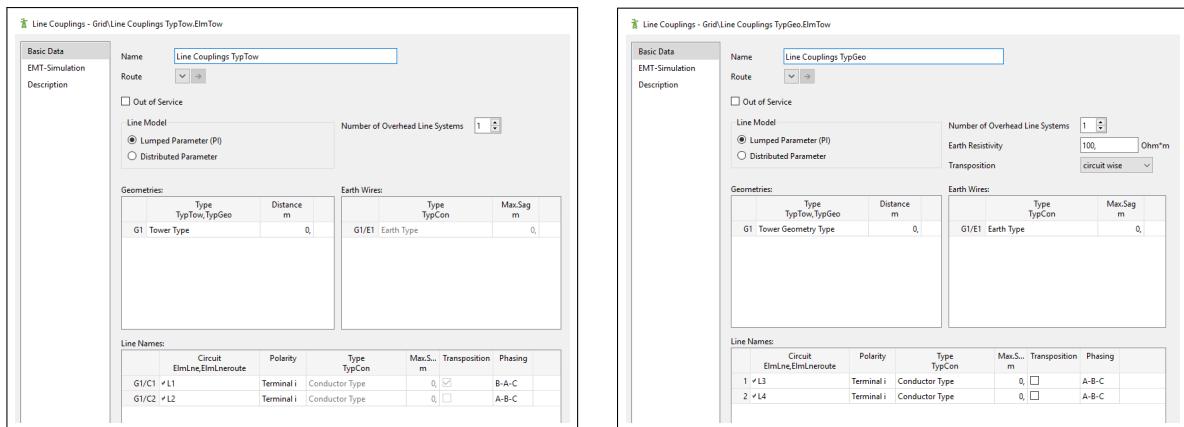
- Separate conductor types can be used for each circuit and earth wires. Note that earth wires are only to be entered if the *Input Mode* is set to *Geometrical Parameters*.
- Once the conductors are defined, the rest of the parameters of the tower type should be entered. The transposition can be selected as:
  - None
  - Circuit-wise
  - Symmetrical
  - Perfect
 More information about these options can be found in the [Technical References Document](#) (Overhead Line Constants).
- 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 11.3.5: Geometry of the Tower Type (*TypTow*)

- 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.



(a) Using Tower Type

(b) Using Tower Geometry Type

Figure 11.3.6: Line Couplings Element

#### 11.3.4 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 sys-

tem. 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 11.3.7 illustrates that there are essentially two different kinds of cable system that can be constructed in *PowerFactory*.

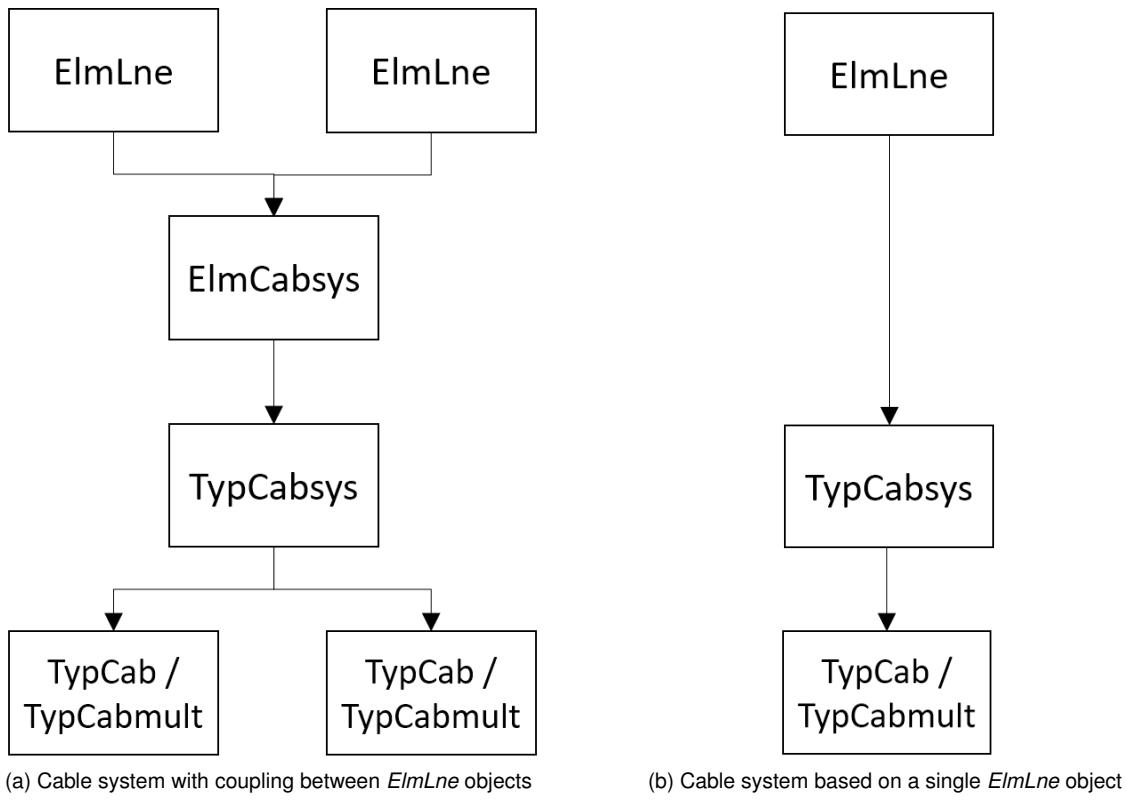


Figure 11.3.7: Cable System Overview

The cable system shown on figure 11.3.7a 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 *TypCabor* *TypCabmult* objects.

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 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. Multi-select the cables to be coupled which are drawn in the single line diagram. Right-click and select *Define → Cable System* from the context sensitive menu.
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 tower type, shown in figure 11.3.8, the number of circuits should be defined. Then the cable types should be selected, by double clicking on the *TypCab*, *TypCabmult* field.

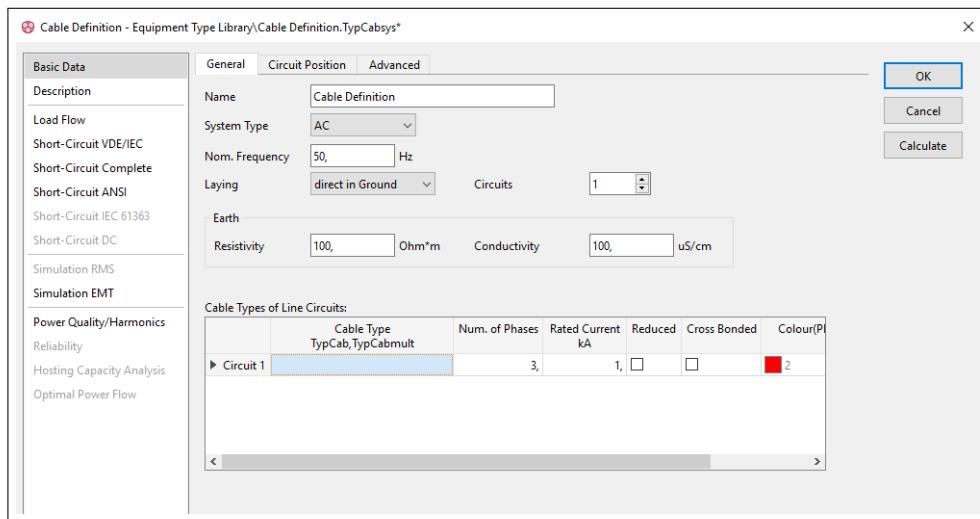
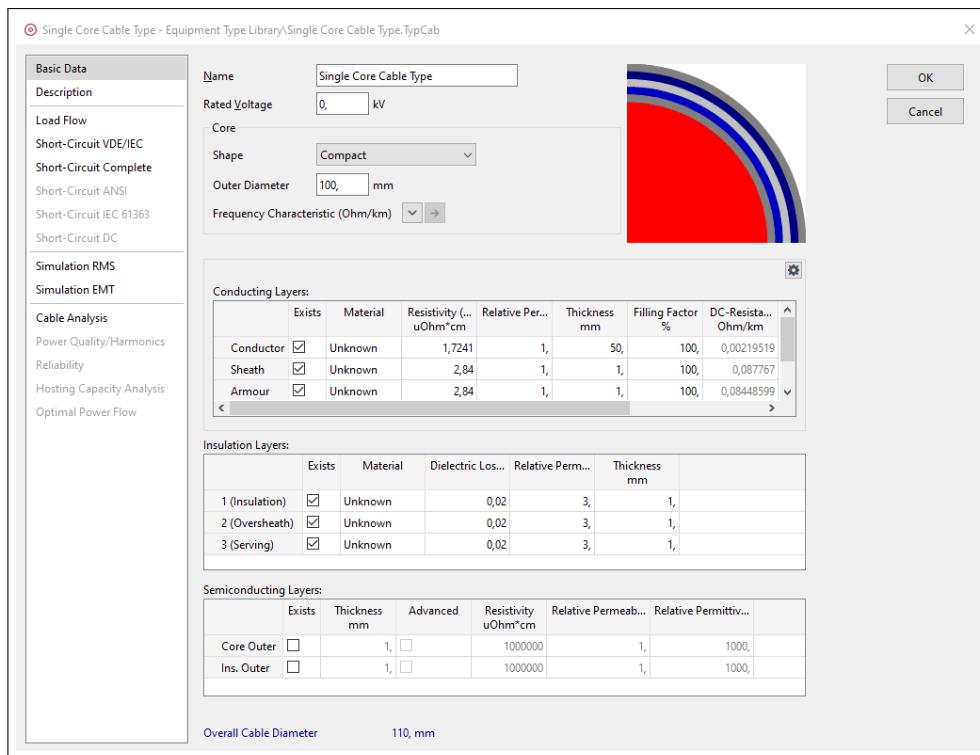
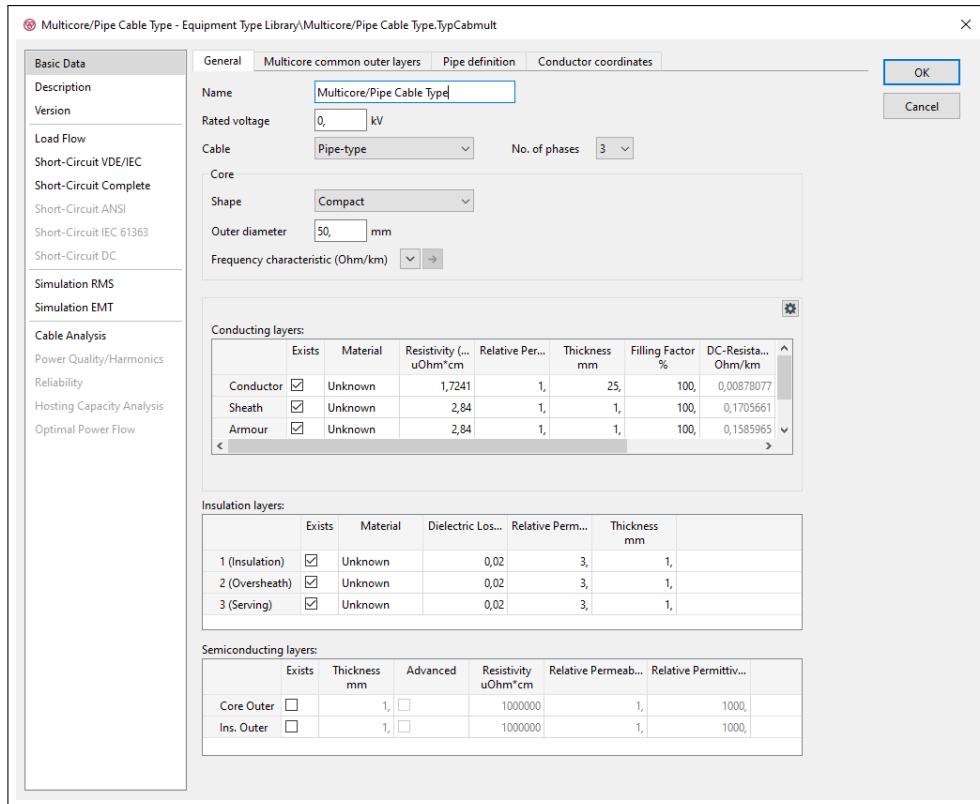


Figure 11.3.8: Cable Definition Type (*TypCabsys*)

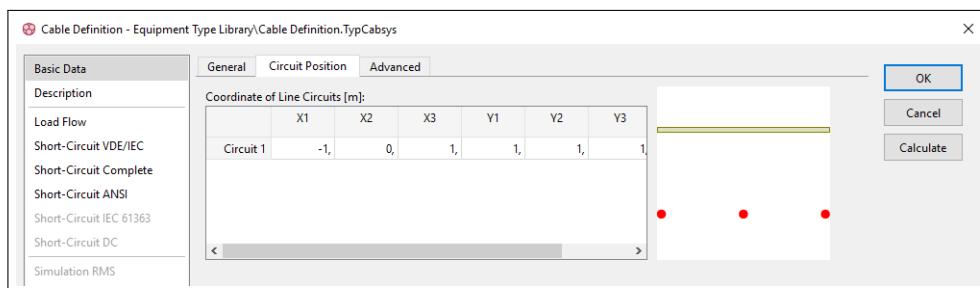
4. Once again, a dialog pointing to the *Equipment Type Library* will open for selection of a Single Core or Multicore/Pipe Cable Type. If no appropriate type is yet defined, the button *New Object* () can be used to define a new type.
5. On the Basic Data page of the dialog of the Single Core Cable type, shown in figure 11.3.9, the nominal 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 11.3.9: Single Core Cable Type (*TypCab*)

- On the Basic Data page of the dialog of the Multicore/Pipe Cable Type, shown in figure 11.3.10, just as for the Single Core Cable type the nominal voltage, and details of the core construction should be defined along with the characteristics of the various conducting, insulating and semiconducting 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 11.3.10: 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, the positions of the circuits can be defined by inserting the coordinates, 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 Cable Systems.

Figure 11.3.11: 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 11.3.7b 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.

## 11.4 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*s)
- 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 11.4.1, the graphical symbol of the object will change. An illustration for the Synchronous Generator element is shown in figure 11.4.2. 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.

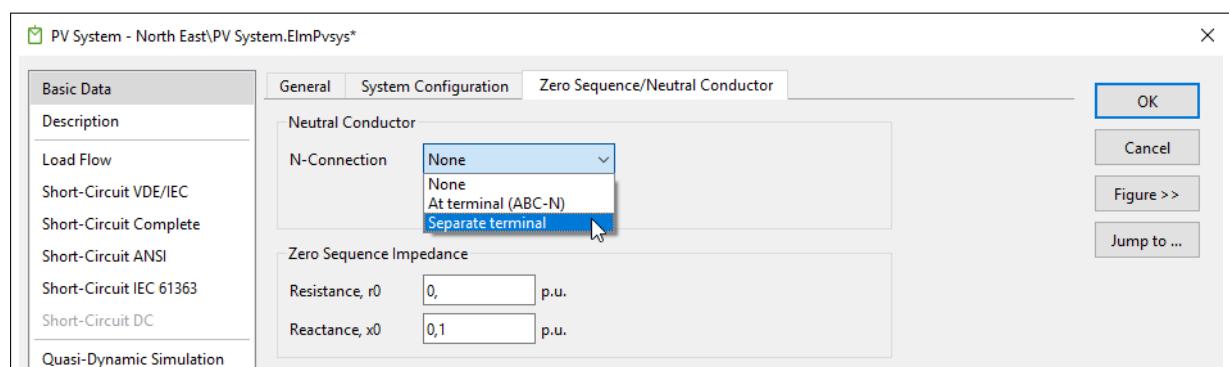


Figure 11.4.1: Zero Sequence/Neutral Connection Tab

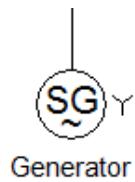


Figure 11.4.2: Generator with N-Connection via separate terminal

To connect the neutral of the Element to a neutral busbar, right click on the element and select Connect Element. An example of a single line diagram with the interconnection of neutral wires is shown in figure 11.4.3. A Neutral terminal is configured by ensuring that the Phase Technology of the terminal is set to N as shown in figure 11.4.4.

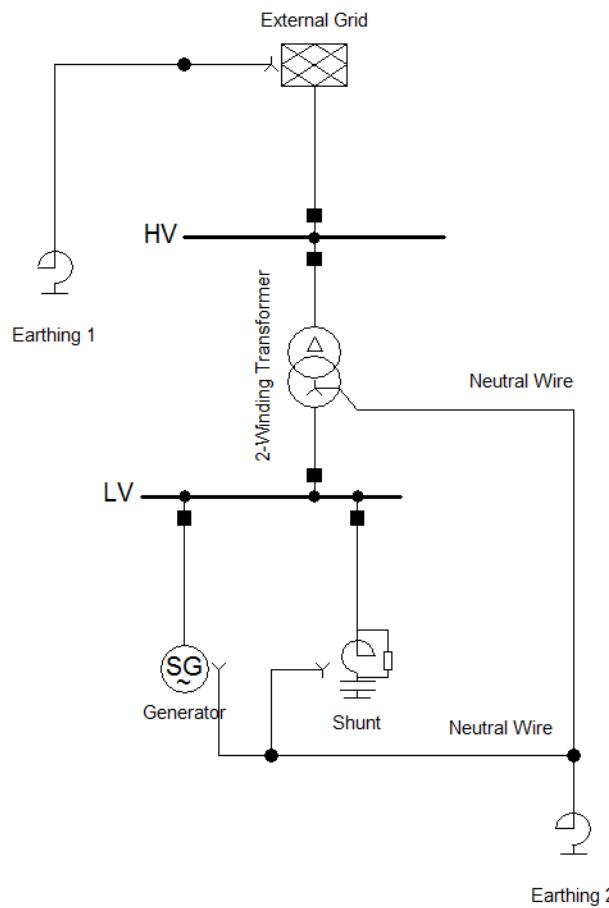


Figure 11.4.3: Grid with neutral winding connection

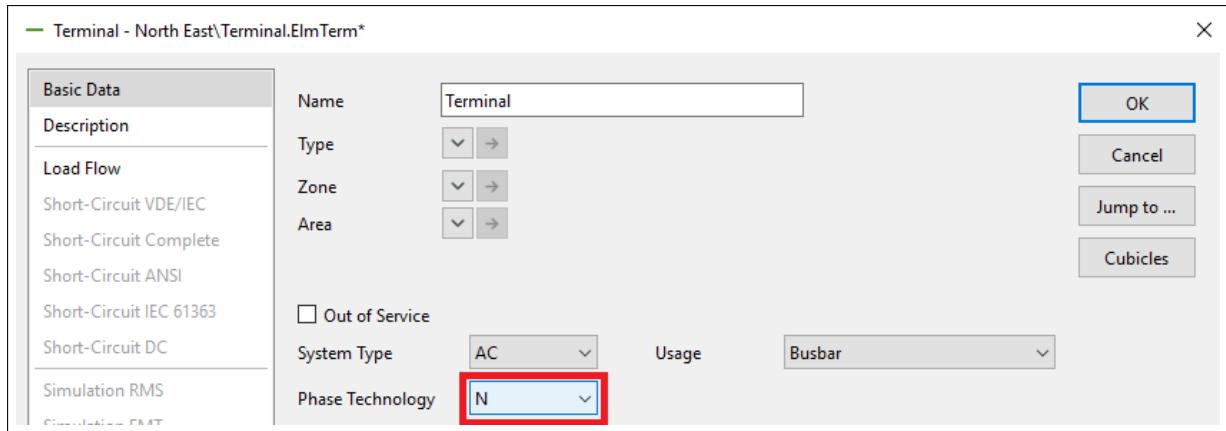


Figure 11.4.4: Set neutral Terminal

## 11.5 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.

### 11.5.1 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 section [10.5](#).

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.

### 11.5.2 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 in the 'Element' field select *Cubicle (StaCubic)*. 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.

---

### 11.5.3 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 [11.2](#) (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 sensitive 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 [11.2](#) (Defining Network Models using the Graphical Editor). For information about the definition of Running Arrangements refer to Section [14.3.10](#) (Running Arrangements).

#### 11.5.4 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 [11.2](#) (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 sensitive 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 [11.5.2](#).
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.

### 11.5.5 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 sensitive 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.

---

## 11.6 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, 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 in the Project Overview window and choosing *Show Graphic → 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. It replaces the *Draw Existing Net Elements* tool of previous versions and enhances its functionality by a semi- and fully automated creation of network diagrams.

### 11.6.1 Action

#### 11.6.1.1 Generate new diagram for

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 → Entire grid* are set in pages

*Node Layout* (section 11.6.2), *Edge Elements* (section 11.6.3) and *Protection Devices* (section 11.6.4)

- **detailed representation of:** this option can be used for substations, branches and sites. It creates a detailed diagram with all the elements contained inside the original element. No additional settings are needed.
- **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 11.6.1.2.
- **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 15.5 (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 11.6.2) and *Edge Elements* (section 11.6.3).
- **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.

### 11.6.1.2 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 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 11.6.2), *Edge Elements* (section 11.6.3) and *Protection Devices* (section 11.6.4)
- **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 11.6.3) and *Protection Devices* (section 11.6.4)
- **Protection devices:** when this option is selected, protection devices are included into the current diagram according to the options set in the *Protection Devices* page described in section 11.6.4.
- **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.

- **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.

### 11.6.1.3 Assisted manual drawing

This action replaces the earlier *Drawing existing Net Elements* tool. Upon execution, a window will appear, listing all the elements which are not yet graphically represented in the diagram. This option is only available if the diagram is not in “freeze” mode.

#### Drawing Existing Busbars

Click on the symbol for busbars (—) in the drawing toolbox. The symbol of the busbar (terminal) is now attached to the cursor.

If the list is very large, press the button *Adjacent Element Mode* (). This activates the selecting of distance (number of elements) from elements in the selection of the *Neighbourhood Expansion*. Select the Distance of 1 in order to reduce the number of busbars (terminals) shown.

If the button *Use 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 elements part of drawn composite nodes* () 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.

The marked or selected element can now be visualised or drawn by clicking somewhere in the active diagram. This element is drawn and disappears from the list.

Note that the number of elements in the list can increase or decreases depending on how many elements are a distance away from the element lastly drawn. Scroll down the list, in case only certain elements have to be visualised.

Close the window and press **Esc** to return the cursor to normal. The drawn terminals (busbars) can be moved, rotated or manipulated in various ways.

#### Drawing Existing Lines, Switches, and Transformers

Similar to the busbars, elements like lines and transformers connecting the terminals in the substation can be drawn.

Execute the *Assisted manual drawing* action of the *Diagram Layout Tool*. For lines select the line symbol () from the drawing toolbox, for transformers select the transformer symbol (, and so on.

Similar to terminals, a list of all the lines (or transformers, or elements which have been chosen) in the network, that are not in the active diagram, is shown.

For each selected line (or transformers...) a pair of terminals, to which the line is connected, are marked in the diagram. Click on the first terminal and then on the second. The selected line is drawn and removed from the list of lines.

Continue drawing all lines (or transformers...), until the list of lines is empty or all the lines to be drawn have been drawn. 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 a first line section, then press **Tab** or double click on the diagram and arrows will appear to indicate that the line connects to a terminal that is not shown.

**Note:** Before placing elements onto the graphic users may find it useful to configure and display a background layer. This will be an image of an existing single line diagram of the system. It may be used to 'trace' over so that the *PowerFactory* network looks the same as current paper depictions; see Section 9.3.6 for more information on layers.

---

## 11.6.2 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*.
- **Draw each composite as single node:** this check box only has an impact if the corresponding grid contains composite elements (e.g. *ElmSubstat*, *ElmTrfstat*, *ElmBranch*) which graphically combine internal nodes, switches, transformers, lines, etc. If checked, the graphical representation of the composite elements are created, otherwise each of the internal elements of the composite elements is created separately in the diagram.
- **Consider physical line length:** 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.

If the option *Generate complete diagram → 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.
- **Horizontal/Vertical node spacing:** this option defines the distance between the feeder nodes, can be set to *low*, *medium* or *high*.
- **Draw each composite as single node:** as explained in the options for node insertion.
- **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.

## 11.6.3 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.
- **Insertion of one-port devices connected to substations:** this option should be checked if the option *Draw each composite as single node* is selected from the *Node Layout* page, but the user still wishes to show the one port elements (e.g. loads, shunts) connected to the composite node of the substation in the diagram.

#### 11.6.4 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*).

An example of automatically inserted protection devices is shown in figure [11.6.1](#)

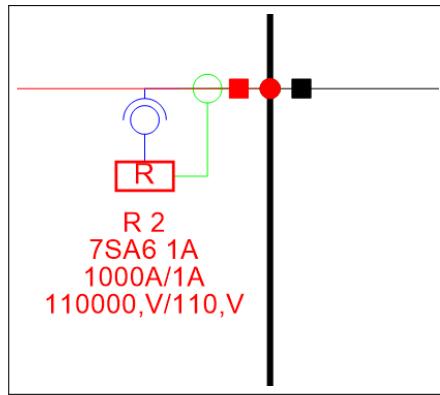


Figure 11.6.1: Protection devices in the single line diagram

### 11.6.5 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
  - Edge elements
  - Bays and sites

On the Bays and Sites page, the options to show the Bay and Site representations can be selected independently. 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.

### 11.6.6 Interchanges

The settings on the Interchanges page take effect on the following actions:

- Generate new diagram for
  - area interchange type

The options on the Interchanges page give the user some control over the positioning of elements on the graphic:

- **consider positions from current diagram** is typically used if an overview diagram is currently active, for example.
- **prefer geographic positions** will use the Geographical Coordinates of the network elements.

If nothing is selected, the layout will be determined automatically.

## 11.7 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 data object 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.

This approach may lead to problems and should therefore be used carefully.

# Chapter 12

## Network Model Manager

### 12.1 Introduction

The Network Model Manager is a browser for all calculation relevant objects. The objective of this chapter is to provide detailed information about this data management tool. Before starting, users should ensure that they are familiar with Chapter 4 (*PowerFactory* Overview).

### 12.2 Using the Network Model Manager

The Network Model Manager shows all objects relevant for the calculation. It can be accessed by clicking the button *Open Network Model Manager...*  in the main icon bar.

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 a good overview, they are sorted into groups, such as *Substations/Terminals/Switches* or *Network Components*. 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*  automatically activated. This makes it easy to edit the parameters of an object, without using the object dialog.

Figure 12.2.1 shows the Network Model Manager window, where some of the groups are collapsed and the class *Busbar* is selected.

The screenshot shows the Network Model Manager interface. On the left is a tree view of categories: All Components, Groupings (Grid, Area, Virtual Power Plant, Boundary, Zone), Substations/Terminals/Switches (Busbar, Substation, Terminal, Breaker/Switch, Site, Cubicle), Network Components (All Branch Components, Static Generator, General Load, Shunt/Filter, Synchronous Machine, Line, 2-Winding Transformer, Line Couplings), Load Flow Controller Models (Power-Frequency Controller, Station Control), and Protection Devices (Relay Model, Relay and Sub-relay, Common Time Characteris..., Directional Relay). The main area is a grid table with columns: Name, In Folder, Grid, Type, Zone, ElmZone, Area, Out of Service, System T..., Usage, Phase Technology, Nom.L-L Volt, and N. The table lists various objects like BB, BB1, SW, etc., across different regions (NE, NW, SE, SW) and areas (Northeast, Northwest, Southeast, Southwest). A filter bar at the top and tabs at the bottom (Flexible Data, Characteristics, Distributions, Basic Data, Description, Load Flow, Short-Circuit VDE/IEC, Short-Circuit Complete, Short-Circuit ANSI) are also visible.

Name	In Folder	Grid	Type	Zone	ElmZone	Area	Out of Service	System T...	Usage	Phase Technology	Nom.L-L Volt
BB	NE_04	North East		Northeast	Eastern Area			AC	Busbar	ABC	400,
BB	NW_01	North West		Northwest	Western Area			AC	Busbar	ABC	400,
BB	SE_02	South East		Southeast	Eastern Area			AC	Busbar	ABC	400,
BB1	NE_01	North East		Northeast	Eastern Area			AC	Busbar	ABC	400,
BB1	NE_02	North East		Northeast	Eastern Area			AC	Busbar	ABC	400,
BB1	NE_03	North East		Northeast	Eastern Area			AC	Busbar	ABC	400,
BB1	NW_02	North West		Northwest	Western Area			AC	Busbar	ABC	400,
BB1	NW_03	North West		Northwest	Western Area			AC	Busbar	ABC	400,
BB1	SE_03	South East		Southeast	Eastern Area			AC	Busbar	ABC	400,
BB1	SW_01	South West		Southwest	Western Area			AC	Busbar	ABC	400,
BB1	SW_02	South West		Southwest	Western Area			AC	Busbar	ABC	400,
BB1	SW_03	South West		Southwest	Western Area			AC	Busbar	ABC	400,
BB1.1	SE_01	South East		Southeast	Eastern Area			AC	Busbar	ABC	400,
BB1.2	SE_01	South East		Southeast	Eastern Area			AC	Busbar	ABC	400,
BB2	NE_01	North East		Northeast	Eastern Area			AC	Busbar	ABC	400,
BB2	NE_02	North East		Northeast	Eastern Area			AC	Busbar	ABC	400,
BB2	NE_03	North East		Northeast	Eastern Area			AC	Busbar	ABC	400,
BB2	NW_02	North West		Northwest	Western Area			AC	Busbar	ABC	400,
BB2	NW_03	North West		Northwest	Western Area			AC	Busbar	ABC	400,
BB2	SE_03	South East		Southeast	Eastern Area			AC	Busbar	ABC	400,
BB2	SW_01	South West		Southwest	Western Area			AC	Busbar	ABC	400,
BB2	SW_02	South West		Southwest	Western Area			AC	Busbar	ABC	400,
BB2	SW_03	South West		Southwest	Western Area			AC	Busbar	ABC	400,
BB2.1	SE_01	South East		Southeast	Eastern Area			AC	Busbar	ABC	400,
BB2.2	SE_01	South East		Southeast	Eastern Area			AC	Busbar	ABC	400,

Figure 12.2.1: The Network Model Manager window



Figure 12.2.2: Icon bar of the Network Model Manager

The functions of the buttons of the icon bar of the Network Model Manager shown in Figure 12.2.2 are:

1. The *Refresh* button can be used to update the Network Model Manager. This is necessary, for example, when objects are deleted in the Data Manager or the Single line diagram while the Network Model Manager is open. The deleted objects will still be displayed in the Network Model Manager until the refresh button is pressed.
2. A mouse click on the *Edit Object* button will open the dialog window of the object selected on the right hand side. A selected object is indicated by a marked row and an arrowhead next to the object symbol. Another possibility to open the edit dialog of an object is to double click on the corresponding symbol of the object.
3. One or more objects can be deleted by marking the objects to be removed and then pressing the *Delete Object* button. A query window will appear to request confirmation of the deletion (unless this has been suppressed via the relevant user setting).
4. The button *Copy (with column headers)* copies the data of a selection with the corresponding column header(s) into the Windows Clipboard. The content can then, for example, be pasted into a spreadsheet.
5. The *Detail Mode* can be activated and deactivated. If it is activated (button is pressed), the table on the right will display all the object parameters of the class (e.g. Busbars, as shown in Figure 12.2.1). The tabs at the bottom of the table give access to the same pages that are

available via the object dialog window.

If the *Detail Mode* is deactivated, a table will appear, in which columns with predefined parameters are shown. This table and the Flexible Data page (available in *Detail Mode*) can be extended, by pressing the *Variable Selection*  button.

6. By clicking on *Variable Selection* , variables/parameters to be displayed in the table can be chosen. For more information about how to handle the selection dialog, refer to Section 19.3 (Variable Selection).
7. Objects which are Out-of-Service or non-relevant for the calculation can be hidden by clicking the button *Hide Out-of-Service-Objects and Objects non-relevant for Calculation* .
8. The Network Model Manager provides the following filter functions, which are available when at least one column is filtered. How to filter columns is described in section 10.4 (Auto-Filter functions in Data Manager and browser windows).
  - If one or more columns are filtered, the button *Edit Filter*  will become accessible. After the button has been pressed, an edit dialog will open, in which a filter name can be entered. Furthermore the filters for each parameter can be modified by double clicking the relevant cell 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, one or more column filters can be consolidated to one common filter, which can then be saved under any name to reuse it, by clicking the *Save Filter*  button. A window will appear, to enable 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.

# 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-sensitive 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-sensitive 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 (see Figure 13.2.1).

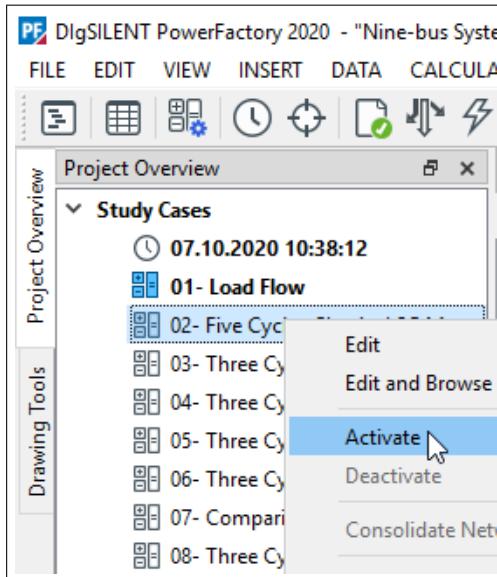


Figure 13.2.1: Activating a study case from the Project Overview window

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-sensitive 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 time trigger 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-sensitive 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-sensitive 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. 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 13.2.2.

The Study Case Manager can be accessed by clicking on the  icon on the main toolbar or from *Tools* → *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.

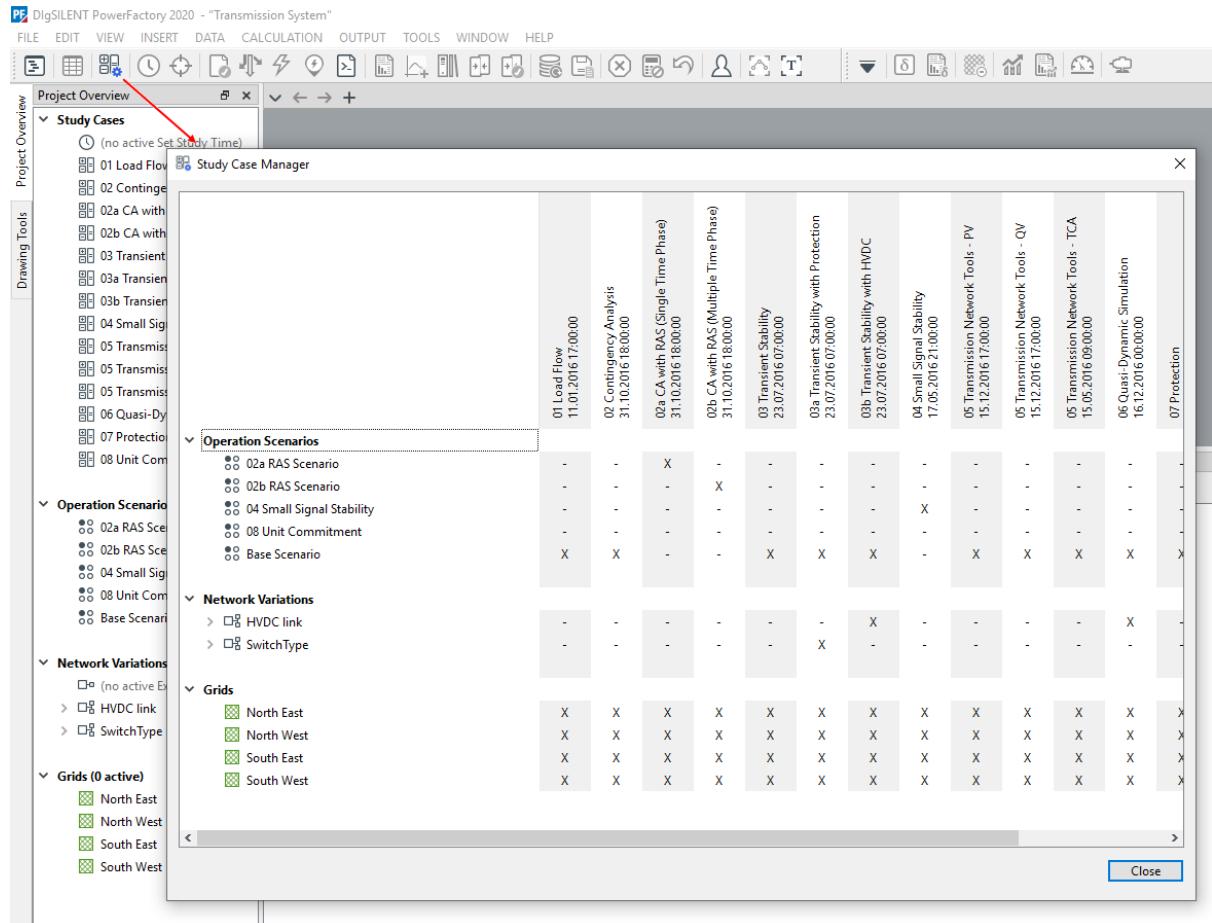


Figure 13.2.2: Study Case Manager

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-sensitive 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 8: Basic Project Definition, Section 8.1.2 (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-set 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-sensitive 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* → *Study Case* from the main menu, or right-click the study case in the Data Manager and select *Edit* from the context-sensitive menu. A dialog will appear. On the *Basic Data* page, the user can define the name and an owner of the study case. The output units of the calculated variables are defined in the *Output Variables* field. The grids that are linked to a study case may be viewed by pressing the **Grids/System Stages** 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).

The *Calculation Options* page is used to configure the basic algorithm for the study case calculations. The following options are available:

- 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. If the check box *Calculate results for all breakers* is ticked, results of reduced elements may then be post-calculated.
- 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. Since version 15.2, 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 tabs *Calculation matrices* and *Advanced* offer access to additional calculation options which tune performance and robustness of the linear system solver.
- The button **Set all calculation options to default** will restore all default options on the *Calculation Options* tab.

Please note that alteration of default options is only recommended under the supervision of the *DIGSILENT* support experts.

The *Description* page is used for user comments.

---

**Note:** To edit the study time one can alternatively press on the “Date/Time of Calculation Case” button . This will open the study case time trigger window. In addition, the time of the simulation case is displayed in the lower-right corner of the program window. Double-clicking on this field provides access to the same window.

---

## 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 (expansion stages).

Figure 13.8.1 shows a study case, 'Study 1' which contains two load flow calculation commands ( , 'Ldf 1' and 'Ldf 2'), one command for an OPF calculation , one command for the calculation of initial conditions , and one transient simulation . The dialog of each of calculation command in *PowerFactory* is described in the chapter corresponding to that calculation function.

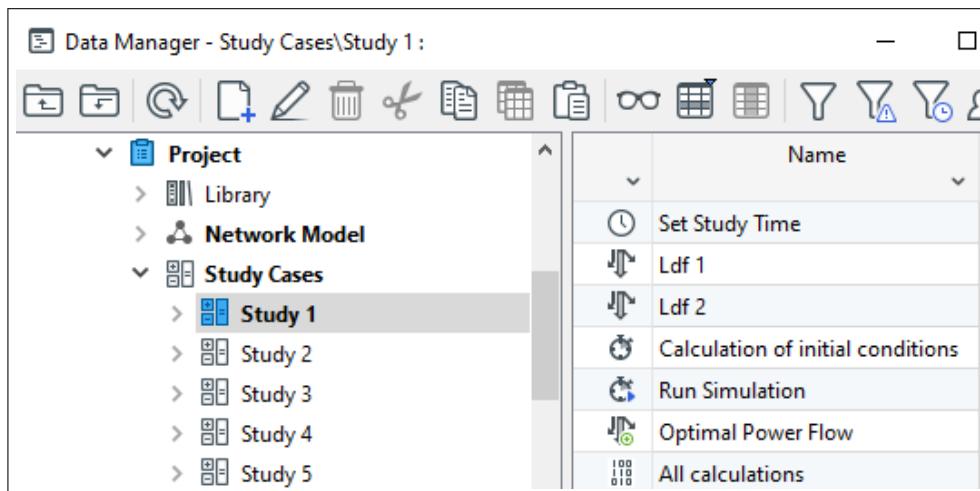


Figure 13.8.1: Calculation commands in a study case

Actions such as generating a report of the actual calculation results or the state of the defined network

---

components are carried out via commands (in this case *ComSh* and *ComDocu*, respectively). 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 29.5 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:

- Dispatch Event (*EvtGen*)
- External Measurement Event (*EvtExtmea*)
- Intercircuit Fault Events (*EvtShcII*)
- 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 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.

### 13.9.2 External Measurement Event

External measurement events can be used to set and reset values and statuses of external measurements.

### 13.9.3 Inter-Circuit Fault Events

This type of event is similar to the short-circuit event described in Section 13.9.10 (Short-Circuit Events (*EvtShc*)). 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*)

### 13.9.4 Events of Loads

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 nominal 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 nominal power of the load), over the time specified by the *Ramp Duration* (in seconds). The load ramping starts at the time of the event.

### 13.9.5 Message Event

A message will be printed to the output window at the specified time in the simulation.

### 13.9.6 Outage of Element (*EvtOutage*)

The *Outage of Element* 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!

---

### 13.9.7 Parameter Events

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-sensitive 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.

### 13.9.8 Save Results

This event is only used for *PowerFactory* Monitor applications. It cannot be used during time-domain simulations.

### 13.9.9 Save Snapshot event

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 29.9.

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).

### 13.9.10 Short-Circuit Events

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. 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.

### 13.9.11 Stop Events

Stops the simulation at the specified time within the simulation time frame.

### 13.9.12 Switch Events

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, which will show the Element Selection dialog (*IntNewobj* as shown in Figure 13.9.1). This dialog can be used to create a new switching event.

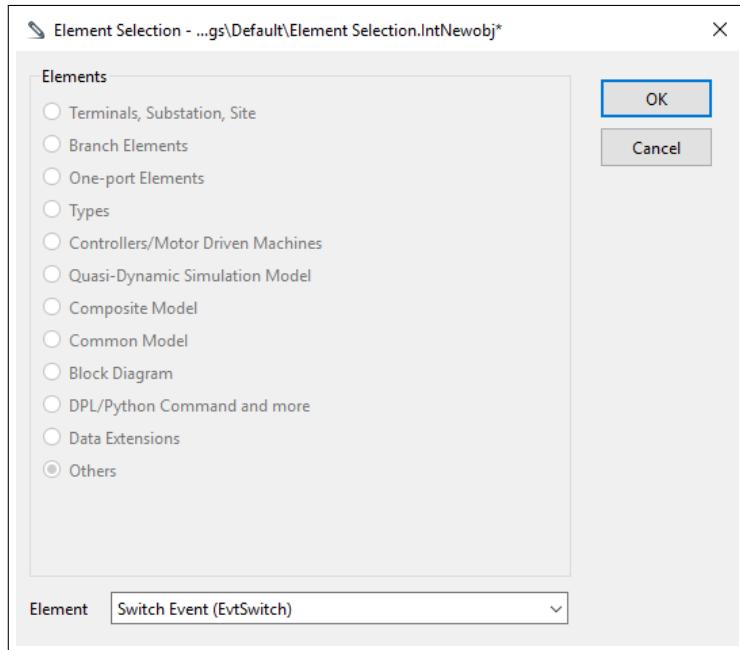


Figure 13.9.1: Creation of a new switch event (*IntNewobj*)

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.

### 13.9.13 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.

### 13.9.14 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.

### 13.9.15 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 Planning and Faults Cases for contingency analysis.

## 13.10 Simulation Scan

For details of Simulation Scan modules, refer to Chapter 29: RMS/EMT Simulations, Section 29.8.

## 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 19.6.

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*. Information regarding the use and definition of characteristics, scales and triggers is given in Chapter 18 (Parameter Characteristics, Load States, and Tariffs).

## 13.13 Graphics Board

The study case folder contains a folder called the Graphics Board folder (*SetDesktop*, ). This folder contains references to the graphics which are to be displayed. This folder, similar to the Summary Grid folder, is automatically created and maintained and should generally not be edited by the user.

The references in the graphics board folder are created when the user adds a grid to a study case.

*PowerFactory* will ask the user which graphics pertaining to the grid should be displayed. At any time, the user may display other graphics in the grid by right-clicking the grid and selecting *Show Graphic*. Graphics may be removed by right-clicking the tab at the top of the page and selecting *Close Page*.

The study case and graphics board folder also contain references to any other graphics that were created when the study case was active.

# Chapter 14

# Project Library

## 14.1 Introduction

The project library stores the following categories of data:

- **Equipment Types** (Section [14.2: Equipment Type Library](#))
- **Operational Data** (Section [14.3: Operational Library](#))
- **Scripts** (See Chapter [23: Scripting](#))
- **Table Reports** (See section Table Report Methods of the [DPL Reference](#))
- **Templates** (Section [14.4: Templates Library](#)).
- **User Defined Models** (See Section [30.3: User Defined \(DSL\) Models](#))

This chapter describes the *Equipment Type Library*, *Operational Library*, and *Templates library*. Note that in addition to the project *Library*, the global *Library* includes a range of pre-defined types, models, templates, and scripts (refer to Chapter [4: PowerFactory Overview](#), Section [4.5: Data Arrangement](#) for details).

## 14.2 Equipment Type Library

The *Equipment Type Library* is used to store and organise *Type* data for each class of network component. Once a new project is created, an *Equipment Type Library* is automatically set by the program within the *Library* folder.

To create or edit a folder in the *Equipment Type Library*:

1. On the *Equipment Type Library* folder in the left pane of the *Data Manager* right-click and select *New → Project Folder* from the context sensitive 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 these folders select the *New Object* icon  and select the appropriate type class. Alternatively, types can be copied from other projects or the global library. If the type class does not match the folder filter, an error message is displayed.

**Note:** By default new block definitions (used by dynamic models) created from block diagrams are also stored in the Equipment Types Library. Chapter 30 (Models for Dynamic Simulations) provides details related to dynamic modelling and block definitions.

Figure 14.2.1 shows the equipment library of a project containing generator, load, and transformer types, sorted using library sub-folders.

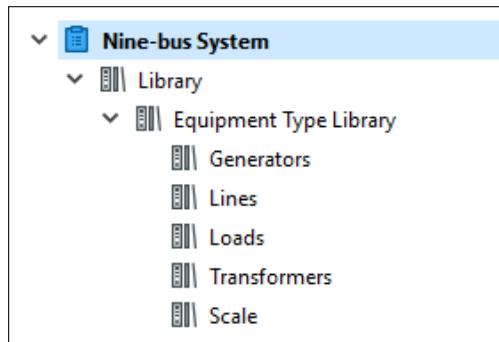


Figure 14.2.1: The Equipment Library

Unlike the “Global Library”, which is accessible to all users, the local *Equipment Type Library* is used to define types that are to be used in the specific project. It can only be used by the project owner, and users with which the project is shared.

There are three options available for defining *Type* data for network components, as illustrated in (Figure 14.2.2):

1. Select *Global Type* from the Global Library. The *Data Manager* is launched in the “Global Library”.
2. Select *Project Type*. The *Data Manager* is launched in the local *Equipment Type Library*.
3. *New Project Type*. A new type will be defined and automatically stored in the local *Equipment Type Library*.

Note that **Global Types** and **Project Types** buttons can be used to quickly switch between the global and local libraries (Figure 14.2.2).

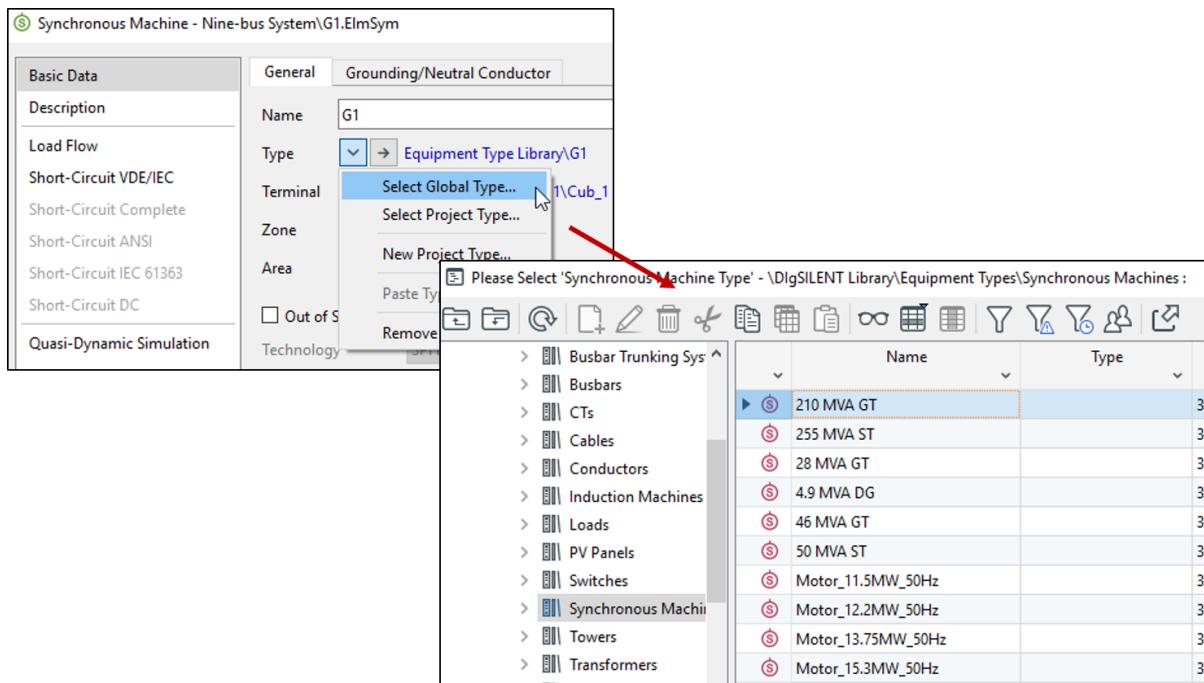


Figure 14.2.2: Selecting a Synchronous Machine Type

## 14.3 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, “G 190M-18kV Ver-1” and “G 190M-18kV Ver-2”. The Capability 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

Within the *Characteristics* folder in the *Operational Library*, the *Scale* folder is used to store the scales used by the parameter characteristics. Refer to Chapter 18 (Parameter Characteristics, Load States, and Tariffs) for details.

This section describes the definition and application of operational data objects.

### 14.3.1 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.

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 .
- In the *Element Selection* dialog 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 *Complete Short-Circuit* 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 26 (Short-Circuit Analysis). For further information about variations and expansion stages, refer to Chapter 17 (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 *StaSwitch Type* object.

---

### 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).

### 14.3.2 Characteristics

Characteristics can be assigned to many element parameters. The use of Characteristics is described in detail in Chapter 18. Such characteristics are normally held in this folder of the Operational Library.

### 14.3.3 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 14.3.7. 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. In the Element Selection dialog 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 16 (Operation Scenarios).

### 14.3.4 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 29.5: Events (*IntEvt*).

The *Faults* folder stores two types of subfolders:

1. *Fault Cases* folders, which in turn store objects that represent *Simulation Events* . *Simulation Events* 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* (*Simulation Events* or individual *Events*).

---

**Note:** The use of *IntEvt* objects extends beyond *PowerFactory*'s reliability analysis functions. Time domain simulations (EMT/RMS) make reference to *IntEvt* objects, in order to include simulation events which take place during a time-domain simulation. In this case the execution time sequence of the events must be defined. In the case of fault representations in the Operational Library by means of fault cases, only short-circuit and switching events are relevant.

---

Note that the calculation commands provided by the reliability assessment function of *PowerFactory* use Contingencies objects (*ComContingency* and *ComOutage*) to simulate the outage (and subsequent recovery) of one or more system elements. To avoid duplication of data, these objects can refer to previously defined *Simulation Events* (*IntEvt*). For information regarding the functionality of fault cases and fault groups in contingency analysis tools refer to Chapter 27 (Contingency Analysis). For the use of fault cases to create outages for the contingency analysis tools refer to Chapter 44 (Reliability Assessment).

The following sections provide a details of how to define *Fault Cases* and *Fault Groups*.

#### 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. For further information refer to Chapter 44 (Reliability Assessment).
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. For further information refer to Chapter 44 (Reliability Assessment).

To create new *Fault Cases* or new *Fault Groups* folders, open the *Faults* project folder from the *Operational Library* and use the *New Object* icon (select *Fault Cases*(*IntFltcases*) or *Fault Groups* (*IntFltgroups*) respectively).

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 *Define* → *Fault Cases* from the context-sensitive menu.
3. Select from the following options:
  - **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.
  - **Multi 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.

This menu entry is only active if more than one component is selected. The fault case is automatically created in the database after selection.

- **Multi 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. This menu entry is only active if more than one component is selected. If only one component is selected, then no fault case will be created. The fault case is automatically created in the database after selection.
- **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 ...).

---

## 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. In the edit dialog, specify the name of the *Fault Group*, and Add Cases (*IntEvt*) to the *Fault Group*.

### 14.3.5 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 *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* object 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) generator 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 about 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. Operational Limit and the Max. Active Power Rating specified on the Synchronous Machine *Load Flow* page.

---

## 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. Enter data points to define the generation limits, and *Append Rows* to add the required number of rows to the table.
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.

#### Defining a Variation of a Capability Curve

Similar to circuit breaker ratings (see Section 14.3.1 (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).

#### 14.3.6 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 14.3.1 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.

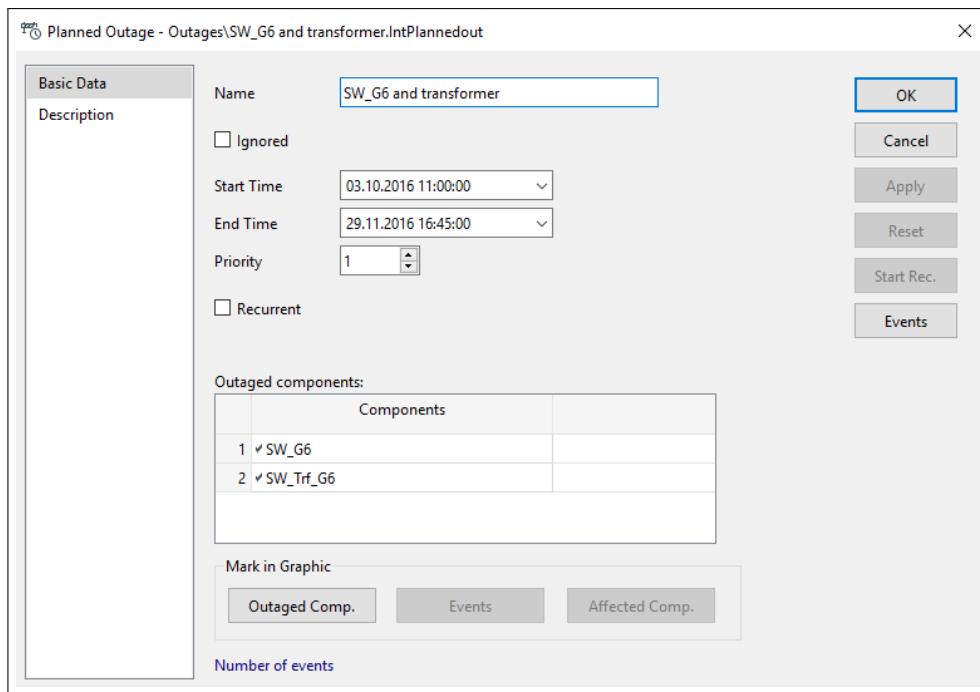


Figure 14.3.1: Planned Outage object dialog

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 [42](#) for how to create Planned Outage objects and handle them via the toolbar.

### 14.3.7 Planned Outages *IntOutage*

Note that this subsection refers to the original *IntOutage* object, which is now superseded by the *IntPlannedout* object described in section [14.3.6](#). 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.

## 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.

## 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 deenergised (if circuit breakers are open). A deenergised 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 deenergised and not earthed, but should be outaged.
- Components that are deenergised and earthed, but should NOT be outaged.
- Components that are deenergised, not earthed and should be outaged.
- Generators that are not derated, but should be outaged.
- Generators that are derated, but should NOT be outaged.

### 14.3.8 QP-Curves

Q(P)-curves (*IntQpcurve*) are used by generators (*ElmSym*, *ElmGenstat* and *ElmAsm*) 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 of the machine.

### 14.3.9 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.

### 14.3.10 Running Arrangements

*Running Arrangement* objects  store operational data (switch status) for a single substation. As shown in Figure 14.3.2, 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.

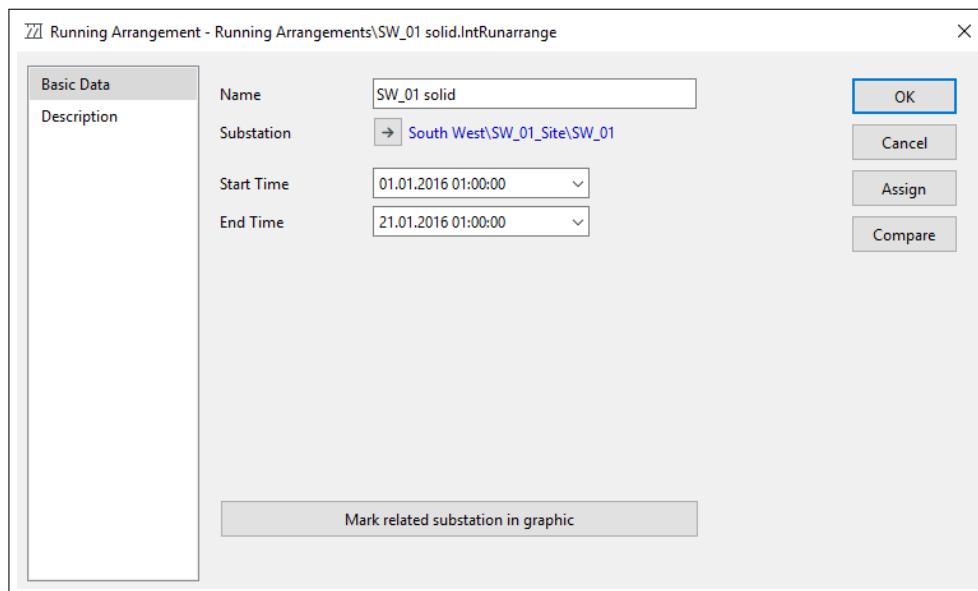


Figure 14.3.2: RA object dialog

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.

**Note:** Running arrangements store only the status of switches of class *ElmCoup*. The status of switches which are automatically created in a cubicle following the connection of a edge element (*StaSwitch* objects) are not considered in a running arrangement.

Further details of how to create, select, apply, and assign Running Arrangements are provided in the following sections.

### 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-sensitive 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.

### 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-sensitive 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).

### 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).

### Assigning a Running Arrangement

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-sensitive 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.

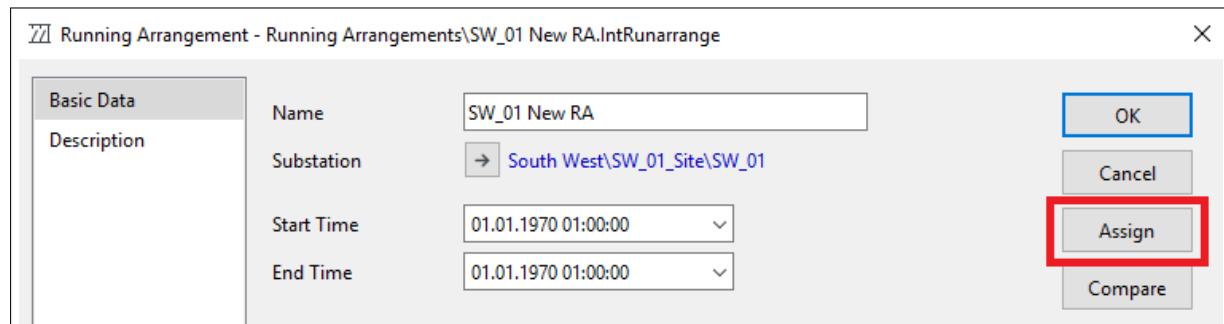


Figure 14.3.3: Running Arrangement Dialog

### Marking Running Arrangements in Graphic

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-sensitive menu displayed when right-clicking on a Running Arrangement in a Data Manager.

#### 14.3.11 Thermal Ratings

*Thermal Ratings* objects (IntThrating) allow the definition of post-fault operational ratings for certain branch elements, depending on the fault duration and the loading prior to the fault. *Thermal Ratings* objects are stored in the *Thermal Ratings* folder in the *Operational Library*. They are two-dimensional

matrices, with cells that contain the “short time” post-fault ratings (in MVA), according to the pre-fault loading (defined in the first column) and the duration of the fault/overloading (defined in the first row).

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*).
- 2- and 3-winding transformers (*ElmTr2*) and (*ElmTr3*).
- Series reactors (*ElmSind*).
- Series capacitors (*ElmScap*).

Note that the rating table given on the *Ratings* page of the *Thermal Rating* object (when option *Consider short term ratings* is enabled) is used solely for the contingency analysis command in *PowerFactory*. In this calculation, the pre-fault loading conditions of the network components are determined after a base load flow calculation. The contingency analysis is then performed using a load flow command, where the post-contingency duration is specified.

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. To configure the table for the short-term ratings (only visible if the option *Consider short term ratings* is checked), go to the *Configuration* page and:
  - Introduce the increasing values for the pre-fault loading axis (Prefault %). By default, values between 0% and 80%, with increments of 5%, up to 84% are set.
  - Introduce the fault duration in minutes. Default values are: 360 min, 20 min, 10 min, 5 min, 3 min).

The pre-fault continuous rating (used as the base to calculate the loading before the fault) and the post-fault continuous rating (assumed as the branch element post-fault rating if the fault duration is larger than the largest duration time defined in the table) are defined on the *Ratings* page.

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).

### 14.3.12 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.

## 14.4 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. Five kinds of templates are supported in *PowerFactory*:

1. Element template for single network elements: New single network elements with the same parameters as the original element are created.
2. Group template for non-composite graphic objects: New groups of objects (including graphical attributes) are created.
3. Substation template (composite node): New substations with the same configuration as the original substation (including its diagram).
4. Secondary Substation template: New secondary substations.
5. Branch template (composite branch): New branches with the same configuration as the original branch (including its diagram).

Templates are normally stored in the *Templates* folder, in the *Library*. When a template for a single network element is defined, a copy of the original element is automatically created in the *Templates* folder. New templates of substations and branches will copy the objects together with all of their contents (including the diagram) to the *Templates* folder. New templates for groups of objects will copy the corresponding objects, together with their graphical information to a subfolder for groups of class *IntTemplate*  within the *Templates* Library.

For further information about working with templates, refer to Section 11.2 (Defining Network Models with the Graphical Editor).

Substation (*composite node*) templates ( or ) , secondary substation () , busbar templates (), branch templates () , and general templates ( ) can be selected from the Drawing Toolbox on the right-hand pane of the *PowerFactory* GUI. To apply an element template:

- Select the symbol for a substation, secondary substation, busbar, branch, or general template as required.
- Select the required template.
- Insert the new element in the single line graphic.

---

**Note:** The use of Substation templates is recommended for diagrams of networks, where components are grouped in branches and substations. In this case the composite nodes can be graphically connected with the composite branch, forming an overview diagram of the complete network.

---

This section explains how to create and use templates.

#### 14.4.1 General Templates

Any kind of single network component (lines, transformers, terminals, etc.) can be used to define an “Element” template; this is done by right clicking the desired element on a single line graphic and selecting *Define Template* from the context sensitive menu, a dialog where the name of the new template is to be written pops up. After the name is given and the **Ok** button is pressed, a copy of the selected element is stored in the templates folder. Similarly, a group of network components can be used to define a “Group” template, which will create a ‘template’ folder, storing the objects from the group together with their graphical information. If a group of elements containing substation and branches has been selected the elements outside the substation will not be added to the template.

## 14.4.2 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 sensitive 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).

---

## 14.4.3 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 sensitive 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.

## 14.4.4 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 sensitive 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 10: Data Manager, Section 11.5.4 (Defining Composite Branches in the Data Manager).

## 14.4.5 Example Power Plant Template

Consider the following example, where there is a power station with multiple transformers, generators, and control systems of the same type. The model can be created using templates as follows:

1. Firstly, define type data for the transformer, generator, and control system.

2. Add a single instance of the generating unit (including generator transformer) to the network model.
3. Define a *Template* by selecting the generator, generator bus, and transformer, then right-click and select *Define Template*. Optionally include the control system model with the template.
4. To create another instance of the newly created template, select the *General Templates* icon () and place it on the single line graphic.

#### 14.4.6 Wind Turbine Templates according to IEC 61400-27-1

There are predefined Templates for Wind turbine models according to IEC 61400-27-1 in the Templates Library of *PowerFactory*. More information is available in section *Templates* of the [Technical References Document](#).

# Chapter 15

## Grouping Objects

This chapter describes the management and functionality of the objects used to group network components.

### 15.1 Areas

To facilitate the visualisation and analysis of a power system, elements may be allocated into areas (*ElmArea* ). The single line graphics 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 () in the *Network Data* directory.

To define a new 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 *Define → Area* from the context sensitive menu.
- After the area has been defined, terminals can be added to it by selecting *Add to... → Area...* in their context sensitive menu.

In the edit dialog of the new area you must 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 a single line diagram.

---

**Note:** Areas that are created/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.

---

For information concerning the visualisation of areas within the single line Graphic refer to Chapter 9: Network Graphics, section 9.3.5.1 (Basic Attributes). For information about reporting Area results refer to Chapter 19 (Reporting and Visualising Results).

## 15.2 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.

### 15.2.1 Defining and Editing a New Virtual Power Plant

A new Virtual Power Plant is created by:

- Multi selecting in a single line diagram or in a data browser an initial set of generators to be included in the Virtual Power Plant;
- Then pressing the right mouse button and selecting *Define* → *Virtual Power Plant* from the context sensitive menu.

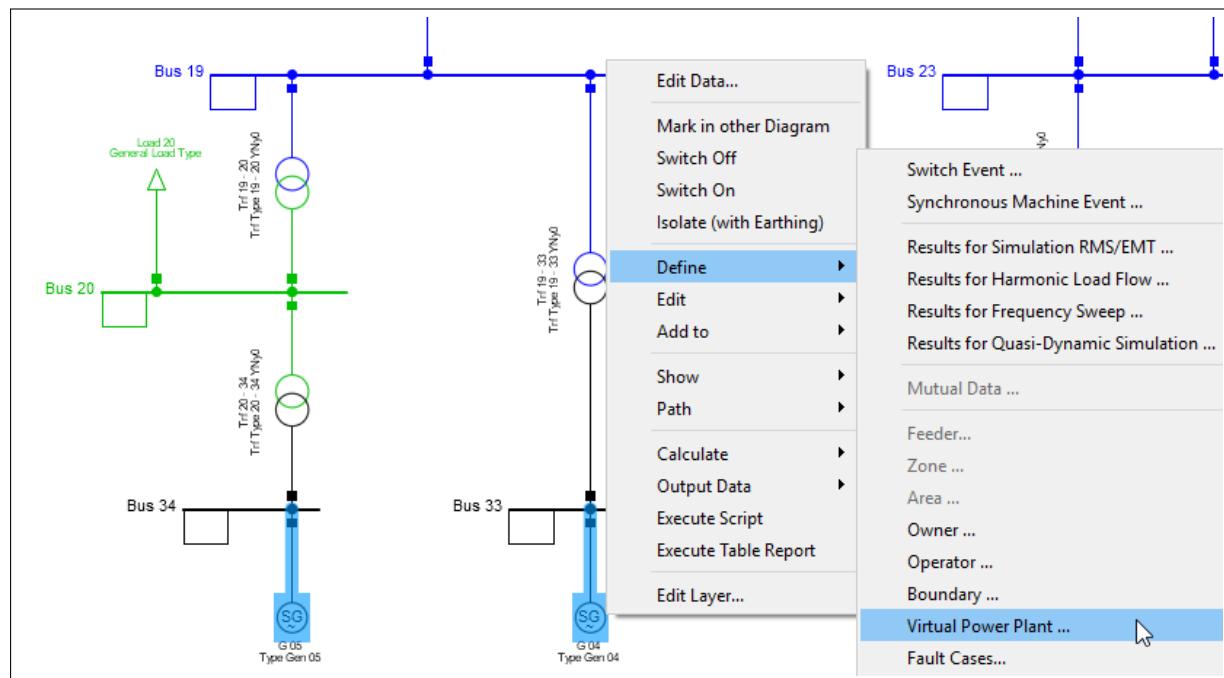


Figure 15.2.1: Defining a Virtual Power Plant

Alternatively you can create a new empty Virtual Power Plant by using the Data Manager:

- Open a Data Manager.
- Find the Virtual Power Plant folder (⌚) and click on it.
- Press the icon for defining new objects (➕).
- select *Others*.
- Then select *Virtual Power Plant (ElmBmu)* in the list box.
- Assign a suitable name to the Virtual Power Plant.

- Press **OK**.

The rules which 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'. The dispatch of the belonging generators (variable "pgini" from the Load Flow tab 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 pops up. Otherwise the dispatch is set according the user 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 Optimisation tab 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.

---

**Note:** The Virtual Power Plant active power is part of the operation scenario subsets and therefore is stored in the active operation scenario (if available). The active power is stored in the active expansion stage (if available) if no active operation scenario is active. Virtual Power Plants that are created/deleted when a recording expansion stage is active; become available/non available only if the corresponding variation is active and the expansion stage activation time is earlier than the current study time.

---

### 15.2.2 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.

### 15.2.3 Inserting a Generator into a Virtual Power Plant and Defining its Virtual Power Plant Properties

Generators are added to an existing Virtual Power Plant by adding a reference in the 'Optimisation' tab of their edit dialog. Notice that a generator can belong to at most one Virtual Power Plant. Define the Merit Order and must run properties as required.

You also can add a generator to a Virtual Power Plant by clicking with the right mouse button on the element in the network graphic and choose *Add to... → Virtual Power Plant...* from the context sensitive menu.

## 15.3 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* or directly on the branch elements by right clicking on them and selecting *Define → Boundary*....

The Boundaries are stored in the project folder *Boundaries* (grid icon) within the *Network Data*.

**Note:** Boundaries 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.

### 15.3.1 Boundary Definition Tool

The Boundary Definition Tool (grid icon) is located within the *Additional Tools* toolbox as shown in figure 15.3.1.

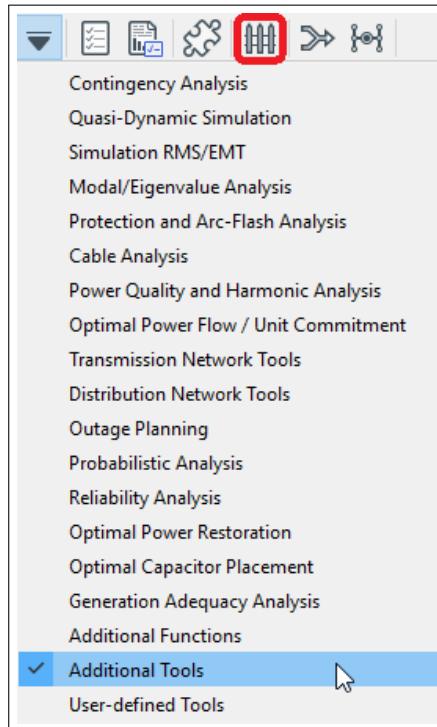


Figure 15.3.1: Additional Tools toolbox

The following options are available when using the Boundary Definition Tool command:

#### 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 finish 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.

### 15.3.2 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 sensitive 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 sensitive 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.

#### 15.3.2.1 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.)

## 15.4 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.

---

## 15.5 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 Definitions (*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 (➡) in the *Network Data* folder.

A new Feeder is created by right-clicking on a cubicle (that is, when the cursor is held just above the breaker in the single line diagram) and selecting *Define → Feeder...*. Once the option Feeder has been selected, the Feeder dialog pops up. There you can define the desired options for the new object. After pressing **Ok**, the new Feeder is stored in the Feeders folder of the Network Model.

Any existing Feeder can be edited using its dialog (double click the target Feeder on a data browser). 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.

**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 select all the elements of a Feeder in the desired single line diagram. 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 9: 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 *Define → Feeder...* from the context sensitive menu that appears. This step is illustrated in Figure 15.5.1.
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.

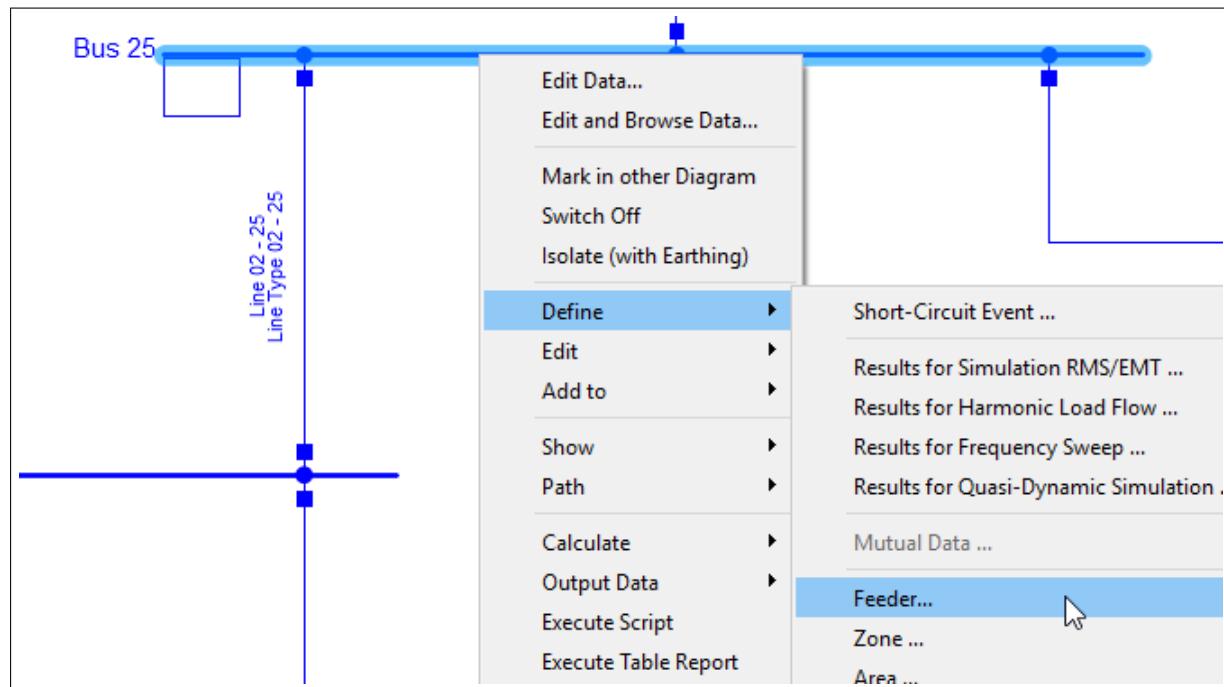


Figure 15.5.1: Definition of Feeders from a terminal by right-clicking the terminal

**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.

### 15.5.1 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 [41](#)

---

#### 15.5.1.1 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.

#### 15.5.1.2 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.

---

#### 15.5.1.3 Feeder Tool Command

Feeder Tools is a built-in command (*ComFeedertool*) in *PowerFactory* and can be started via the right-mouse context-sensitive menu, by clicking on an element of a feeder as shown in Figure [15.5.2](#). A radial feeder must be defined prior to using the command.

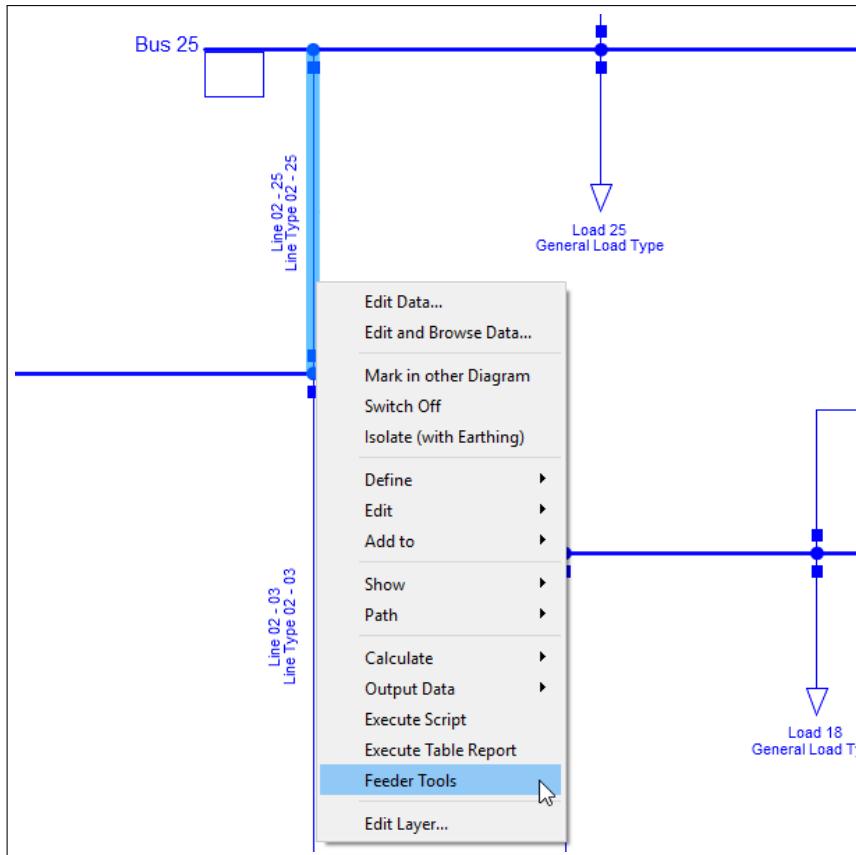


Figure 15.5.2: Feeder Tool

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.

#### 15.5.1.4 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 15.5.3. 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).

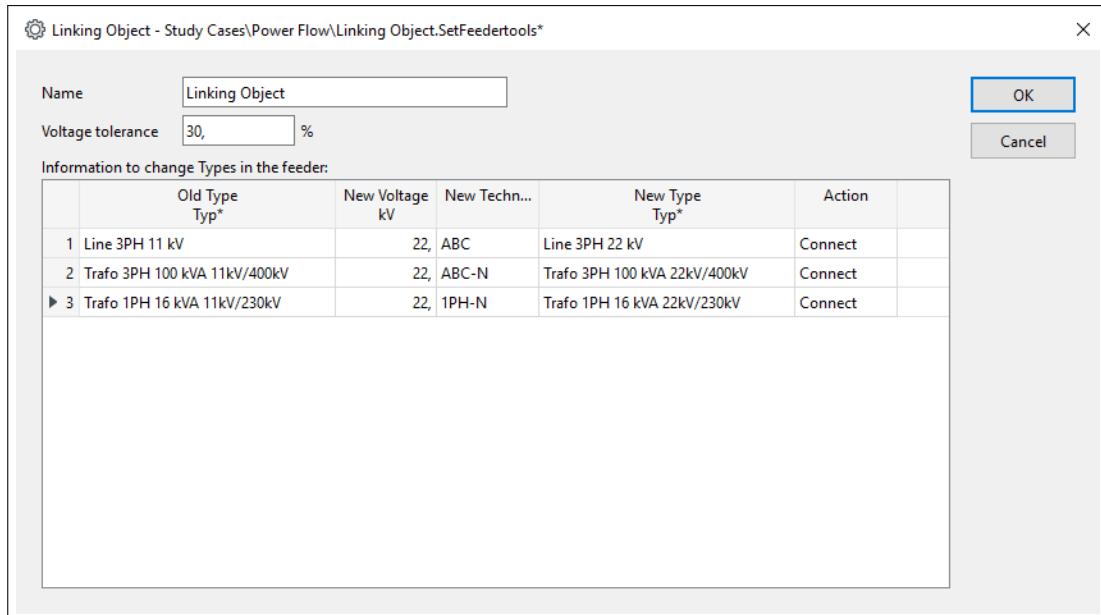


Figure 15.5.3: Linking Object

## 7. Validation rules for types:

- a. 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*).
- b. 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.
- c. 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.
- d. 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.

## 15.6 Meteo Stations

It is often the case that *groups* of wind generators have a wind speed characteristic that is correlated. *PowerFactory* can represent such a correlation through the *Meteo Station* (*ElmMeteostat*  object). The meteorological stations are stored within 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 46.4), or via the single line diagram by right-clicking on an appropriate element and selecting *Define* → *Meteo Station* (or *Add to* → *Meteo Station*) from the context-sensitive menu. Note that the ability to define a *Meteo Station* 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.

---

## 15.7 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 objects. The Operator objects (*ElmOperator*,  ) are stored in the *Operators* folder ( ) in the *Network Model* directory.

To create a new operator:

- In the Data Manager open the Operators folder from the Network Model.
- Click on the 'New Object' icon.
- The edit dialog of the new operator pops up:
  - Give a name to the new object.
  - Select a colour to represent the operator in the corresponding colouring mode of the single line diagram.
  - Press **Ok**.

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 tab of their dialog.

---

**Note:** Operators 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

---

## 15.8 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 Model* 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 tab of their dialog.

---

**Note:** Operators 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

---

## 15.9 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 *Path* → *New* from the context sensitive 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 Model*.

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 *Path* → *Add* to from the context sensitive 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 *Path* → *Remove Partly* from the context sensitive menu. The Remove option of the Path context sensitive menu 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 9: Network Graphics. For information about the use of the path definitions for the analysis of the protective devices, refer to Chapter 33 (Protection).

**Note:** Paths 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

## 15.10 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](#)).

## 15.11 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.

Zones are created by selecting a node or multi-selecting elements (at least one node-element has to be among them), right-clicking and choosing *Define → Zone...* from the context sensitive menu. The option *Add to → Zone...* can be selected when a zone(s) have already been defined. Single-port elements are directly assigned to the zone, its connected node is part of. For multi-port elements (like lines or transformers) one of the available terminals has to be chosen, from which the zone assignment is inherited.

# 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 four 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.

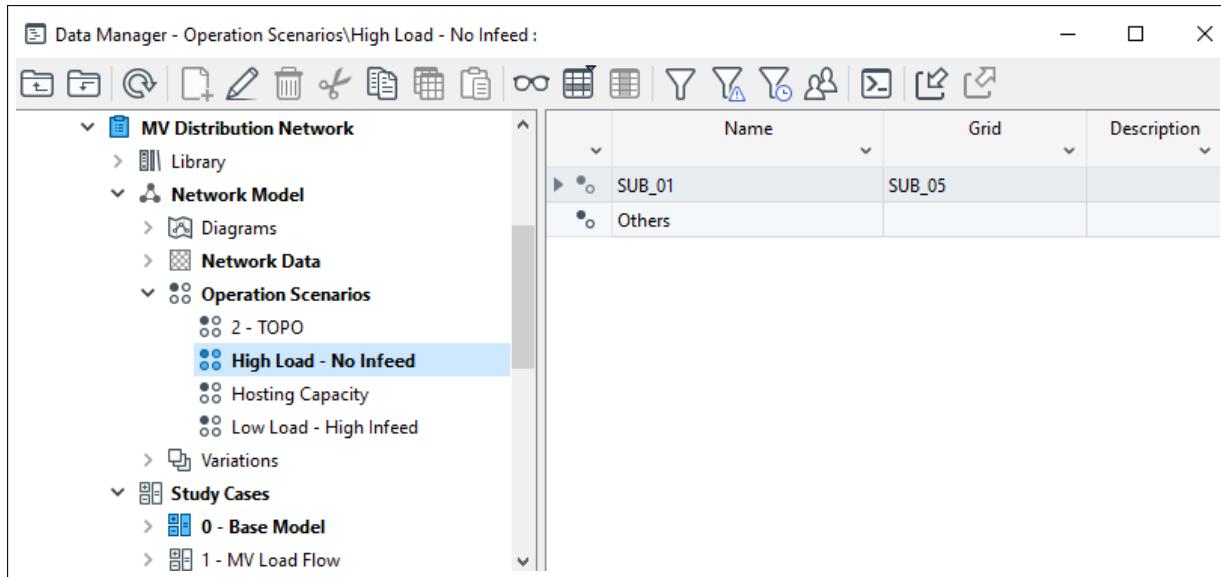


Figure 16.2.1: Operation Scenarios and Subsets

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-sensitive menu as shown in Figure 16.3.1.  
The dialog of the new operation scenario pops up.

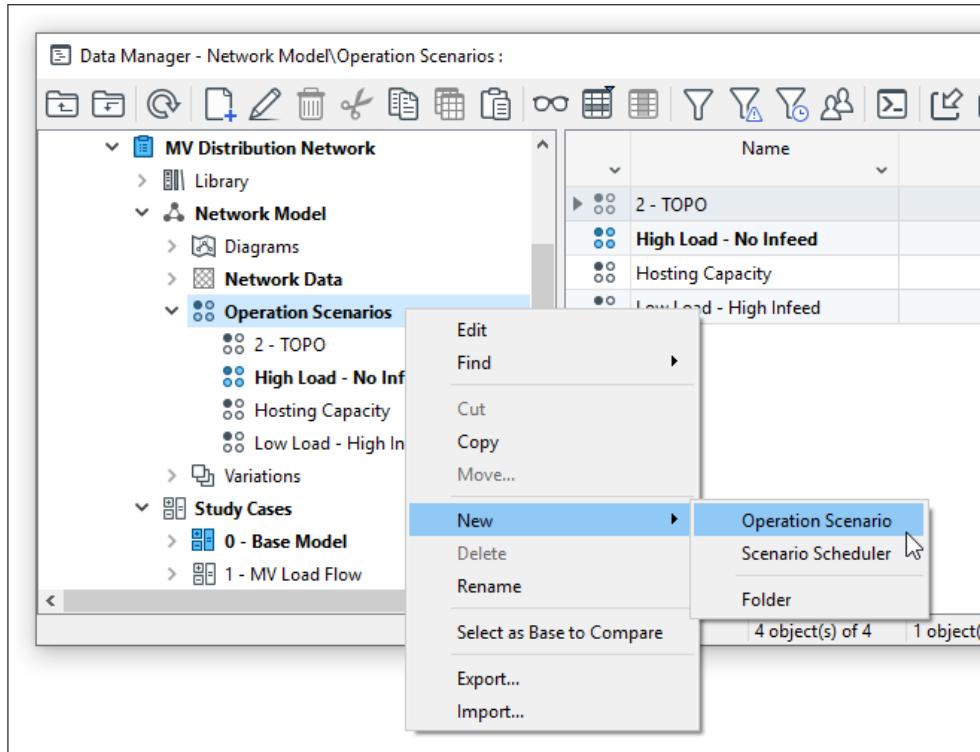


Figure 16.3.1: Creating a new operation scenario object using the Data Manager.

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 16.5.1.

### 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.

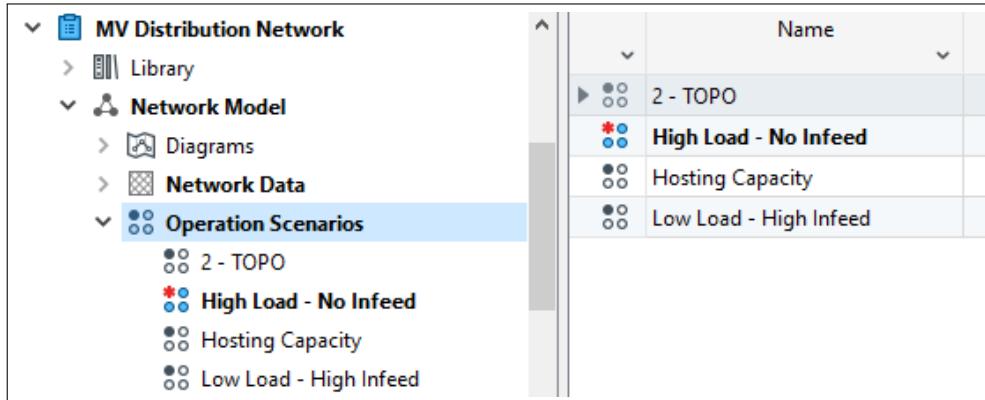


Figure 16.3.2: An asterisk indicates unsaved changes in operation scenarios

### 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-sensitive menu (right mouse button) entry *Action -> Save* of the operation scenario (see Figure 16.3.3).

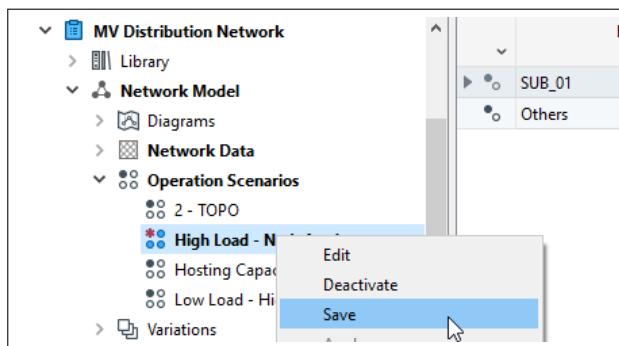


Figure 16.3.3: Saving an operation scenario using the context-sensitive menu

**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 sensitive 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 sensitive 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.

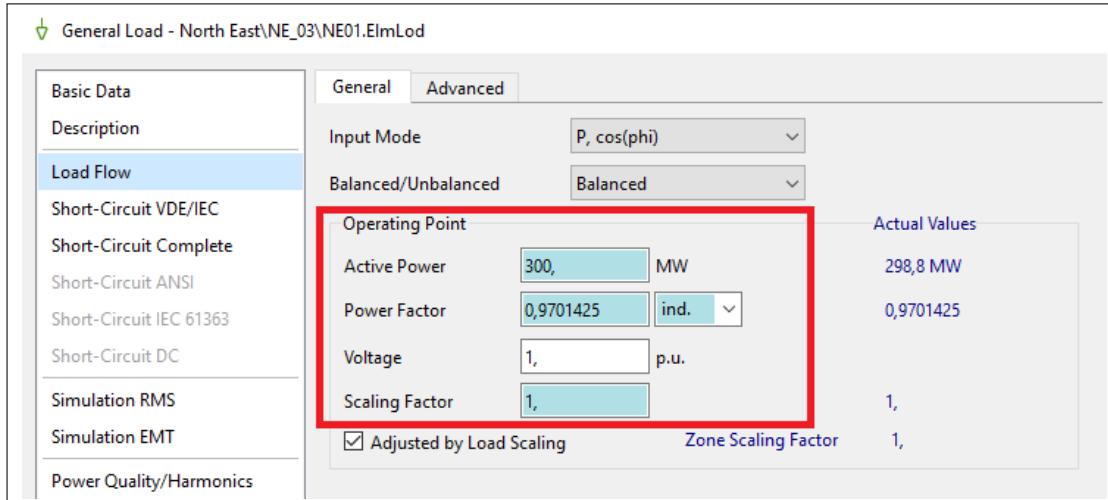


Figure 16.3.4: Blue highlighted operational data in an element dialog

Name	In Folder	Grid	Input M...	Balanc...	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 Administering Operation Scenarios

In this sub-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.

#### 16.4.1 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 sensitive 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.

---

#### 16.4.2 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 sensitive 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.

#### 16.4.3 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 sensitive 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-sensitive menu when right-clicking on an operation scenario (*Action* → ...).

---

#### 16.4.4 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*.

#### 16.4.4.1 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 sensitive 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.

#### 16.4.4.2 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 sensitive 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.

#### 16.4.4.3 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 (SetOpdselection)*.

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".

#### 16.4.5 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 sensitive 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.

#### 16.4.6 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.

### 16.4.7 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  A object selection window will appear.
3. From the *Element* drop down menu choose the 'Scenario Scheduler' (IntScnsched).
4. Press **OK**. The scenario scheduler object dialog will appear as shown in Figure 16.4.1. Give the scheduler a name.

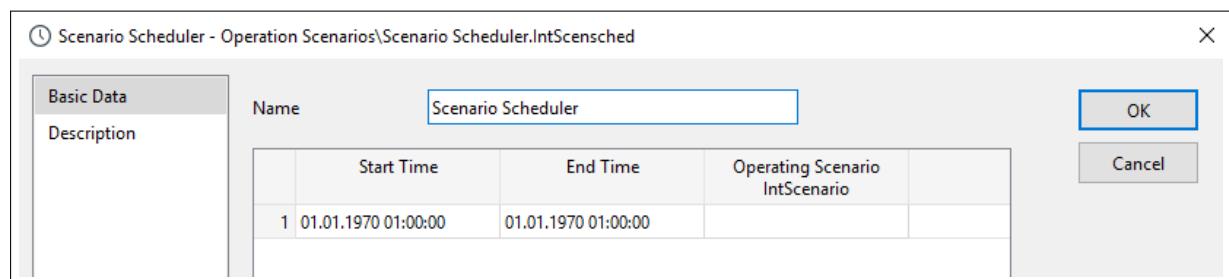


Figure 16.4.1: The Scenario Scheduler (IntScnsched) dialog

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 sensitive 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.

---

## 16.5 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.

### 16.5.1 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 a 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 ().

---

### 16.5.2 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 Scenario Configuration folder object 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. Open the scenario configuration folder 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 red icons in the Data Manager.
  - 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:
  - 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 dark red 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-sensitive menu select *New* → *Variation*.
  - In a Data Manager, select the *Variations* folder and click on the *New Object* icon (⊕). Ensure that the *Element* field is set to *Variation (IntScheme)*, and press **Ok**.
2. Define the Variation *Name*.
3. Optionally set the Variation *Colour*. This is used to highlight modifications introduced by the Variation in the Single Line Graphic.
4. On the second page of the *Basic Data* tab, 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-sensitive 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 (*IntStage*):

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 icon bar. Set the ‘Element’ field to *Expansion Stage (IntStage)* and press **Ok**.
2. Define the Expansion Stage *Name*.
3. Set the Expansion Stage 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 Chapter 49 (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 17.8.3).
- 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 17.8.1).

## 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 Calculation 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*, or *Edit → Project Data → 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 “Set 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 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 (IntScheduler)*.
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.

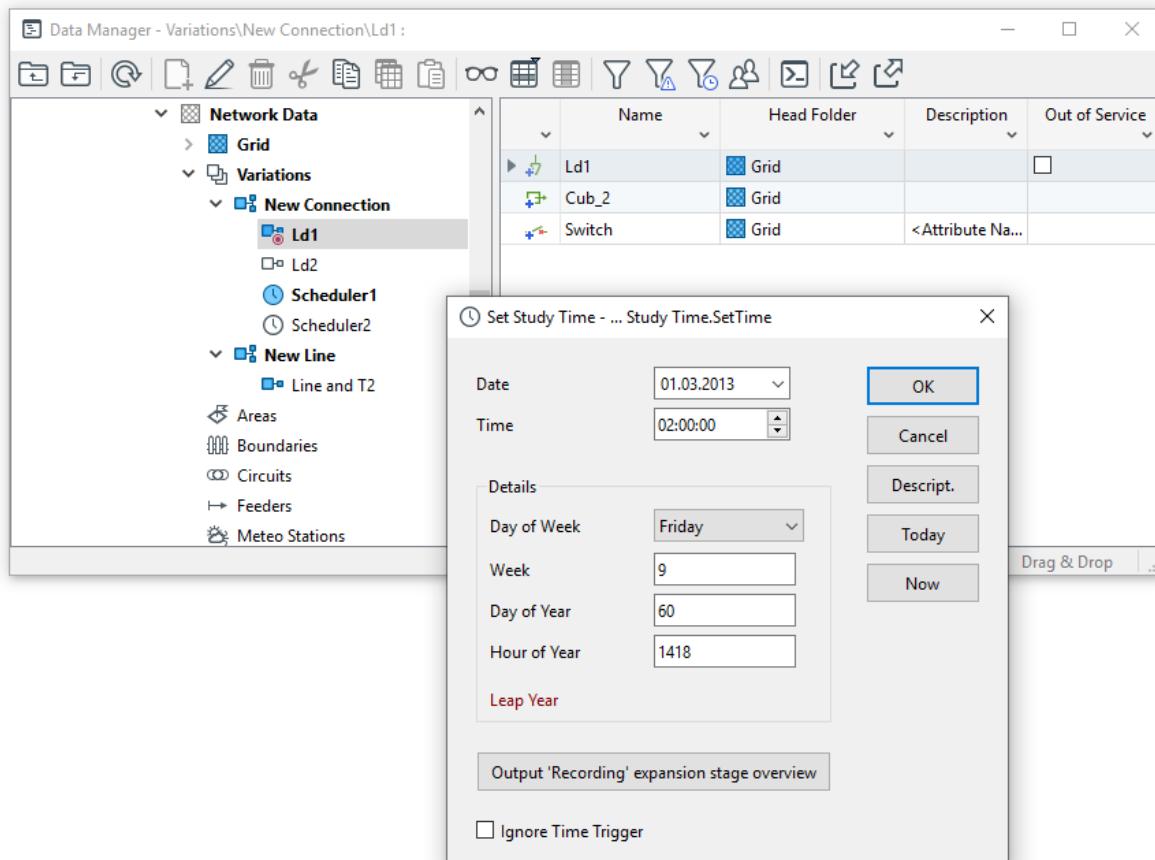


Figure 17.7.1: Example Variations and Expansion Stages - Data Manager

Figure 17.7.2 shows the Single Line Graphic of the associated network. Since the Expansion Stage “Ld2” is inactive, the Load “Ld2” is shown in yellow.

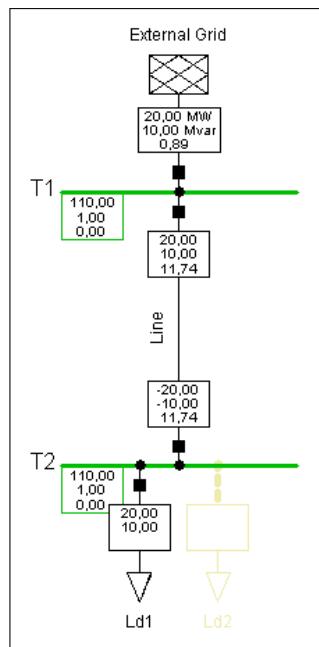


Figure 17.7.2: Example Variations and Expansion Stages - Single Line Graphic

## 17.8 Variation and Expansion Stage Housekeeping

### 17.8.1 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.

### 17.8.2 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-sensitive 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-sensitive menu select *Consolidate*.

---

### 17.8.3 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-sensitive 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.

### 17.8.4 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. Comparison results symbols indicate the differences between each listed object, defined as follows:

- **-** The object exists in the “base object” but not in the “object to compare”.
- **+** The object exists in the “object to compare” but not in the “base object”.
- **△** The object exists in both sets but the parameters’ values differ.
- **=** The object exists in both sets and has identical parameter values.

To compare two Variations:

1. In an active project, right-click on a non-active Variation and from the context-sensitive menu select *Select as Base to Compare*.
2. Right-click on the (inactive) Variation to compare and from the context-sensitive menu select *Compare to “Name of the base object”*.
3. The Merge Tool dialog (*ComMerge*) is displayed. By default, all of the contained elements are compared. The *Compare* fields can be configured however, to compare only the objects or selected subfolders.
4. Once the *Compare* options are set, press the **Execute** button.
5. When prompted, select **Yes** to deactivate the project and perform the comparison.

Figure 17.8.1 shows an example comparison of two Variations (based on the example presented in Section 17.7), where the Variation “New Line” is set as the “Base” for comparison. The “Assigned from” options are set such that all Expansion Stages from both “New Line” and “New Connection” Variations will be merged into a single Variation, which will retain the name of the “Base” Variation “New Line”.

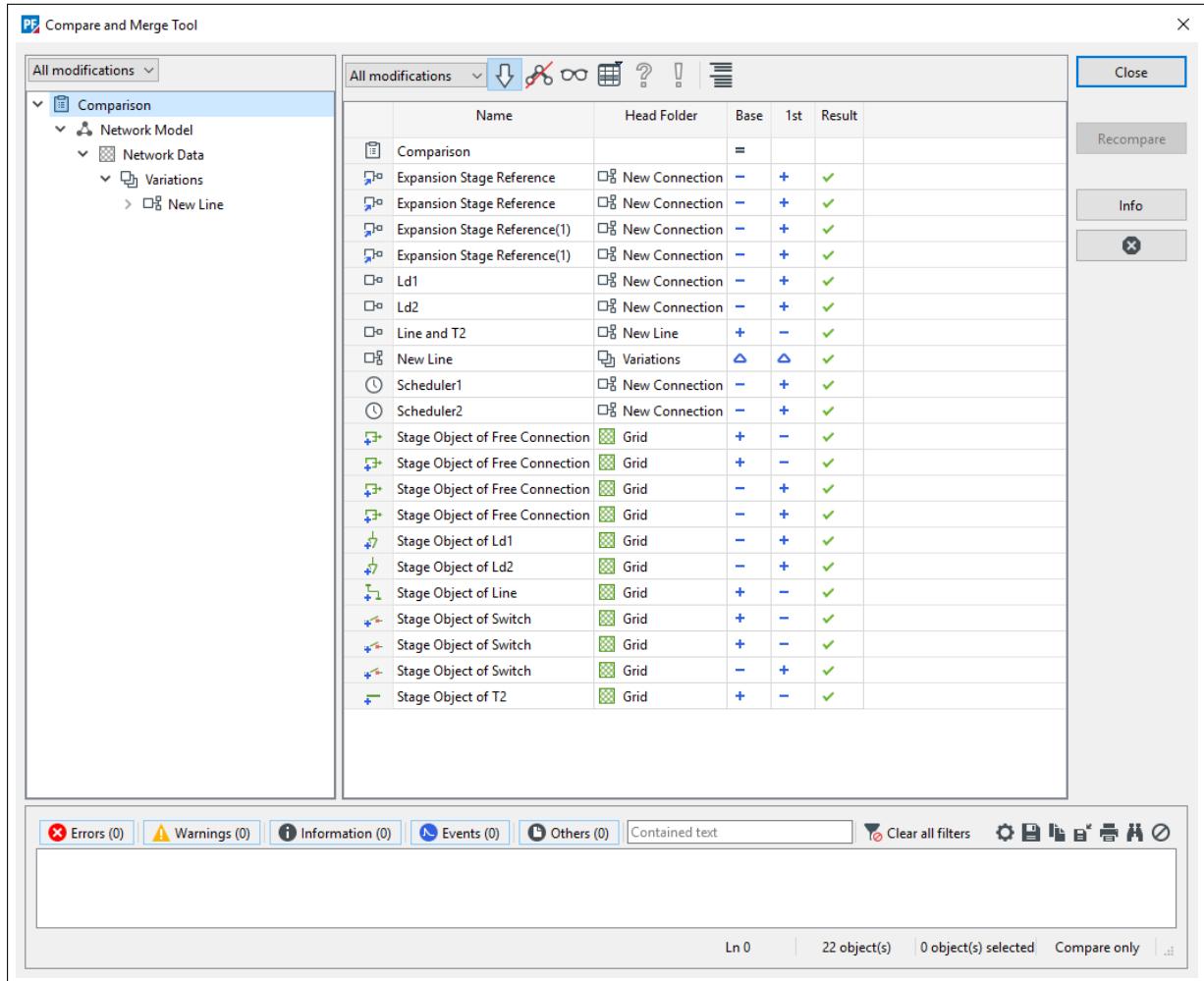


Figure 17.8.1: Merge Tool Window

Refer to Chapter 21: Data Management, Section 21.4 (Comparing and Merging Projects) for further details on use of the Merge Tool.

### 17.8.5 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.

### 17.8.6 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.

### 17.8.7 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 (*ElmLine*) of length 1km. When Variation 1 is recorded, the length of the line is updated from the base case value to a new value of 10km. 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 15km not 10km. 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 10km 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 → Show Project Overview Window*). Then, by Right clicking in the Project Overview Window on the title line of the Network Variations section. A contextual menu as illustrated in Figure 17.8.2 will appear. The option Error Correction Mode should be selected from the contextual menu.

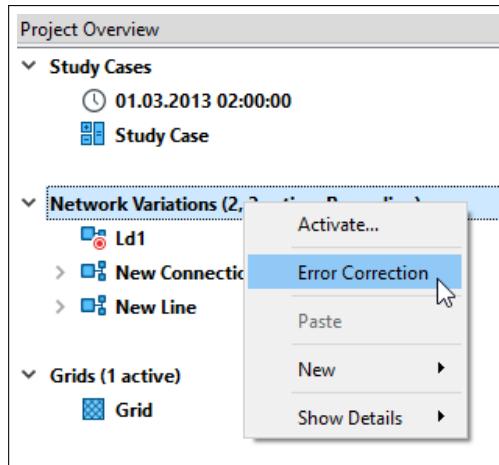


Figure 17.8.2: Activating Error Correction Mode

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 1km to 10km. The 10km value will be updated to 15km. 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 *ElmLine* 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.

## 17.9 Compatibility with Previous *PowerFactory* Releases

### 17.9.1 General

Prior to *PowerFactory* v14, “System Stages” were used to analyse design alternatives as well as different operating conditions. They recorded model changes (addition/removal of equipment, topology changes, etc.), operational changes (switch positions, tap positions, generator dispatch, etc.), and graphical changes. Since version 14.0, the System Stage definition has been replaced by Variations and Operation Scenarios, which provides enhanced flexibility and transparency.

When importing (and then activating) a project that was implemented in a previous *PowerFactory* version, the activation process will automatically make a copy of the project, rename it (by appending \_v14 or \_v15 to the project name) and migrate the structure of the copied project.

The migration process creates new *Project Folders* (such as Network Data, Study Cases, Library folders, etc.) and moves the corresponding information to these project folders. Additionally, existing *Stations* and *Line Routes* elements are migrated to their corresponding definition in v14 and v15 (i.e. Substations and Branches).

If the project contains *System Stages*, they **will not be converted automatically**. They will be still be defined, and functions related to their handling will still be available. If the user wishes to take full advantage of the *Variation* and *Operational Scenario* concepts, then the System Stages must be converted manually. The procedure is described in the following section.

### 17.9.2 Converting System Stages

The conversion process of System Stages is described with reference to an example project opened in *PowerFactory v14*, with the structure shown in Figure 17.9.1. The project contains three grids “Grid 110 kV”, “Grid 220 kV” and “Grid 33 kV”. Each Grid contains a “2010 Base Case” System Stage with three System Stages “2010 MAX”, “2010 MIN”, and “2011 Base Case”. The “2011 Base Case” stage in-turn contains two stages, “2011 MAX” and “2011 MIN”. The Study Cases are configured so that the “2011 MAX” Study Case and the “2011 MAX” stages are active.

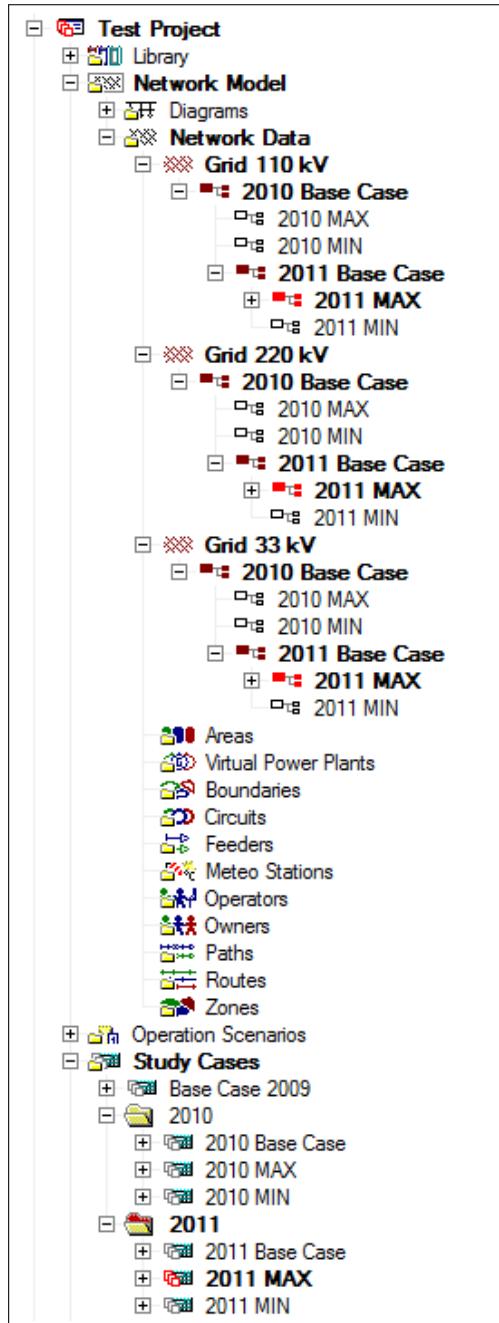


Figure 17.9.1: Example Project - System Stage Structure

To convert the System Stages to Variations / Operation Scenarios:

1. Activate the *Study Case* that uses the base grids (in this example “Base Case 2009”), so that no System Stage is active.

2. Create a *Variations* folder inside the *Network Data* folder by opening the *Data Manager* window and from the left pane select the *Network Data* folder (located inside the *Network Model* folder), right-click and select *New → Project Folder*. In the dialog window that appears, type in a name (for example “Variations”) and select “Variations” as the folder type. Press **OK**.
3. Define a *Variation* inside the Variations folder. From the *Data Manager* window select the Variations folder, right-click and select *New → Variation*. In the dialog window that appears, type in a name (for example “2010”). Press **OK**, and select **Yes** to activate the new Variation.
4. The Expansion Stage dialog will be displayed. Type in a name and set the activation time as appropriate (in this case, it is set to 01.01.2010). Press **OK**, and select **Yes** to set the stage as recording. After this step, the Variation should be active and the Expansion Stage be recording.
5. From the Data Manager, select a Study Case that uses System Stages (it should not be active), right-click and select *Reduce Revision*. This will copy both network data and operational data from the System Stages used by the study case into the recording Expansion Stage, and will delete the System Stages (to copy operational data to an Operation Scenario, an Operation Scenario must be active at this step). In this example, the “2010 Base Case” is reduced, followed by the “2011 Base Case” - this is because the complete System Stage branch, containing all System Stages between the selected stage and the target folder are reduced. Figure 17.9.2 shows the result of reducing the “2010 Base Case” and “2011 Base Case” to Variations.

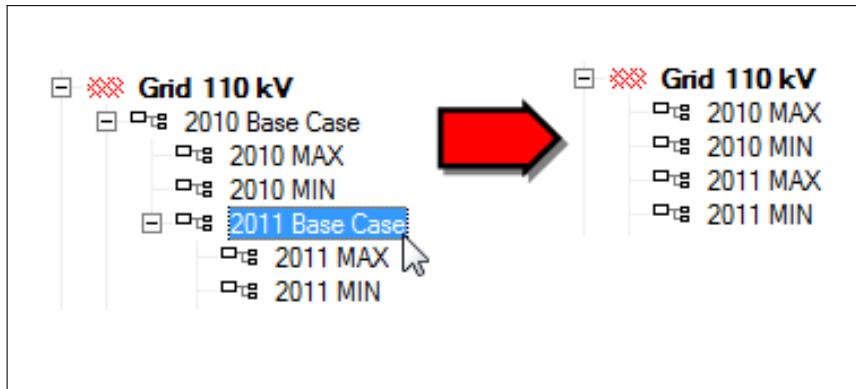


Figure 17.9.2: Reduce Revision performed for the 2011 Base Case

6. After converting System Stages “2010 Base Case” and “2011 Base Case” (with Network Data modifications) to Variations, and System Stages “2010 MAX”, “2010 MIN”, “2011 MAX”, and “2011 MIN” (with operational modifications) to Operation Scenarios, the Variations and Operation Scenarios are assigned to Study Cases. Figure 17.9.3 shows the resulting project structure for this example, where all System Stages have been converted to Variations and Operation Scenarios.

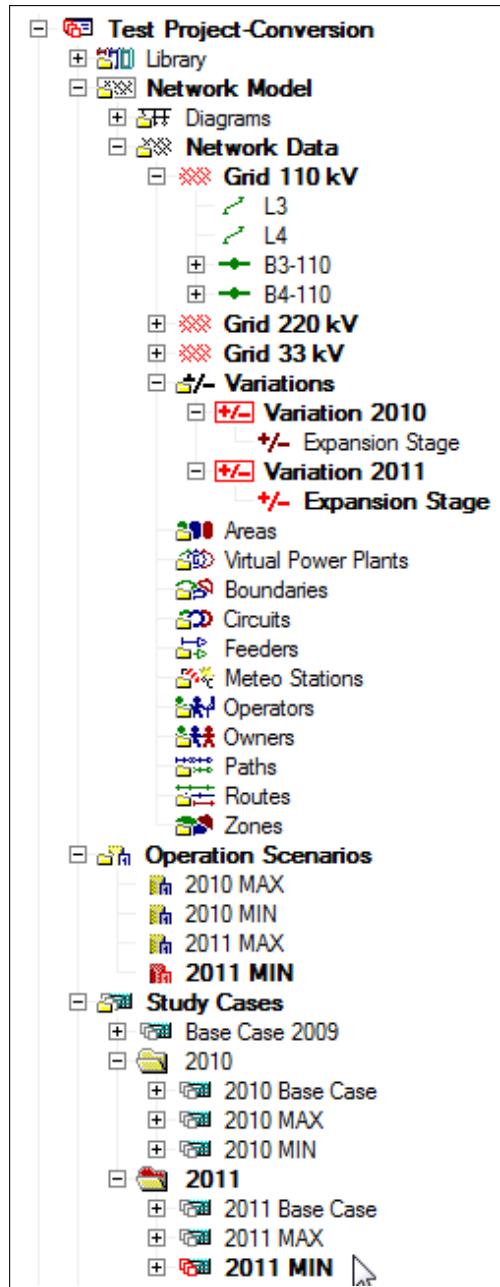


Figure 17.9.3: Resulting Project Structure

# 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

### General Background

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*.
- The Global *Characteristics* folder within *Database → Library*.

Studies which utilise characteristics are known as 'parametric studies'.

## 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*, ). After a new scale has been defined, a trigger is automatically created in the active study case folder (see also Chapter 13, Section 13.12: Triggers). 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.

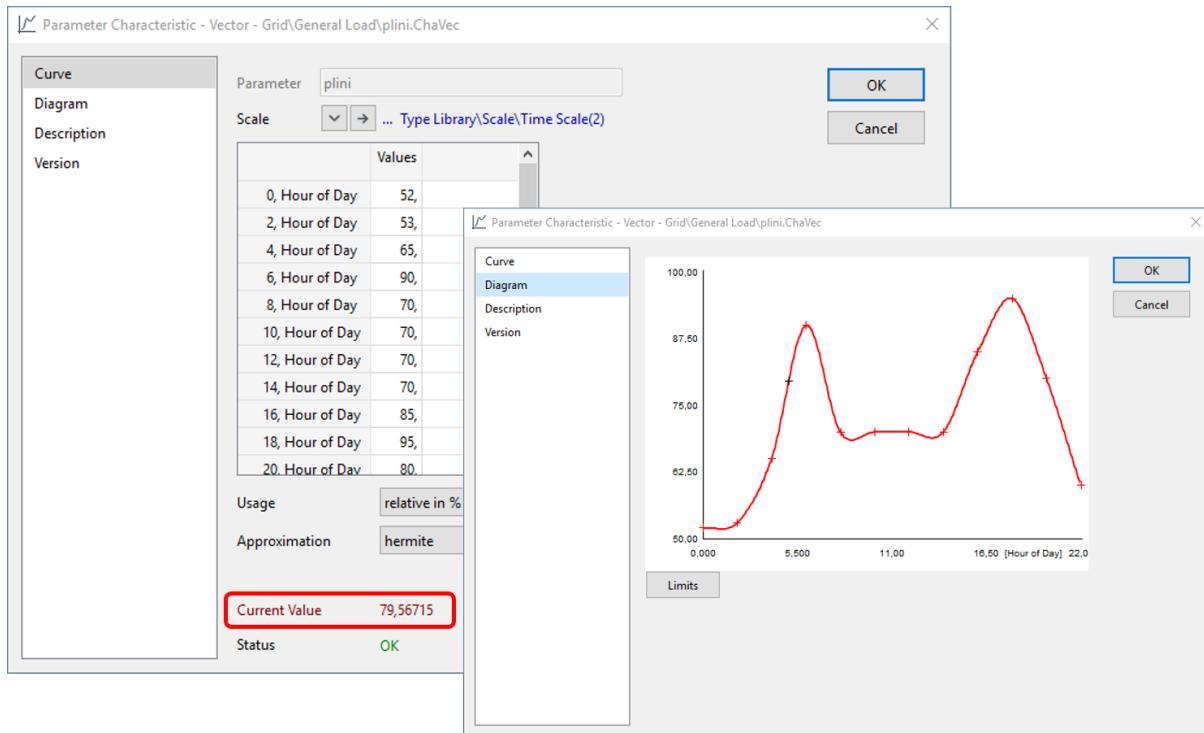


Figure 18.2.1: Illustration of Scales and Triggers

## 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
<a href="#">18.2.1:</a> 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.
<a href="#">18.2.2:</a> Profile Characteristics	Parameter(s) are to be modified according to seasonal variation and the day, week and month set in the Study Time.
<a href="#">18.2.4:</a> Scalar Characteristics	Parameter(s) are to be manually modified by a scaling factor.
<a href="#">18.2.5:</a> Vector Characteristics with Discrete Scales	Discrete parameter states are to be selectable.
<a href="#">18.2.5:</a> Vector Characteristics with Continuous Scales	Parameter states may be interpolated between entered values.
<a href="#">18.2.5:</a> Vector Characteristics with Frequency Scales	Parameter(s) are to be modified with Frequency.
<a href="#">18.2.5:</a> Vector Characteristics with Time Scales	Parameter(s) are to be modified based on a user-defined scale referencing the Study Time.
<a href="#">18.2.6:</a> Matrix Parameter Characteristics	Parameter states are based on two variables, and may be interpolated between entered values.
<a href="#">18.2.7:</a> Parameter Characteristics from Files	Parameter states and the trigger (optional) is to be read from a file.
<a href="#">18.2.8:</a> Characteristic References	Reference link between a parameter and a Characteristic

Table 18.2.1: Summary of Parameter Characteristics

## Usage

With the exception of the Scalar Characteristic, 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.

## 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.

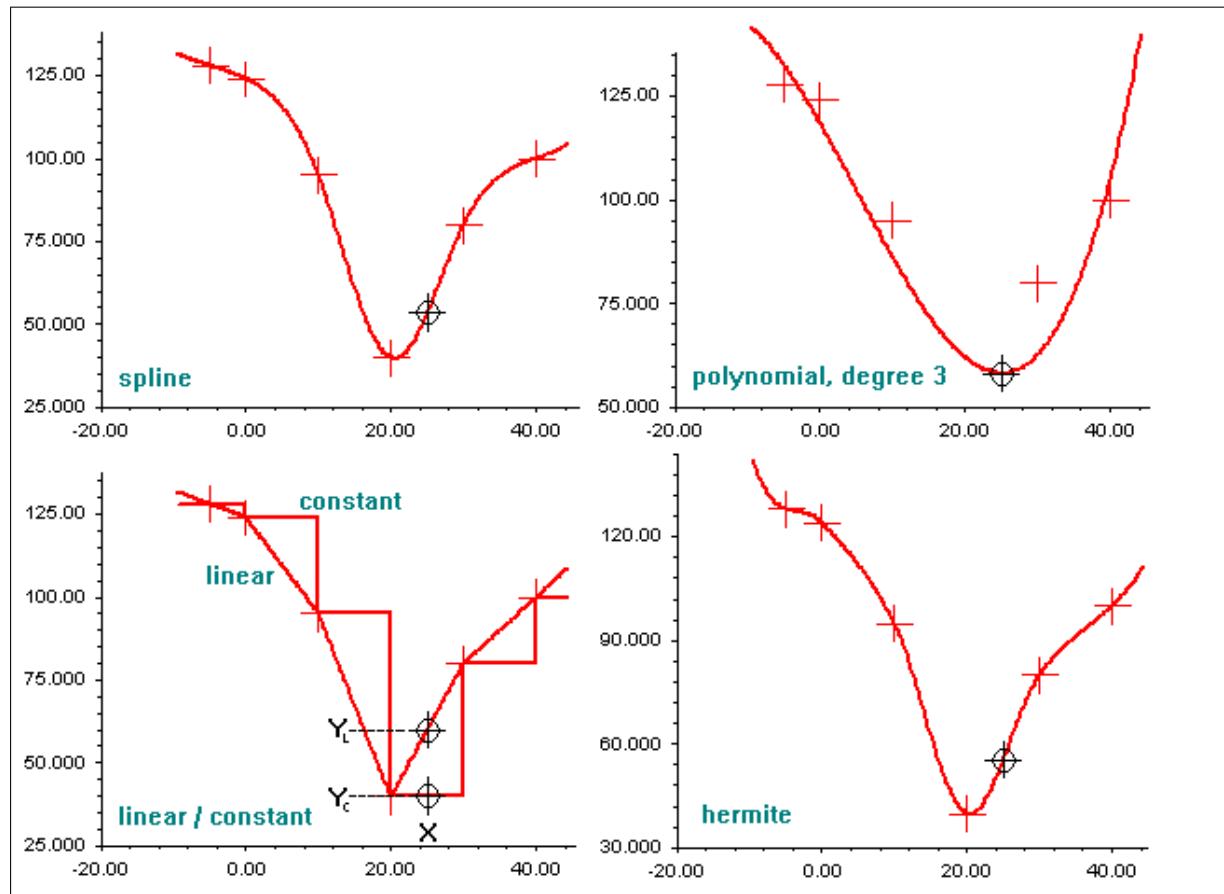


Figure 18.2.2: Approximated characteristics

Note that Approximation methods are not available for discrete characteristics.

### 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.

#### 18.2.1 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*). The time characteristic (*ChaTime*) uses an internally defined Recurrence period that is convenient to define a periodically recurring characteristic. The user simply selects a Recurrence and enters the corresponding values. The Recurrence values available are:

- Daily
- Weekly
- Monthly
- Yearly
- None

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.

### 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**.

### 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 'Unit' of time or 'Time Stamped Data' format, 'Time Column' and 'Data Column' and 'Column separator' and 'Decimal separator'.
- Define the 'Usage' and 'Approximation'.
- Press **OK**.

### 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**.

### 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.

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 supported.
  - Enter the access details (Username and password) for the database system.
  - Enter the database name.
- Two *Table modes* are offered, as illustrated in figure 18.2.3 below:
  - "Value Column" is used when individual database tables exist for individual element characteristics. In this case the user must provide the Table name, the Time Column name and the Value Column name.
  - "Id Column" is used if there is a common table, where data for more than one characteristics is stored. In this case, the table will have an additional column for id-references, and so the column name and id must be supplied.
- 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**.

Value Column

Timestamp	Active Power	Reactive Power
2019-01-01 00:00:00	11.167	3.350
2019-01-01 00:15:00	11.145	3.344
2019-01-01 00:30:00	11.132	3.340
.....		

Id Column

Timestamp	Id	Active Power	Reactive Power
2019-01-01 00:00:00	Load01	11.167	3.350
2019-01-01 00:00:00	Load02	2.449	0.735
2019-01-01 00:00:00	Load03	30.488	9.146
.....			
2019-01-01 00:15:00	Load01	11.145	3.344
2019-01-01 00:15:00	Load02	2.300	0.690
2019-01-01 00:15:00	Load03	28.034	8.410
.....			
2019-01-01 00:30:00	Load01	11.132	3.340
2019-01-01 00:30:00	Load02	2.130	0.639
2019-01-01 00:30:00	Load03	27.001	8.100
.....			

Figure 18.2.3: Database table modes

**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)").

### 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.

### 18.2.2 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**.

### 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.

---

**Note:** All daily and yearly characteristics must be relative. No absolute-value characteristics are permissive

---

### 18.2.3 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**.

### 18.2.4 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**.

### 18.2.5 Vector Characteristics

Vector Characteristics may be defined with reference to *Discrete Scales*, *Continuous Scales*, *Frequency Scales*, and *Time Scales*.

#### 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:
  - 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** twice.
- Define the ‘Usage’ 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.

### 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.

### 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.

### 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 's', 'm', and 'h' 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 definition procedure is similar to that of the continuous characteristics, although the Time Scale (*TriTime*) scale is selected.

## 18.2.6 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.

### 18.2.7 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 *New Characteristic* → *Characteristic from File...*
- 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.

### 18.2.8 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.

### 18.2.9 Edit Characteristic Dialog

Once a parameter has a characteristic defined, then an option to *Edit characteristic(s)* becomes visible on the parameters context sensitive 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.

### 18.2.10 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 18.2.4 (remember that the browser must be in “detail” mode to see these tabs). Note also that the data colouring indicates that characteristics are applied.

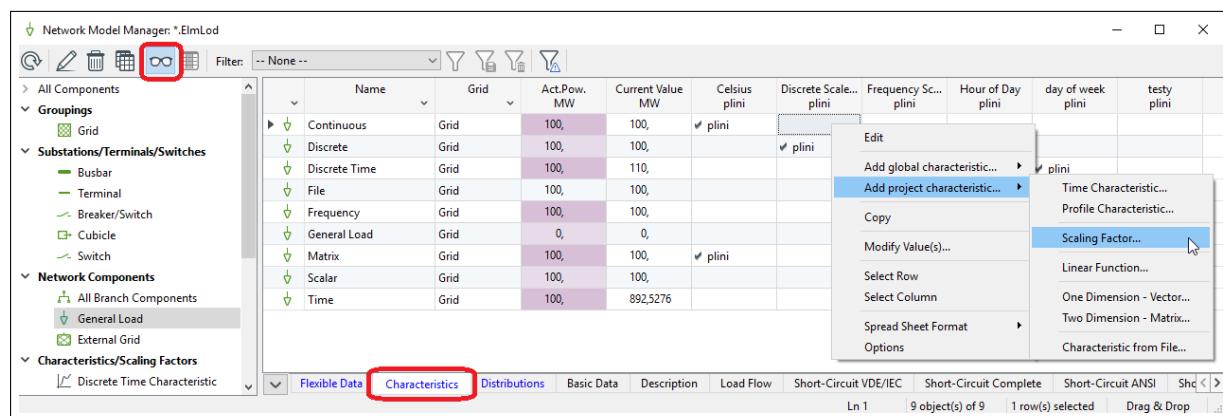


Figure 18.2.4: Network Model Manager Characteristics tab

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* → *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.

### 18.2.11 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 Profile 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 18.2.5.

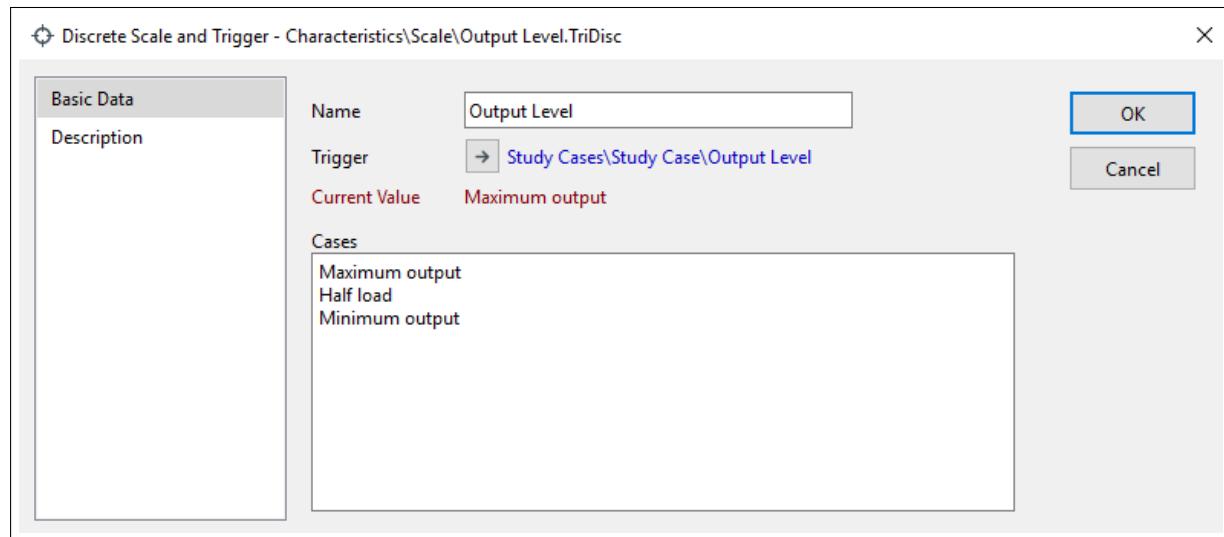


Figure 18.2.5: Active Power Discrete Scale and Trigger

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 18.2.6 shows the resultant vector characteristic, including a reference to the Scale “Output Level” and the current parameter value.

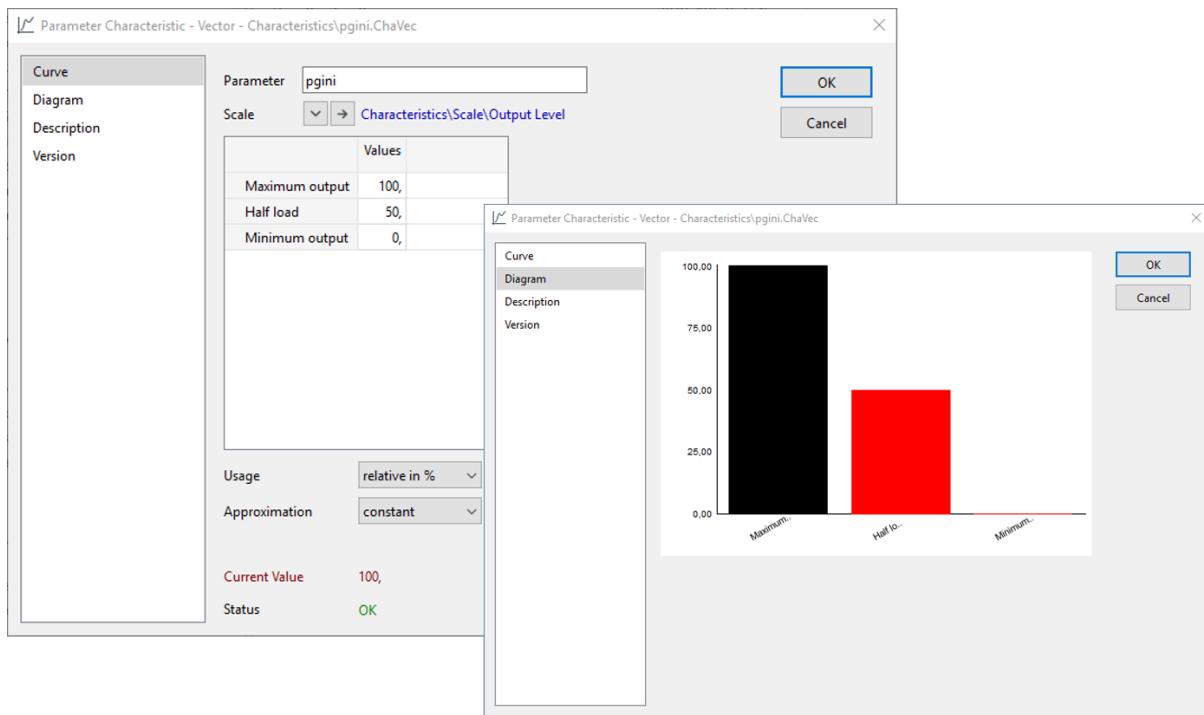


Figure 18.2.6: Active Power Parameter Characteristic

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 18.2.7.

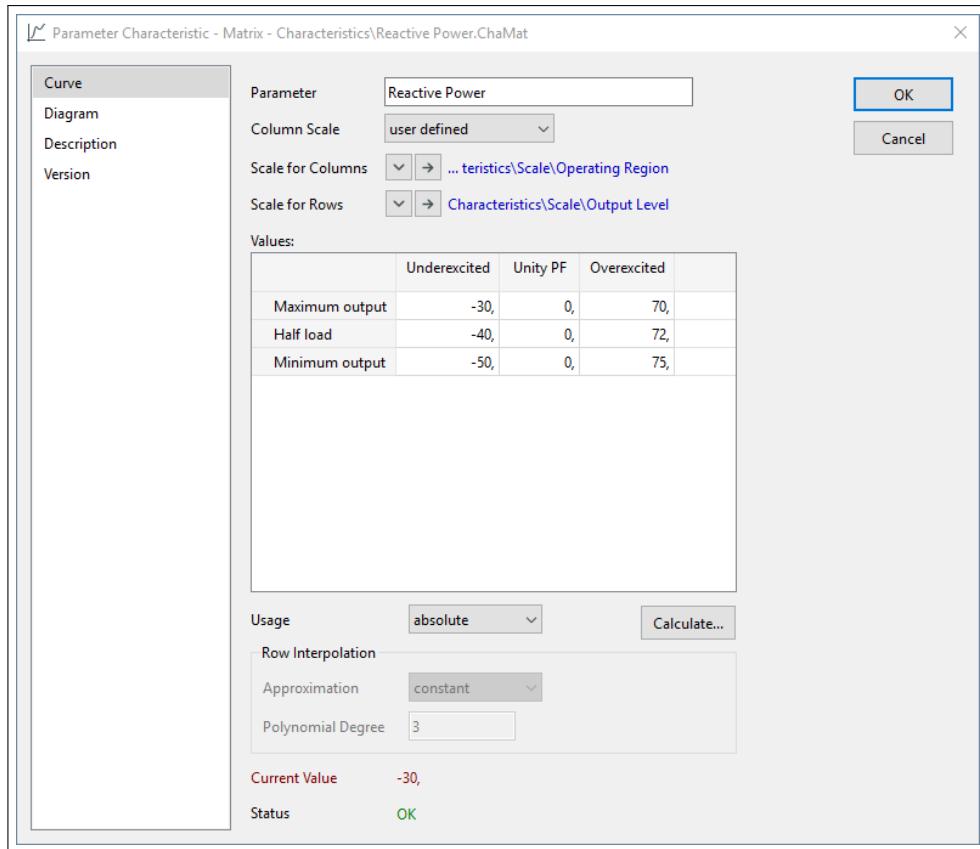


Figure 18.2.7: Reactive Power Matrix Characteristic

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 18.2.8).

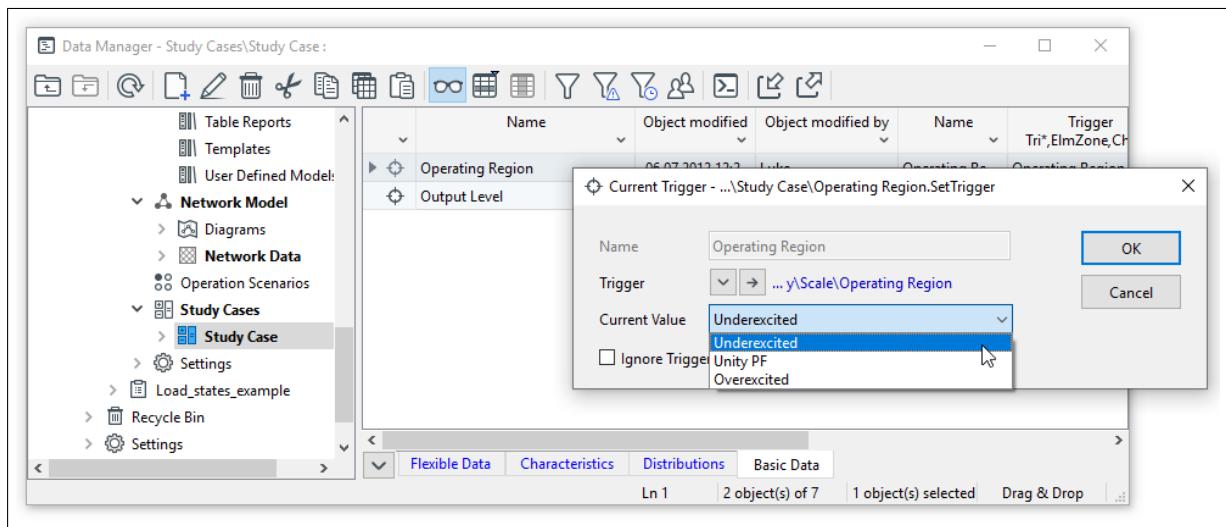


Figure 18.2.8: Setting of Discrete Triggers

## 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 18.2.1 and Vector Characteristics with Time Scales (*TriTime*) in Section 18.2.5 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:

#### 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.

#### 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.

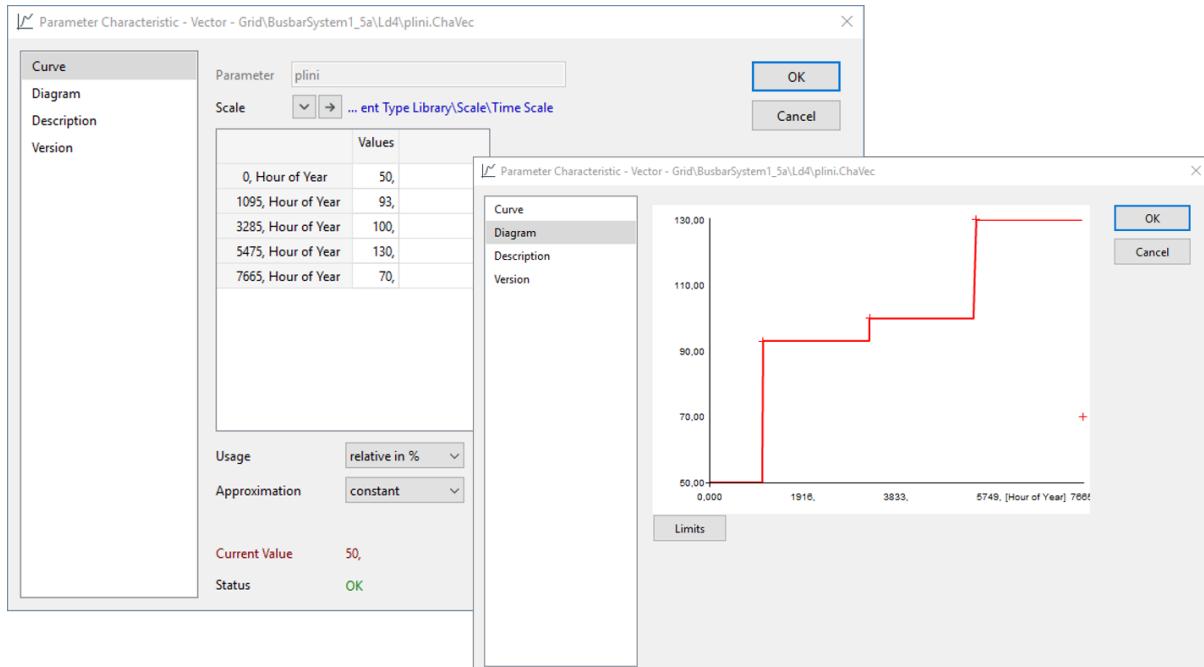


Figure 18.3.1: Load State Vector Characteristic

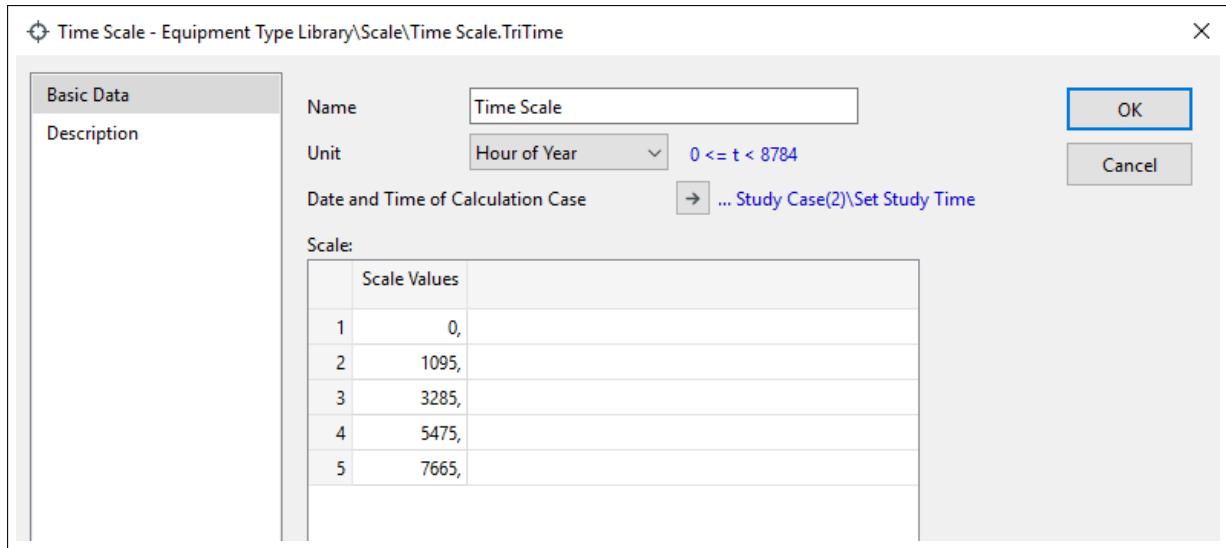


Figure 18.3.2: Time Scale for Load State Characteristic

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.

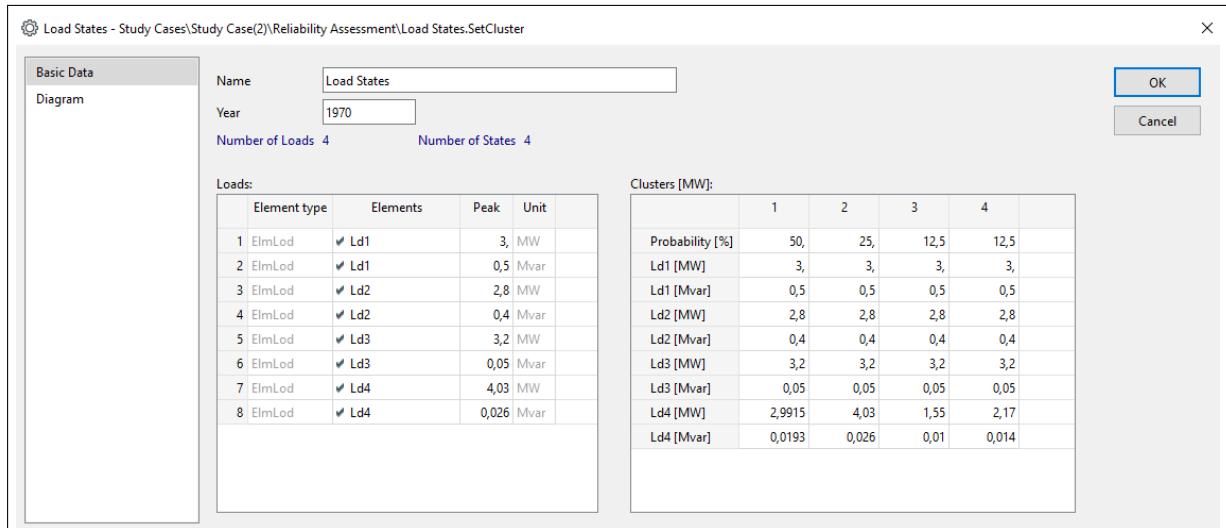


Figure 18.3.3: Load States (SetCluster) dialog box

## 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 (0.05pu by default).
6. Optional: Force Load State at  $S = 1.0 \text{ p.u.}$  so that a state is created at  $P = 1.0 \text{ pu}$ , 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 () (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:

#### 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.

### 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 pu 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 pu of substation load).

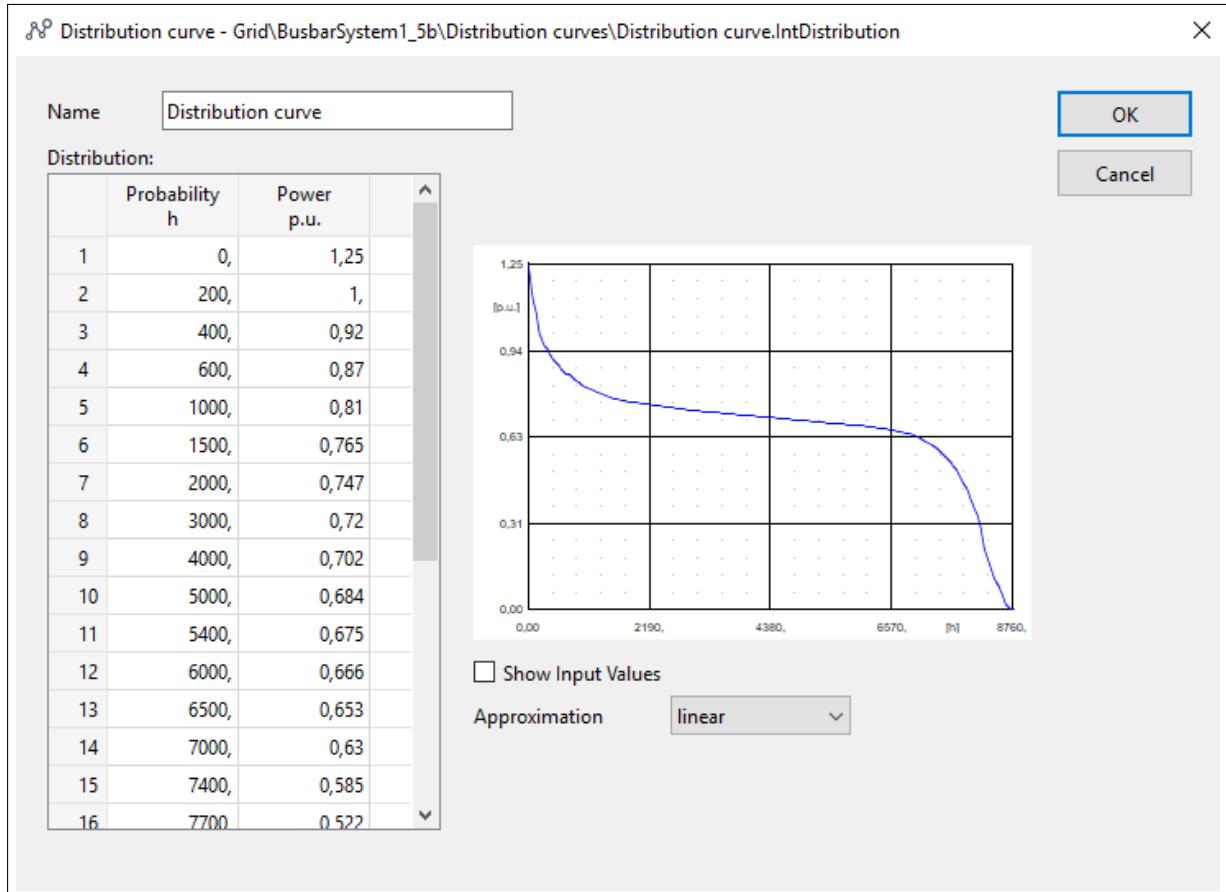
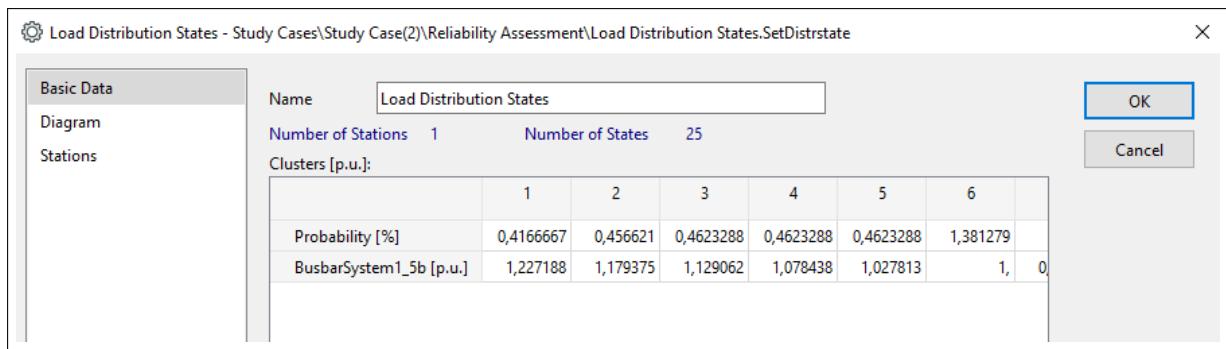


Figure 18.4.1: Substation Load Distribution Curve (*IntDistribution*)

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 'Equipment Type Library' 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 a time tariff object by pressing the *New Object* button from the data browser toolbar. A type creation dialog should appear.
4. Select *Time Tariff* and press **OK**. A 'Time Tariff' dialog box will appear.
5. 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.
6. Enter values for the Time Tariff (right click and 'Append rows' as required).
7. Press **OK** to return to the load element reliability page.
8. Optional: enter a scaling factor for the Tariff.

#### 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.

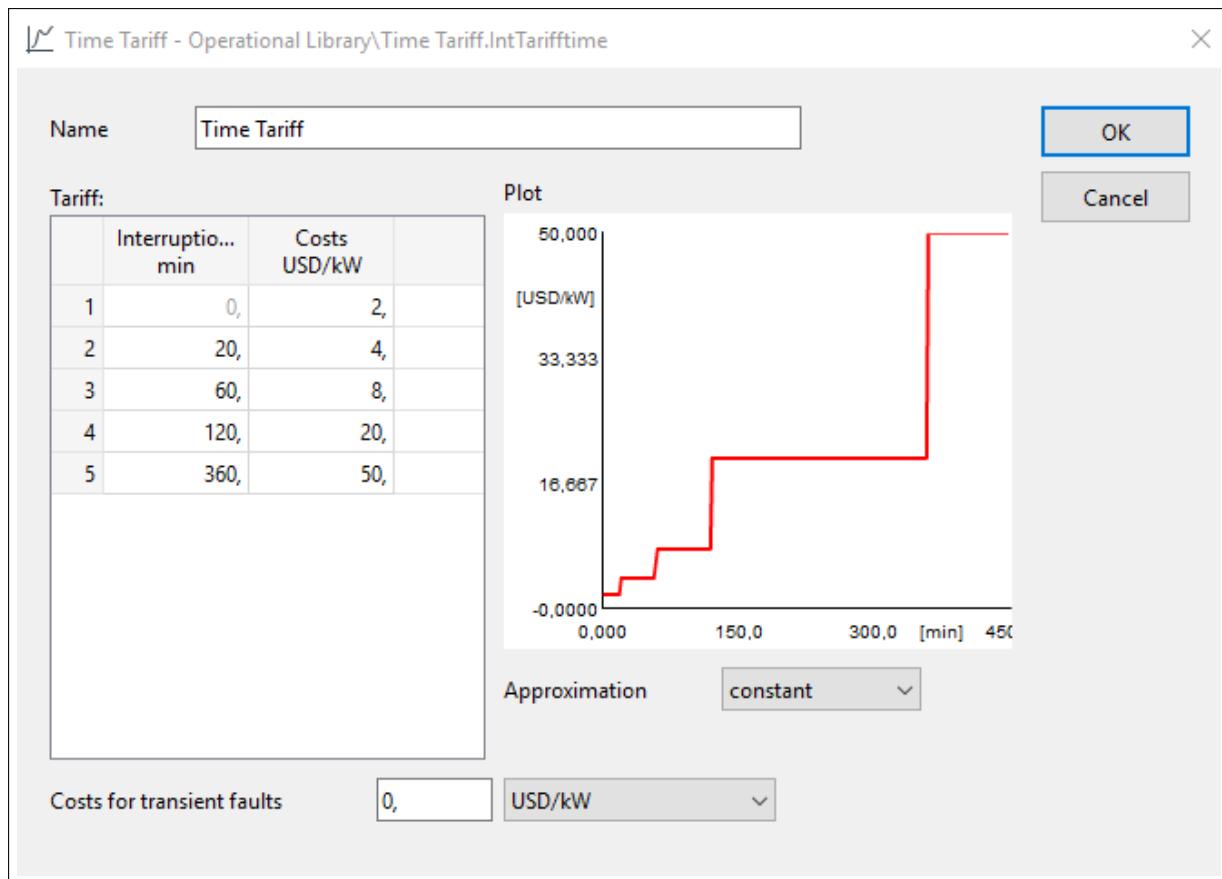


Figure 18.5.1: Example Time Tariff

## 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 'Equipment Type Library' 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. A type creation dialog should appear.
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.
7. Optional: enter a scaling factor for the Tariff.

### 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

$$\$9,20 \cdot 2,50 \cdot 1000 = \$23000 \quad (18.1)$$

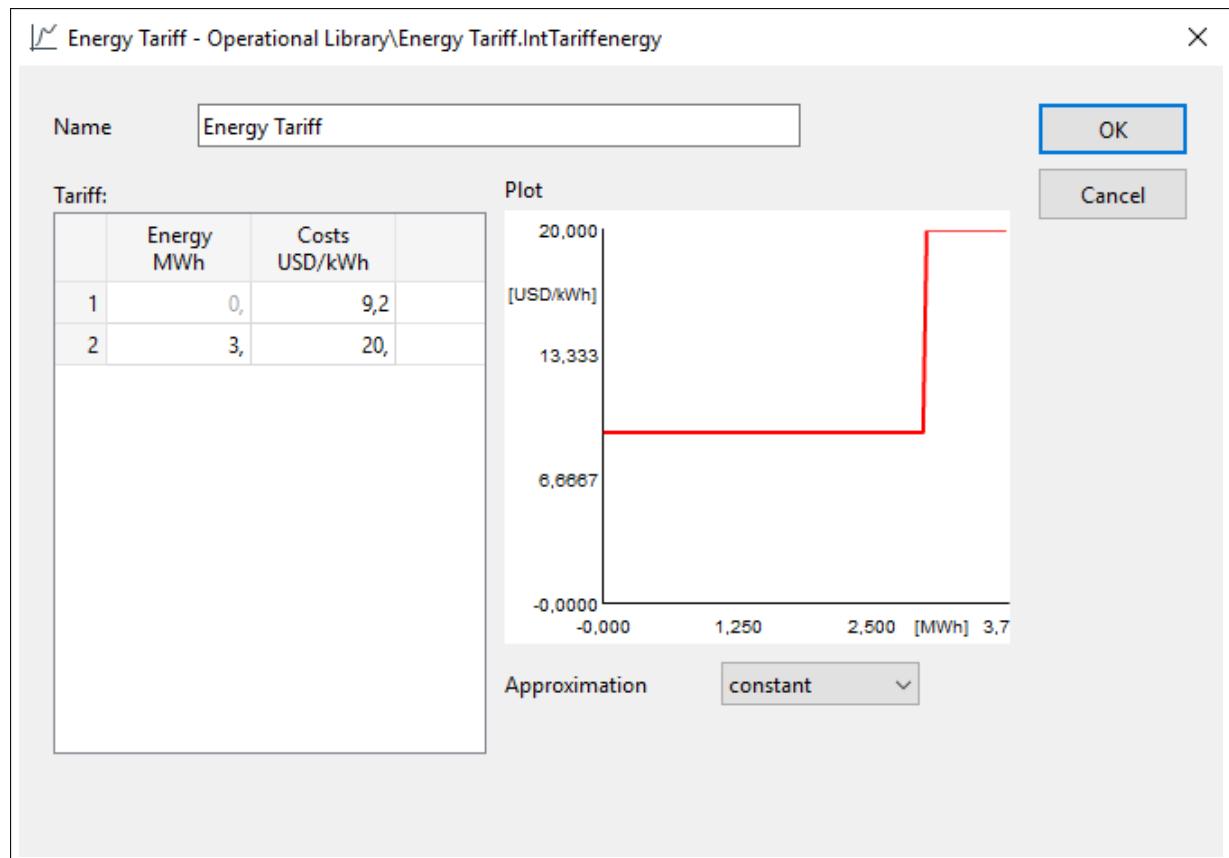


Figure 18.5.2: Example Energy Tariff

# 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 9.5.
- 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 19.4 describes the predefined reports available in *PowerFactory* to present data in the output window.
- Section 19.5 describes the option to compare steady state calculations results.
- Section 19.6 describes the *Results File* object to store results or selected variables.
- Section 19.7 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 9, Section 9.5 (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...*

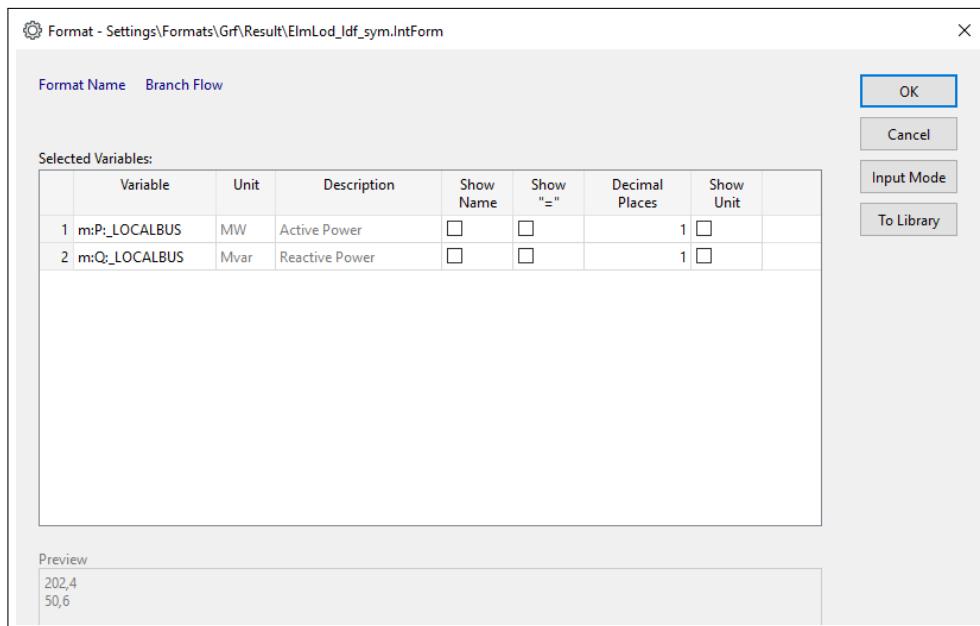


Figure 19.2.1: The Format dialog

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.

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 *DIGSILENT* 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 D (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* (see section 10.6).

The variable selection dialog is shown in Figure 19.3.1. The object for which the variables are defined is marked in red, the calculation to which these variables belong to in green, and the selected variables in blue.

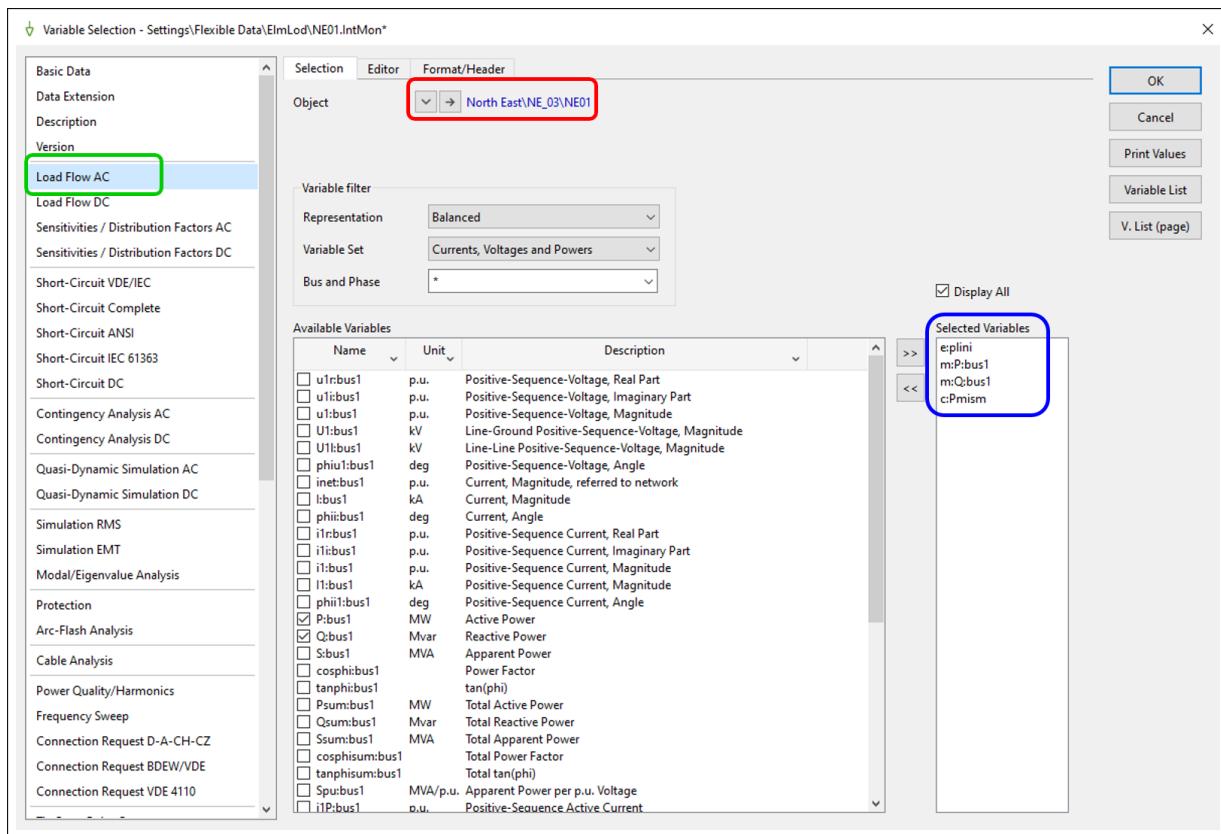


Figure 19.3.1: Example of a variable selection dialog

The variable selection object contains the following fields:

- **Object:** the object (normally a network component), whose variables are going to be monitored.
- **Class Name:** if no object has been selected the *Class Name* field becomes active. The use of the class name instead of the object for variable definition is only valid for some calculation types (e.g Quasi-dynamic simulation).
- **Display Values during simulation in output window:** this is only visible when the variable selection is being done for a time domain simulation (RMS/EMT). 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.
- **Page:** the first step is to select the page on the left-hand side of the dialog, because the variables are sorted by calculation function (e.g. Basic Data, Load Flow AC, Short-Circuit DC, etc.).
- **Variable Filter:** within this panel further filtering options are available:
  - **Representation:** depending on the type of calculation to be monitored (balanced or unbalanced), it is possible to toggle between balanced and unbalanced variable selections.
  - **Variable Set:** the variables are further sorted into variable sets, which are described below in the subsection 19.3.1.
  - **Bus and Phase:** this filter allows the selection of individual phase information where appropriate, or the selection of bus for multi-port elements.
- **Available Variables:** all the variables available for display, according to the above selections, are listed here. To help the user to find the required variables, a drop-down arrow on each column header allows the list to be filtered further.
- **Selected Variables:** the selected variables. Variables are placed here by double clicking on them on the *Available Variables* side, by selecting their checkbox, or by selecting them and then pressing

the (») button. Variables can be removed from the *Selected Variables* area by double-clicking on them or by selecting them and then pressing the («) button.

- **Display All:** if this box is checked then all of the selected variables are shown in the *Selected Variables* area. If not checked, the filter selected in the *Filter for* field will also apply to the *Selected Variables* area and only those selected variables in the filtered set will be shown.

The following buttons are available on the right side of the dialog:

- **Print Values:** the current values of all the selected variables are displayed in the output window.
- **Variable List:** a list of all available variables is printed in the output window.
- **Variable List (Page):** a list of all the available variables for the current page (e.g. *Basic Data*) is displayed in the output window.

The second tab of the Variable Selection dialog goes to the *Editor*, where variables can be manually input. If the variable selection dialog is used to define the *Flexible Data* page, an additional tab called *Format/Header* is visible; more information about this tab is available in section 10.6.1: Customising the Flexible Data Page.

### 19.3.1 Variable Selection Filter

The first sorting of the variables is by calculation function. Within these sets variables are sorted into sub-sets. The desired subset can be selected using the drop down menu on the *Variable Set* field. These are the available subsets:

- **Currents, Voltages and Powers:** almost self explanatory - these are the outputs as calculated by a calculation function. The variable is preceded by “m:” (representing ‘monitored’ or ‘measured’) as in `m:P:bus1` for the active power drawn by the load.
- **Bus Results:** variables for the bus/es where the element is connected (usually preceded by “n:” for ‘node’). An element having only one connection to a bus, will obviously only have results for “Bus1.” An element having two connections will have “Bus1” and “Bus2”. This means that the results of objects connected to the object whose variable list is compiled can be accessed.
- **Signals:** variables that can be used as interface between user defined and/or *PowerFactory* models (inputs and outputs). They are preceded by “s:”. These should be used when creating a controller or in a DPL script. These variables are accessible whilst an iteration is being calculated, whereas the other variables sets are calculated following an iteration.
- **Calculation Parameter:** variables that are 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. nominal power). The parameters that actually are available depend on the object type. Calculation parameters are preceded by a “c:”.
- **Element Parameter:** input parameters that belong directly to the object selected (preceded by “e:”).
- **Type Parameter:** input parameters from the corresponding type object that are linked to the element object under consideration; for example, the current rating of a line type that a line element is using.
- **Reference Parameter:** these are variables from objects that are linked or connected to the object under consideration (preceded by “r:”). For example, a line element may be part of a line coupling and the reference parameter will allow us to display the name of the coupling element.

For general use it is sufficient to simply select the variables required and transfer them to the selected variables column. To find a particular variable requires some knowledge of where the variables are stored in the object under consideration.

Additional information about how the result variables are calculated is available in section Technical References of Result Variables of the [Technical References Document](#).

User defined variables are shown when the page *Data Extensions* is selected. Additional information about Data Extensions is available in Chapter [20](#).

## 19.4 Output Reports

*PowerFactory* offers two types of reports which are printed in the output window. The *Documentation of Device Data* prints either all or a part of the data entered in *PowerFactory* the *Output of Calculation Analysis* prints the results of a previously executed calculation in the output window.

### 19.4.1 Documentation of Device Data

The *Output of Device Data* command (*ComDocu*) can be accessed by clicking on the icon  on the main tool menu.

#### 19.4.1.1 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. Another way to select object for the report is to right-click on the objects from the Data Manager or the single line graphics and select *Output Data → Documentation*, this will open the *Documentation of Device Data* command.
- **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.

#### 19.4.1.2 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.

#### 19.4.2 Output of Results

The command *Output of Results* (*ComSh*) is used to produce an output of calculation results. The output can be used in reports or may help in interpreting the results and is accessed by clicking on the icon  from the main tool menu.

Several different reports, depending on the actual calculation, can be created. The radio button on the upper left displays the different reports possible for the active calculation. Some reports may be inactive, depending on the object(s) chosen for output. On the Advanced tab, the *Used Format* gives access to the format(s) used for the report. Some reports are a set of different outputs. For these reports more than one form is shown. If the form is modified it will be stored automatically in the *Settings* folder of the active project. The changed form does not influence the reports of other projects. If *Use Selection* is active, a set of objects (selection) or a single object must be chosen. The report is generated only for these elements. All relevant objects are used if *Use Selection* is not selected. The relevant objects depend on the chosen report. Most reports display a title on top of each page. The reference *Title* defines the contents of the header.

For some reports additional settings are required. These settings depend on the chosen report, the selected objects for output and the calculation processed before. The calculation (left top) and the used format(s) (right top) are always shown.

One option for Load Flow calculations is the *Power Interchange* report. If this option is selected, a tabular report is generated, showing the power interchange between various parts of the network, i.e. Grids, Areas or Zones. When reporting Power Interchange for Grids, a further feature is available: If the several connected networks are modelled, and the connections are represented by boundary nodes held in a dedicated boundary grid, the user may not be interested in power interchanges with the boundary grid itself but rather in the power interchanges between the grids either side of it. Therefore there is now an option provided for the *ElmNet* object, to mark it as a “Fictitious border grid”. If this is done, then it becomes transparent from the point of view of this Power interchange report, which then reports the interchange between the grids of interest.

### 19.5 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.

## 19.6 Results Objects

The results object (*ElmRes*, ) is used by *PowerFactory* to store tables of results. 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 inside the database
- **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 database 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 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](#).

## 19.6.1 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 *ASCII Result Export* command, which enables the definition of the format and the file type used to export the results.

### 19.6.1.1 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
- Textfile
- 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.

### 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*.

#### 19.6.1.2 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 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, etc. These could be in the form of a bar graph, a plotted curve, single displayed variables, tables of values, etc.

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 19.7.1.

Once a plot has been created, it is held in the Graphics Board 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*).

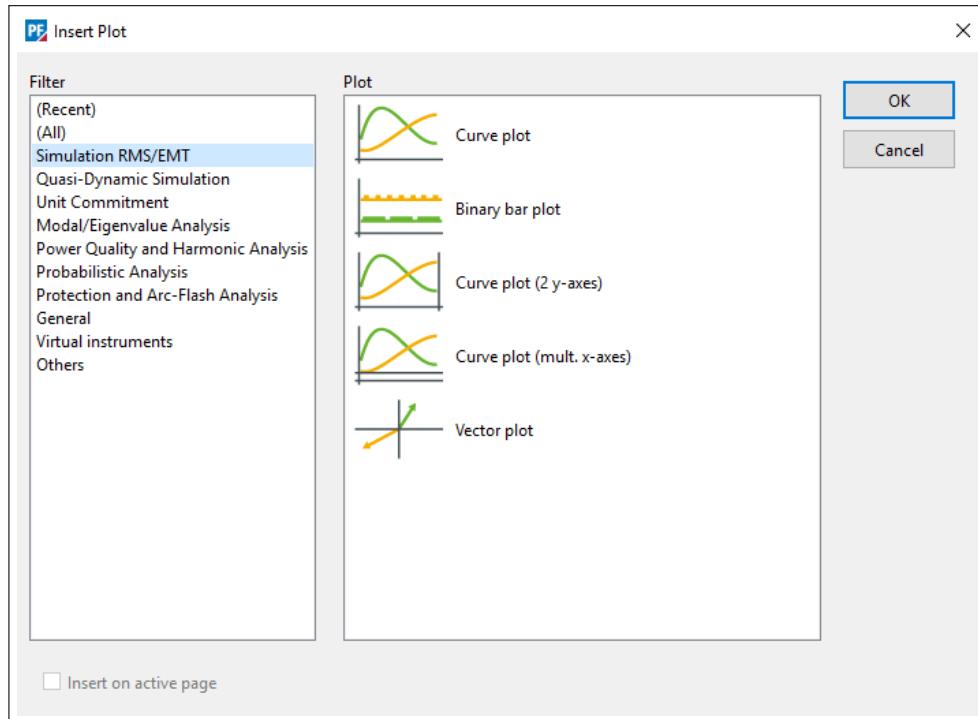


Figure 19.7.1: Insert Plot dialog

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

- Binary bar plot (section 19.7.4)
- Curve plot (section 19.7.1)
- Curve plot (2 y-axis) (section 19.7.2)
- Curve plot (mult. X-axes) (section 19.7.3)
- Vector plot (section 19.7.7)

### 2. Quasi-Dynamic Simulation

- Duration curve (section 19.7.11)
- Complete Generation (section 19.7.13)
- Curve plot (section 19.7.1)
- Curve plot (2 y-axis) (section 19.7.2)
- Curve plot (mult. X-axes) (section 19.7.3)
- Plant Categories (section 19.7.12)
- Renewable and Fossil (section 19.7.14)

### 3. Modal/Eigenvalue Analysis Plots

- Eigenvalue Plot (section 32.3.2.1)

- Mode bar plot (section [32.3.2.2](#))
- Mode phasor plot (section [32.3.2.3](#))

#### 4. Power Quality and Harmonic Analysis Plots

- Curve plot (section [19.7.1](#))
- Harmonic distortion
- Waveform Plot (section [36.5.4](#))

#### 5. Probabilistic Analysis Plots

- Convergence of statistics (section [43.3.8.3](#))
- Correlation plot (section [43.3.8.4](#))
- Distribution estimation (section [43.3.8.5](#))
- Distribution fitting (section [43.3.8.6](#))

#### 6. Protection and Arc-Flash Analysis Plots

- Current comparison differential plot (section [33.10.1](#))
- Curve-input (section [19.7.8](#))
- Phase comparison differential plot (section [33.10.2](#))
- R-X plot (section [33.6](#))
- The 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](#))

#### 7. General Plots

- Bar plot (section [19.7.6](#))
- Binary bar plot (section [19.7.4](#))
- Curve plot (section [19.7.1](#))
- Curve plot (2 y-axis) (section [19.7.2](#))
- Curve plot (mult. X-axes) (section [19.7.3](#))
- Vector plot (section [19.7.7](#))
- Curve-input (section [19.7.8](#))

#### 8. Virtual Instruments (section [19.7.9](#))

- Digital display
- Horizontal scale
- Vertical scale
- Measurement instrument

#### 9. Others

- Button
- Command button
- Network graphic
- Schematic path
- Text
- Voltage profile (along feeder) (section [19.7.10](#))
- Voltage sag plot

All the plot types can be edited either by right-clicking on the plot and selecting *Edit*; 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 19.7.15.

The plots can be exported by selecting the option *File → Export Graphic... →* from the main menu or using the *Diagram Export* icon  on the graphic 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 (\*.tiff)
- File Interchange Format (\*.jpg; \*.jpeg; \*.jpe; \*.jfif)
- Windows Bitmap (\*.bmp)
- Windows Metafile (\*.wmf)

### 19.7.1 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 following figure shows an example of a curve plot.

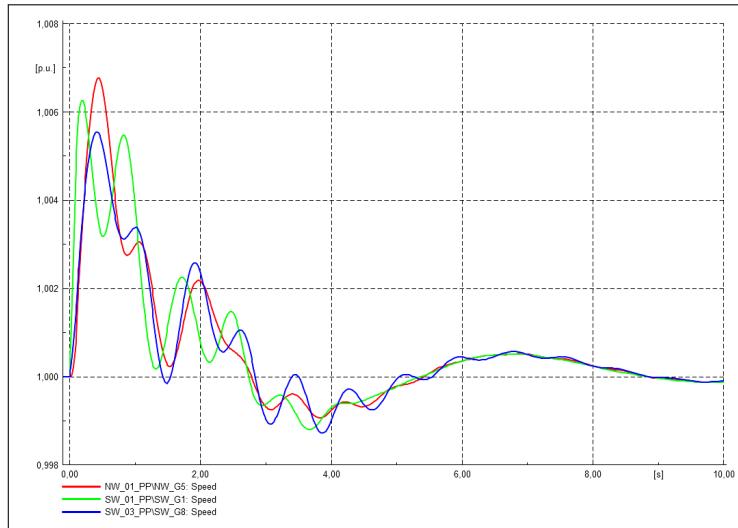


Figure 19.7.2: Curve Plot

#### Y-Axis Page

In the y-axis page of the curve plot dialog, the parameters are defined in two tabs: *Variables* and *Scale*. The *Variables* tab includes the following fields:

- **Automatic:** the colour, line style, and line width of the new curves in the plot will be set automatically when the corresponding option is enabled. The **Apply** button will apply automatic line formats to all existing curves again.

- **Results:** this is a reference to the currently active results file (*ElmRes*). With the selection *Search automatically* the currently used result file of the last calculation is chosen. The option *User defined* gives the user the opportunity to choose another result file. The selected object will be used, if no results file is 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. The curve definition is the same for all the plot types and therefore more information on defining curves is provided in the separate section [19.7.5](#).
- **User Defined Signals:** allows arithmetic calculation of additional results based on *PowerFactory* calculated results. The method to create a calculated result is explained later on in this section on page [279](#).

The *Scale* tab includes the following fields:

- **Axis:** the y-axis options can be defined locally using the default *Local* option; alternatively a plot *Type* can be used and assigned to all the plots. In this way several plots can be easily compared, without the risk of misinterpreting a difference in curve amplitude. If the *Type* option is selected in the *Axis* field, the plot type can be edited by clicking on the → button.
- **Minimum and Maximum:** the y-axis limits can be set manually, or can be auto scaled using the  button. The scale button sets the limits automatically from the curve shape.
- **Scaling:** the y-axis scale can be set to linear, logarithmic or dB.
- **Auto Scale:** the following auto scale options are available:
  - Off: turns any auto scaling function off and displays the results in the range between the given limits.
  - On: will automatically scale the plot at the end of a simulation.
  - Online: will automatically scale the plot during the simulation.
- **Adapt Scale:** settings to adapt the scale to a setpoint. The tick marks can be forced by setting the *Offset* value. If the *Show Deviations from Offset* option is selected, a second axis with a zero baseline at the offset value, will be drawn on the right side of the plot.

### X-Axis Page

In the x-axis page of the curve plot dialog, the parameters are defined in two tabs: *Variables* and *Scale*. The *Variables* tab has the following fields:

- **Name:** the name of the plot.
- **x Variable:** there are numerous options to choose from for this field, as shown in Figure [19.7.3](#). The *Default* value depends on the type of simulation and the results object created during the previous simulation. For time-domain simulations different representations of the time scale are available.

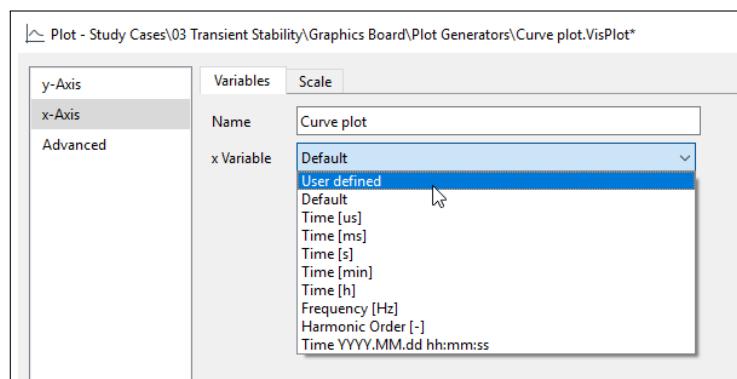


Figure 19.7.3: The variable list available for the x-Axis

If the option *User defined* is selected, any variable can be selected from a results object. In this way an x-y plot can be created. Whilst the curve plot can be used to create x-y plots, there is also a specific plot type to create an x-y plot: the *Curve plot (multiple x-axes)*, which is described in section [19.7.3](#).

The *Scale* tab has the following fields:

- **Axis:** x-axes often needs to be synchronised for all plots or for all plots on one Plot Page, for instance to show the same time-scale in all plots. This is why the default option of this field is *Graphics Board*, which sets the x-axis of all the plots in the project. The other options are:
  - Local: the x-axis scale is only valid for the local plot.
  - Page: all the plots on the plot page will use the same x-axis, also described in section [19.7.15.10](#).
- The *Graphics board* and *Page* scale options can be accessed by clicking on the → button by the *Used Axis* field.
- **Chart:** if ticked, a range and a start value can be set. This will set the x-axis to the specified range. During the simulation, only an x-range, set in the options, is shown and will “wander” along with the calculation time.
- **Minimum and Maximum:** the x-axis limits can be set manually, or can be auto scaled using the  button. The scale button sets the limits automatically from the curve shape.
- **Scaling:** the x-axis scale can be set to linear or logarithmic.
- **Auto Scale:** the following auto scale options are available:
  - Off: turns any auto scaling function off and displays the results in the range between the given limits.
  - On: will automatically scale the plot at the end of a simulation.
  - Online: will automatically scale the plot during the simulation.
- **Adapt Scale:** settings to adapt the scale to a setpoint. The tick marks can be forced by setting the *Trigger* value.
- **Grid Lines:** settings customise the number of grid lines on the plot.
  - Automatic: will draw the grid lines according to the default setting in *PowerFactory*.
  - User defined number of grid lines: will draw the grid lines according to the number specified by the user.

## Advanced Page

On the *Advanced* page of the curve plot, the following additional display options can be selected:

- **Type:** the layout of the plot can be modified by clicking on the → button. The options include adding additional lines to the axis (using the *Help* checkbox) and defining the location of the axis ticks (none, inside, outside or both)
- **Display options:** three check boxes are provided to define the visibility of the axis and the plot name on the plot.
- **Legend:** the position of the legend (none, bottom or right) and the description size is defined.
- **Frame:** this defines the frame of the plot (off, simple, 3D or 3D with user defined label)
- **Presentation:** the way the results are displayed on the plot can be set to the default option *curves* or to *bars*. If the option *curves*, the simulation steps can be marked on the curve by selecting the option *draw steps*.

## Export Button

When clicking on the **Export...** button on the right of the plot, the *ASCII Result Export* command, described in section [19.6.1](#), 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.

---

**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.

---

### Define Results Button

Using this button, additional calculated results can be added to the curve. Once the button is clicked a dialog opens and a new calculated result can be added by clicking on the **New**  icon. The calculated results are also known as *User Defined Variables* and are described below.

### User Defined Signals

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 **New** button on the edit dialog of the curve plot or by using the **Define Results** button on the right of the curve plots dialog. An example of the calculated result dialog is shown in Figure 19.7.4.

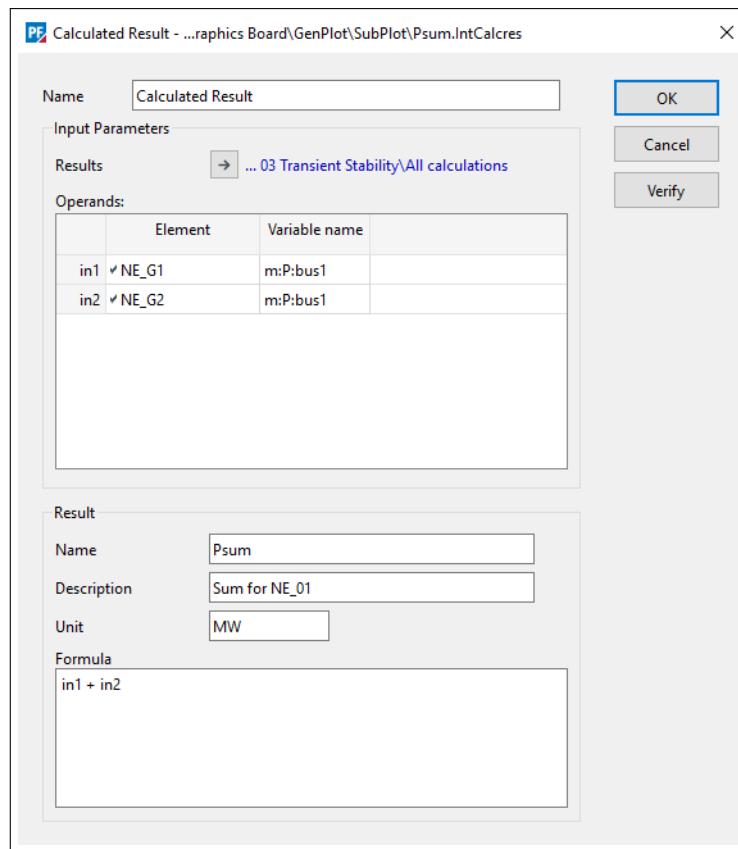


Figure 19.7.4: The calculated results object

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](#).

## 19.7.2 Curve Plot (2 y-axes)

A curve plot with two y-axes is typically used for displaying together signals which have very different scales. The pages on the edit dialog of the plot are the same of the curve plot described in section [19.7.1](#), the main difference is the additional page for the *Y2-Axis*.

The following figure shows an example of a curve plot with 2 y-axes.

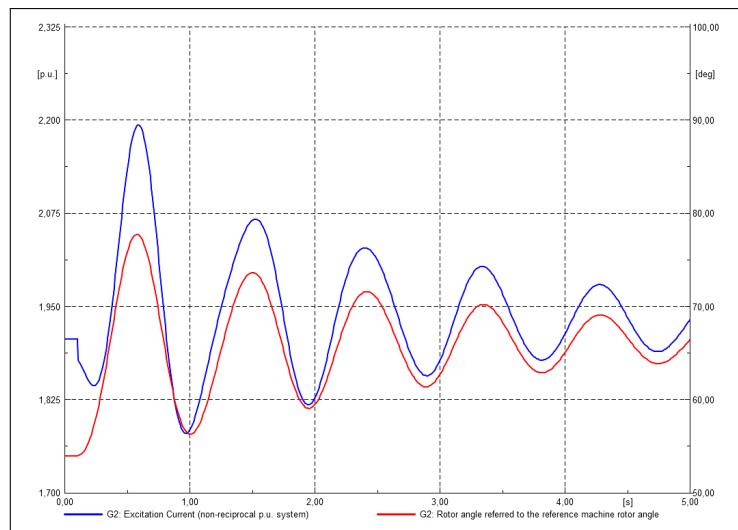


Figure 19.7.5: Curve Plot (2 y-axes)

The following options available for the curve plots are *not* available for 2 y-axes plots:

- dB scale for the y axis
- Option to present the curves as bars
- Option to hide the axes

### 19.7.3 Curve Plot (multiple x-axes)

This plot, also known as XY 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 one element.

The following figure shows an example of a xy curve plot.

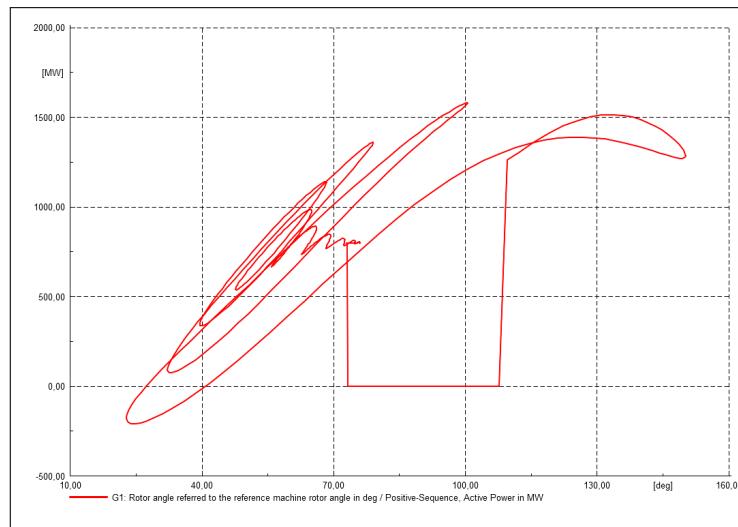


Figure 19.7.6: Curve Plot (multiple x-axes)

## Variables Page

On this page, the variables for the x- and y-axis are specified. Both variables have to be stored in one results file of a simulation. The *Variables* page has the following fields:

- **Automatic:** the colour, line style, and line width of the new curves in the plot will be set automatically when the corresponding option is enabled. The **Set now** button will apply automatic line formats to all existing curves again.
- **Results:** this is a reference to the currently active results file (*ElmRes*). This object will be used if no results file is specified in the *Variables* table.
- **Variables:** 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. To select variables of two different elements, the option *Show x-Element in Table* has to be activated. The curve definition is the same for all the plot types and therefore more information on defining curves is provided in the separate section [19.7.5](#).
- **Show direction arrows for curves:** when ticked, additional arrows indicating the “growing direction” are drawn on the curves.

### Scales Page

On this page, the scales of the two axes can be set locally or alternatively global definitions can be used depending on the *Use local scales* check box. The *Scale* has the following fields:

- **Scaling:** the axis scaled can be set to linear or logarithmic.
- **Adapt Scale:** settings to adapt the scale to a setpoint. The tick marks can be forced by setting the *Offset* value; the offset for each axis is defined on the *scales* table.
- **Auto Scale:** the following auto scale options are available:
  - Off: turns any auto scaling function off and displays the results in the range between the given limits.
  - On: will automatically scale the plot at the end of a simulation.
  - Online: will automatically scale the plot during the simulation.
- **Scales table:** the minimum, maximum and offset value for each axis (used if the auto scale is on) are defined in this table.

### Time Range Page

The time range can be set to the whole simulation time or alternatively select a specified range to show the results pertaining to a specific time range only.

### Advanced Page

On the *Advanced* page of the curve plot, the following additional display options can be selected:

- **Type:** the layout of the plot can be modified by clicking on the → button. The options include adding additional lines to the axis (using the *Help* checkbox) and defining the location of the axis ticks (none, inside, outside or both)
- **Legend:** the visibility and the description size (long or short description) of the legend is defined in this field.
- **Frame:** this defines the frame of the plot (off, simple, 3D or 3D with user defined label)

The plots buttons **Export...**, **Filter...** and **Define Results...** are the same for all the curve plots and are described on page [278](#).

### 19.7.4 Binary Bar Plot

The Binary Bar Plot can be used to illustrate digital signals. If a digital signal is true (i.e. the absolute value of a signal is greater than 0.5), the plot displays it in form of a bar. In contrast to this, false values are represented as lines. An example is shown at the bottom of figure 19.7.7

The pages on the edit dialog of the binary plot are the same as the curve plot described in section 19.7.1. The only difference is that the *Presentation* field is not available in the *Advanced* page since it is by default set to *Bars*.

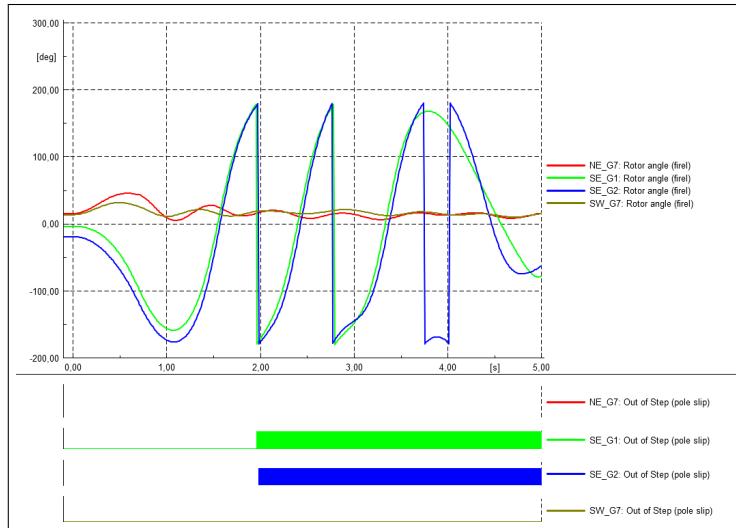


Figure 19.7.7: Binary Bar Plot

### 19.7.5 Specifying Curves for Plots

For the plots mentioned so far, the values of the curves are read from a results object (*ElmRes*) or a calculated results object (*IntCalcres*). Results objects were described in section 19.6 and user defined variables or calculated results on page 279.

Each line in the *Curves* table is used to define a variable to plot and the visual representation of the curve.

- The first column states the results object from which the data to plot the curve will be read. If it is empty, the standard results file will be used, as defined in the reference to *Shown Results* in the same dialog.
- The second column states the power system element, which is selected from the available elements in the results object.
- The third column states the actual variable for the curve, selected from the variables in the results object, belonging to the selected element.
- The next columns specify the style of the individual curve.
- With the last two columns the values of the variable can be normalised (Norm.) to a nominal value (Nom. Value).

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* should 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.

**Note:** Different results files can be used in the same plot. This is useful for comparing curves of different simulations.

### 19.7.6 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 *Show → Bar Plot → “variable”*. An example of the bar plot is shown in the following figure.

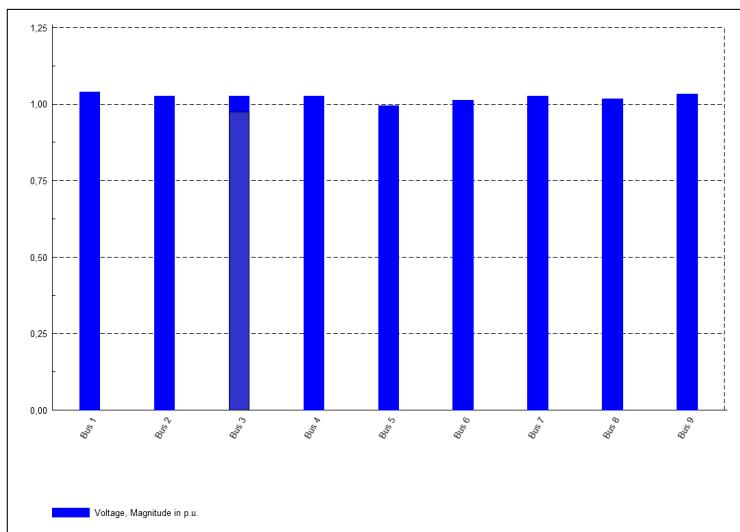


Figure 19.7.8: Bar Plot

#### Y-Axis Page

On the y-axis page of the bar plot dialog, the parameters are defined in two tabs: *Variables* and *Scale*. The *Variables* tab has the following fields:

- **Automatic:** the colour and brush style of the new variables will be set automatically when the corresponding option is enabled. The **Apply** button will automatically apply the selected format to the existing variables.
- **Bars table:** used to specify the variables for all the selected elements as well as their representation. If the plot is inserted by right clicking on the element, the variables are automatically defined on this table. Otherwise it is necessary to define first the element on the x-axis and then the variables can be selected by double clicking on the *variable* field. Once a variable is defined, additional variables can be added using the **Append variables** button.

The *Scale* tab has the following fields:

- **Axis:** the y-axis options can be defined locally using the default *Local* option; alternatively a plot *Type* can be used and assigned to all the plots. The plot type can be edited by clicking on the → button.
- **Minimum and Maximum:** the y-axis limits can be set manually, or can be auto scaled using the button.
- **Scaling:** the y-axis scale can be set to linear, logarithmic or dB.

- **Auto Scale:** the following auto scale options are available:
  - Off: turns any auto scaling function off and displays the results in the range between the given limits.
  - On: will automatically scale the plot at the end of a calculation.
  - Online: will automatically scale the plot during the calculation.
- **Adapt Scale:** settings to adapt the scale to a setpoint. The tick marks can be forced by setting the *Offset* value. If the *Show Deviations from Offset* option is selected, a second axis with a zero baseline at the offset value, will be drawn on the right side of the plot.

### X-Axis Page

The x-axis of the bar plot 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 on the *Net Elements* table. Otherwise the element can be selected by double clicking on the corresponding field.

### Advanced Page

On the *Advanced* page of the curve plot, the following additional display options can be selected:

- **Type:** the layout of the plot can be modified by clicking on the → button. The options include adding additional lines to the axis (using the *Help* checkbox) and defining the location of the axis ticks (none, inside, outside or both)
- **Display options:** three check boxes are provided to define the visibility of the axis and the plot name on the plot.
- **Legend:** the position of the legend (none, bottom or right) and the description size is defined.
- **Frame:** this defines the frame of the plot (off, simple, 3D or 3D with user defined label)

### 19.7.7 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

Figure 19.7.9 shows an example of a vector plot.

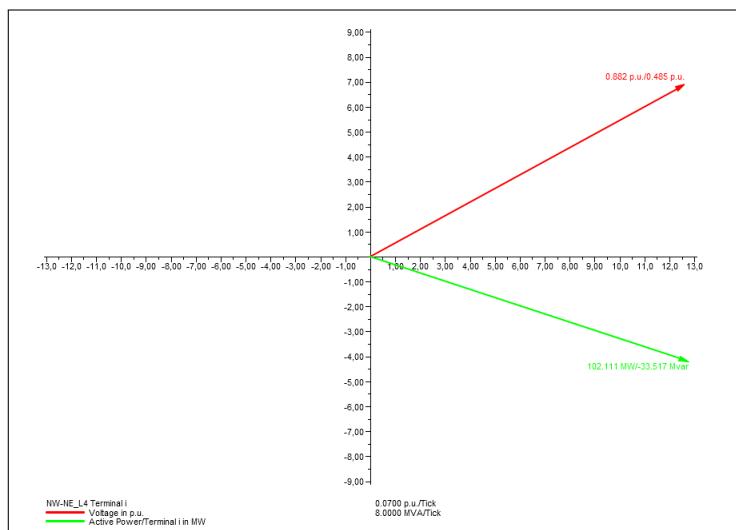


Figure 19.7.9: Vector Plot

---

**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 *Show → Vector Plot→ “variable”*

### Variables Page

- **Variables table:** 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.
- **Automatic:** the colour of the new curves in the plot will be set automatically when the option is enabled. The Set now button will assign the colour to all the existing curves again.
- **Show all phases in plot:** if checked, all the phases of the selected variable will be shown (for unbalanced calculations)
- **Scales table:** in most plots, the x and y scales are given by the minimum and maximum value of each scale. A vector plot can't be defined using minimum and maximum for each scale because the x- and the y-ratio must be equal. The ratio for each unit is therefore set as the parameter units per axis tick. In addition the location of the origin can be defined. If all shown variables have the same unit, the axis are labelled with values and unit. If there is more than one unit, the labels show ticks. A legend showing the ratio of the units is added in the bottom-right corner of the plot.
- **Min. values:** the minimum values for the x and y axes can be defined. This will modify the centre on the plot.
- **Auto Scale:** if turned on, the scales are adapted whenever a new calculation is ready; otherwise the defined scale limits will be used.

### Advanced Page

- **Label of Vectors:** the label of the vector can be displayed in the different coordinate representations. The different coordinate systems allow display with either the real and imaginary values or the magnitude and phase angles of the vectors.
- **Frame:** this defines the frame of the plot (off, simple, 3D or 3D with user defined label)
- **Representation of Coordinates:** the coordinates can be displayed as polar or cartesian representation. For this to be visible at least the main grid should be visible, the visibility of the grid is defined on the context sensitive menu of the plot (i.e. *Right click on plot → Grid*).

Apart from all the options already described, which are also accessible via the context sensitive menu of the plot, the following additional options are available for the vector plot (only available by right click):

- **Edit Element:** opens the edit dialog of the element whose variables are displayed in the plot.
- **Jump to Element:** shows a list of all connected elements from which one can be selected. Here the side of a branch element is automatically checked. The *Jump To* option is not available if there is more than one element shown in the same plot or if there are no calculation results available.
- **Set Origin:** moves the origin to the right-clicked position.
- **Centre Origin:** sets the origin to the centre of the plot.

### 19.7.8 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 19.7.10. This plot then allows plot points to be defined by successive mouse clicks.

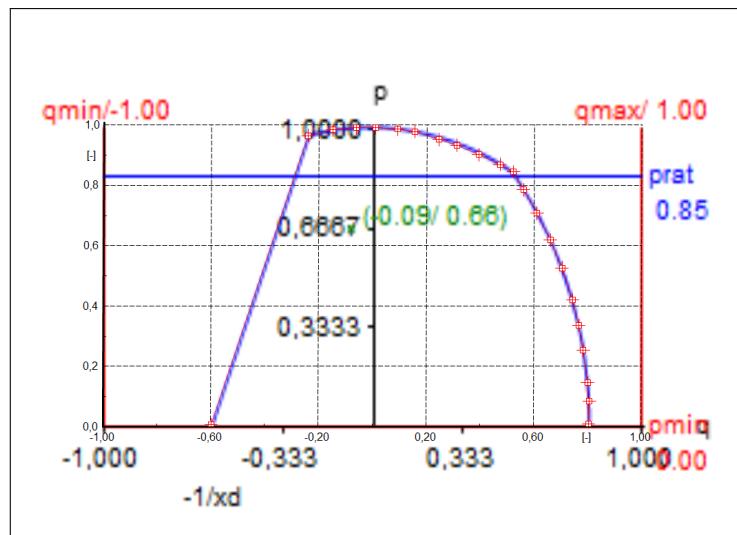


Figure 19.7.10: Curve-Input Plot

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* icon (). From the elements list, the *Matrix (IntMat)* should be selected as shown in figure 19.7.11. The matrix should have at least two columns and one point (inside the curve) has to be manually defined.

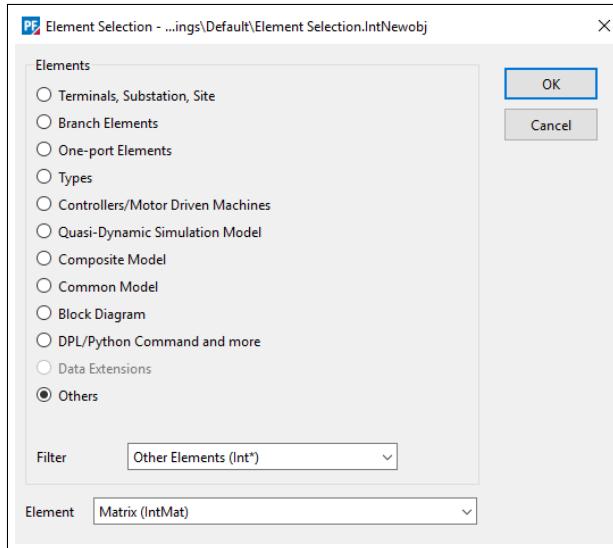


Figure 19.7.11: Defining new matrix

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 sensitive 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

### 19.7.9 Virtual Instruments

The virtual instruments are basically measurement instruments that can be inserted into the plot 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 all the available 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.

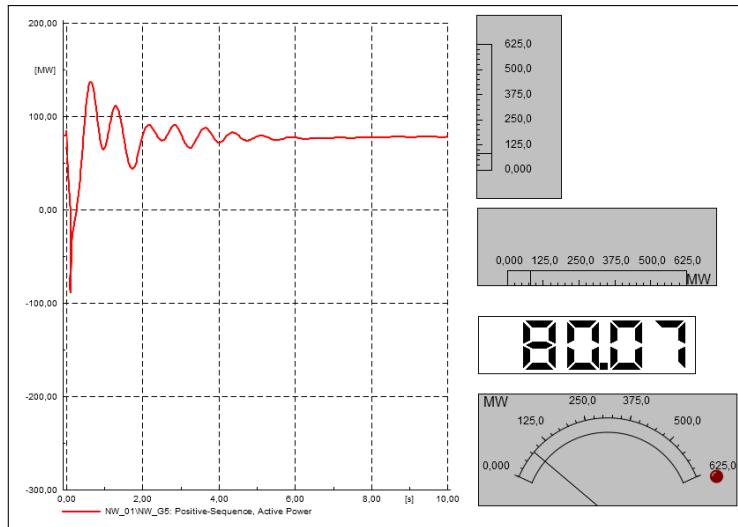


Figure 19.7.12: Measurement Instruments

### 19.7.10 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 sensitive menu of the feeder element, selecting *Show → Voltage Profile*.

An example of a voltage profile plot is shown here:

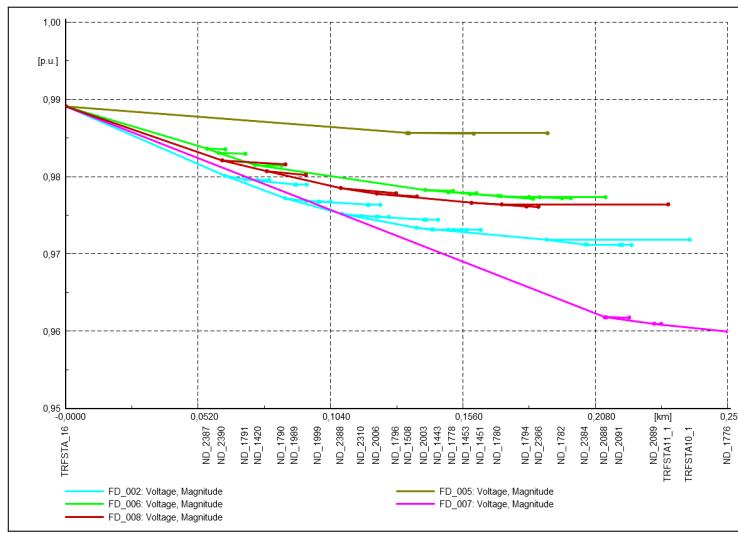


Figure 19.7.13: Voltage profile along feeders

### 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 sensitive 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.
- **Edit and Browse Data:** shows the selected element in the Data Manager.
- **Mark in Graphic:** marks the selected element in the single line graphic(s).

### 19.7.11 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.

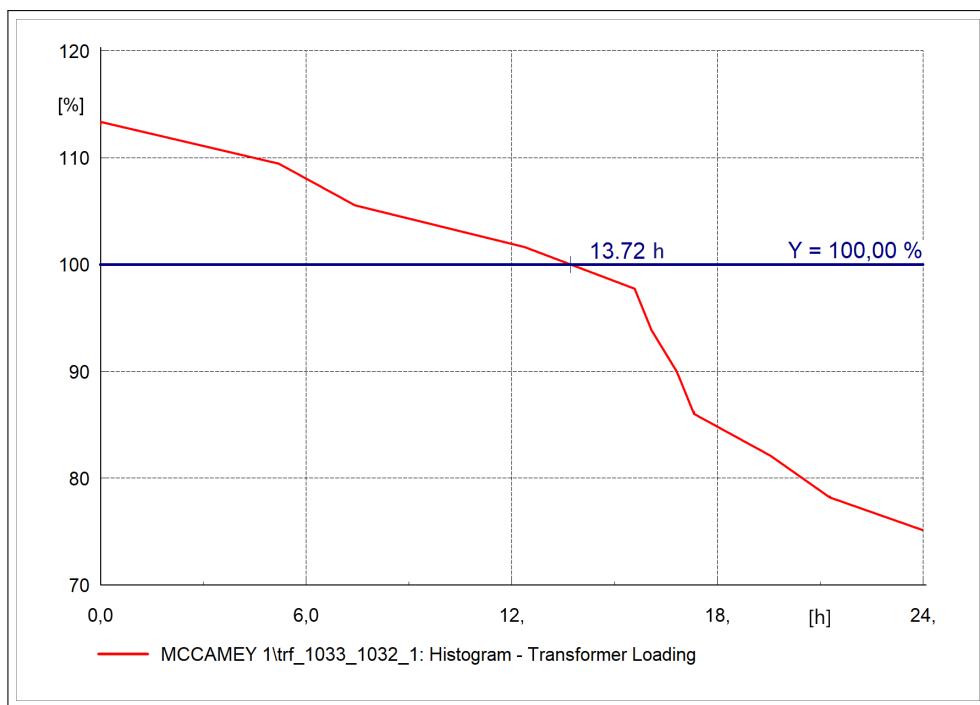


Figure 19.7.14: Duration curve of the loading of a transformer over the duration of a day

The pages on the edit dialog of the plot are the same as for the curve plot described in section [19.7.1](#).

In addition the *Function* for the definition of the **Curves** has to be specified. Several different options can be selected:

- For the **Method** that is used to determine the duration curve *Histogram* and *Bootstrapping* can be selected.
- The **Curve** can be defined through several different functions which are *Cumulative distribution function*, *Probability density function*, *Quantile function* and *Mirrored Quantile function*.
- The different methods that are used for the **Estimation** are *User-defined*, *Automatic parameter deduction*, *Rice Rule*, *Freedman-Diaconis choice* and *Scotts Rule*.
- Different **Units** can be chosen for displaying the duration plot which are *Probability*, *Probability [%]* and *Duration*.
- With the additional option *Limit range to input data* only the current input data is considered.

## 19.7.12 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:

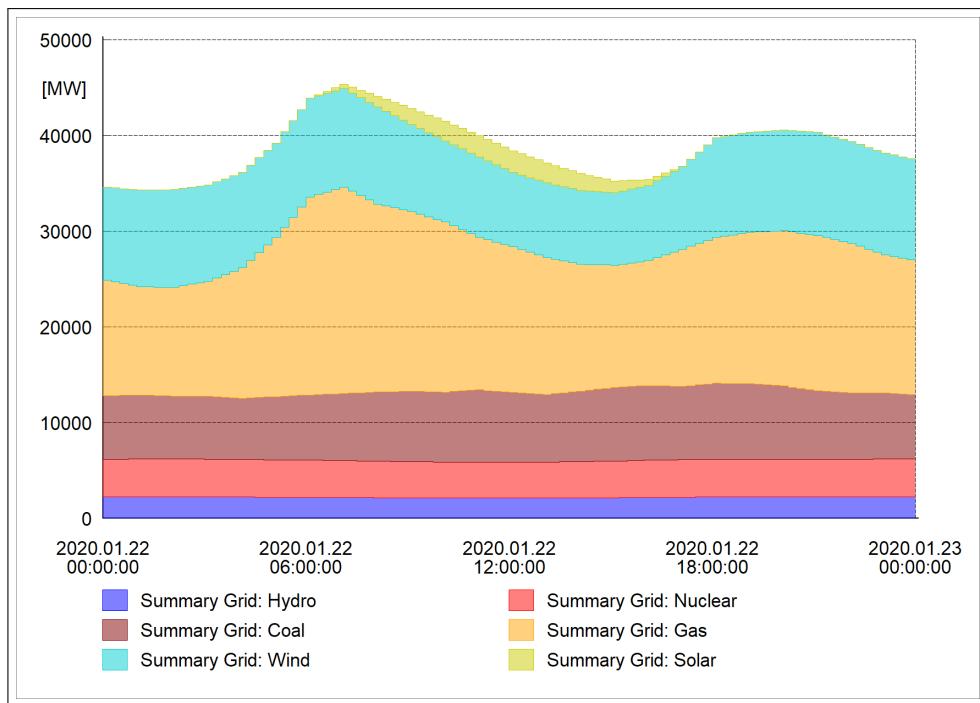


Figure 19.7.15: Total generation in a grid divided into different plant categories

The pages on the edit dialog of the plot are the same as for the curve plot described in section 19.7.1, except for the settings on the page for the *Y-Axis* defining the *Variables*.

In the y-axis page of the energy plot dialog, the *Variables* tab includes the following additional fields:

- **Automatic:** the colour and brush style of the new bars in the energy plot will be set automatically when the corresponding option is enabled. The Apply button will apply automatic brush formats to all existing bars again.
- **Results:** this is a reference to the currently active results file (*E/mRes*). With the selection *Search automatically* the currently used result file of the last calculation is chosen. The option *User defined* gives the user the opportunity to choose another result file. The selected object will be used, if no results file is specified in the *Curves* table.
- **Curves:** the definition table for the curves is fundamentally different to the one that is usually used for other plots because it always shows all available *Plant Categories* by default.
  - 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.
  - In the column *Category* all available plant categories that can be used are shown by default. The first entry is displayed in the plot on the bottom, while the last entry that is defined is displayed on the top.
  - For each category a *Colour* can be defined which is shown in the plot.
  - A *Variable Description* for each entry is available.
  - A *Brush Style* for each defined category can also be modified.
- **Representation:** allows different representations of the shown plot. By selecting *Stacked* the absolute values of the shown variables are used in the plot. The option *Percent* standardizes the values to illustrate the percentage share of the shown variables.

The following options available for the curve plots are not available for the energy plots:

- User-defined signals

- Filter... button
- Export... button
- Automatic scale of the line width
- Representation as curves

### 19.7.13 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 complete generation of different areas in the grid as shown here:

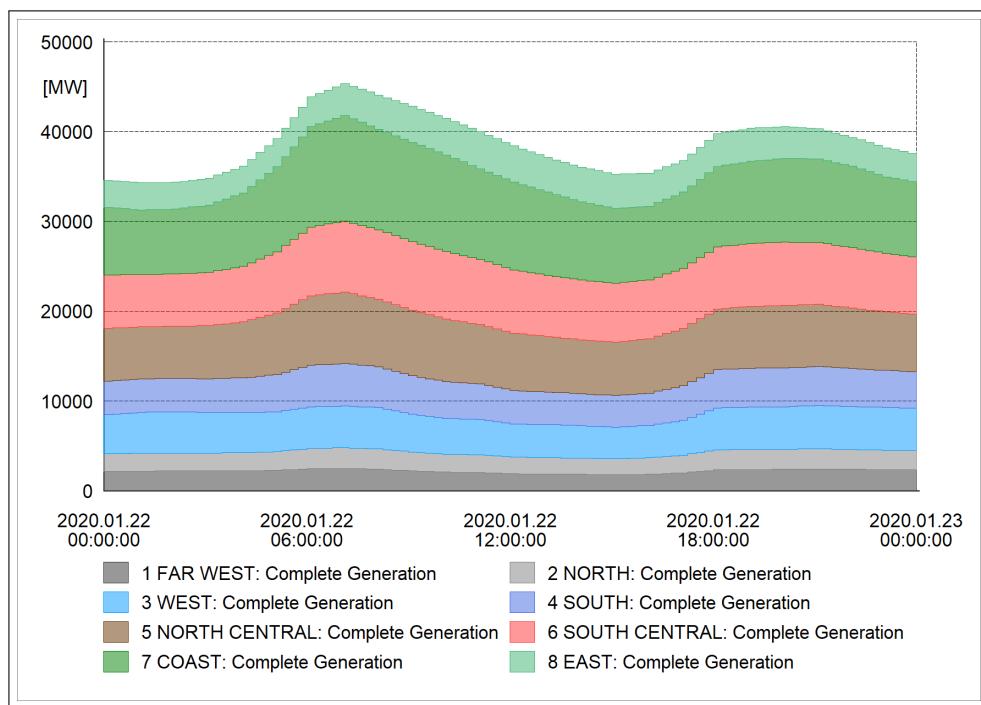


Figure 19.7.16: Total generation in a grid divided into different areas

The pages on the edit dialog of the plot are the same as for the energy plot *Plant Categories* described in section [19.7.12](#).

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.

### 19.7.14 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:

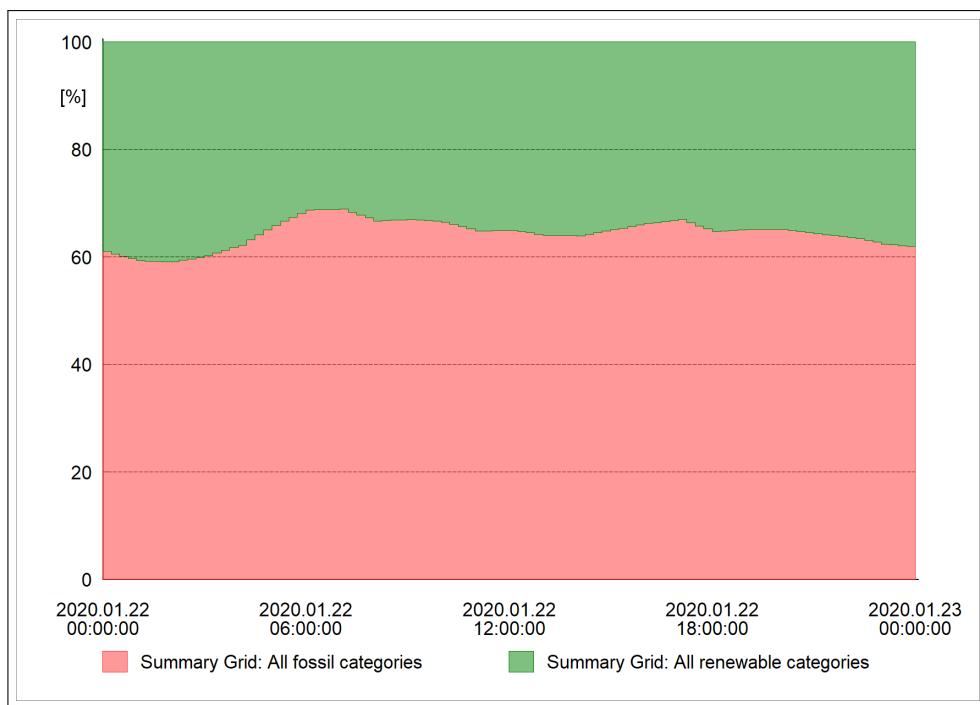


Figure 19.7.17: Total generation in a grid divided into renewable and fossil categories

The pages on the edit dialog of the plot are the same as for the energy plot *Plant Categories* described in section 19.7.12.

The only difference is the default entry for the **Curves** since the plant categories in the grid are divided into fossil and renewable categories.

## 19.7.15 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 19.7.18 and the additional context sensitive menu options are described in the following sections.



Figure 19.7.18: Plots Toolbar

### 19.7.15.1 Insert Plot

The icon inserts a plot in the existing page. Clicking on this button opens the *Insert Plot* dialog, described in section 19.7.

### 19.7.15.2 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 19.7.19 shows an example of the dialog.

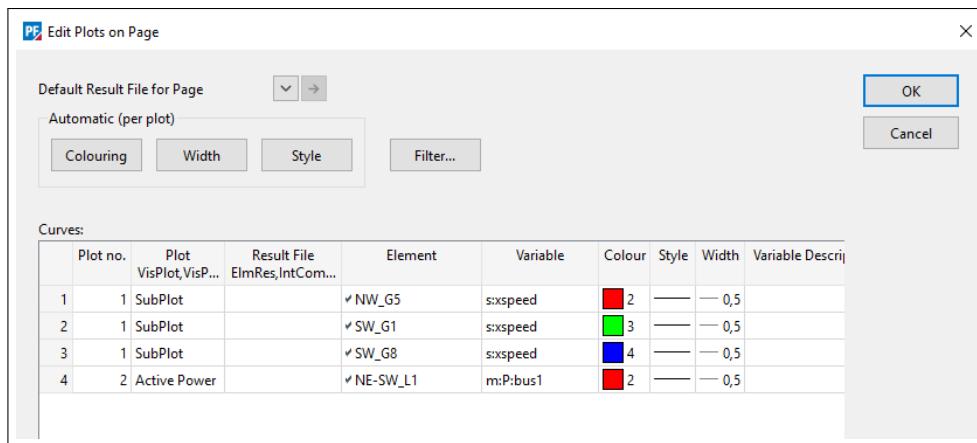


Figure 19.7.19: Editing all plots on the plot page

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 File for Page* is a reference to the results element of the plot page. The **Filter...** button opens the filter dialog. The selected filter will be applied to all plots on plot page.

*Default Results File for Page* is a reference to the default results element of plot page.

### 19.7.15.3 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.

### 19.7.15.4 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

---

#### 19.7.15.5 Scale Buttons

- **Scale x-axis automatically:** scales the x-axis to the start and end of the results file.
  - **Scale y-axis automatically:** scales the y-axis 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.
  - **Zoom y-axis:** zooms in a certain range of the y-axis.
  - **Move x-scale:** moves the position of the x-axis.
  - **Stretch/compress x-scale:** modifies the x-axis scales in order to compress or stretch the shown curve.
  - **Stretch/compress curve:** moves the curve temporarily around a reference point set in the curve.
- 

**Note:** The scale buttons are inactive if there are no plots shown at all or if the x or y axes can not be scaled automatically.

---

#### 19.7.15.6 Labels Buttons

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 visible 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:

- **Statistic Label:** helps to analyse a curve, by labelling, for example, its extrema.
- **Delete Statistic Label:** delete existing statistic label.
- **Text Label:** displays user defined text above and below a line connected to the curve.
- **Value Label:** displays the x/y coordinates of the cross. The label is a text-label filled with the marked coordinates.
- **Gradient Label:** displays the value of the difference between two x- resp. y-values ( $dx$  resp.  $dy$ ) as well as the gradient ( $dy/dx$ ) and the  $1/dx$  value. The label is a text-label filled with the marked coordinates.
- **Format Label:** uses a form to print the displayed text. The form can be selected as local for each label or a common label can be used for all plots of the same type in the active project.
- **Text box:** can be used to display text or tables anywhere in the plot.

### Text, Value and Gradient labels

The text, value and gradient labels are defined using the same object type. The *VisValue* edit dialog contains the following fields:

- *Value*: displays the connected curve position of the label. For labels created as a value-label this position is displayed automatically as label text. “x-Axis” displays the x axis value and “y-Axis” the y axis value. “Time” is only visible for plots showing a trajectory.
- *Text on Top and on Bottom*: text written above and below the horizontal line.
- *Delete Label when a new Simulation is started*: labels in plots showing simulation results are usually automatically deleted when the simulation is started again. To keep labels in such plots, e.g. to compare curves with the last run, this option should be un-selected.

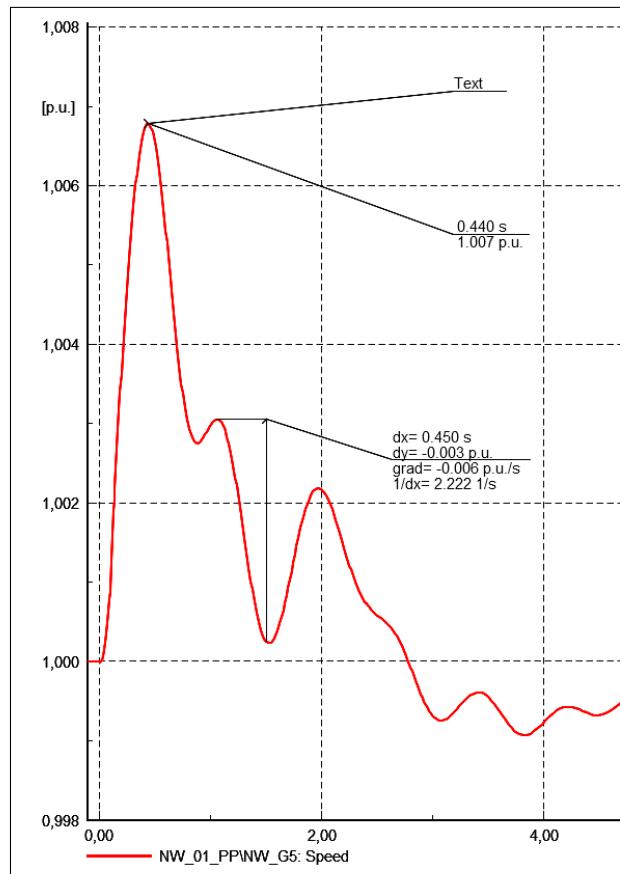


Figure 19.7.20: Text, Value and Gradient labels

### The Format Label

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.

#### Text Box

The text box (*VisText*) can be used to display text somewhere in the plot. After creating the text box, it can freely be moved across the plot. The edit dialog is shown in Figure 19.7.21.

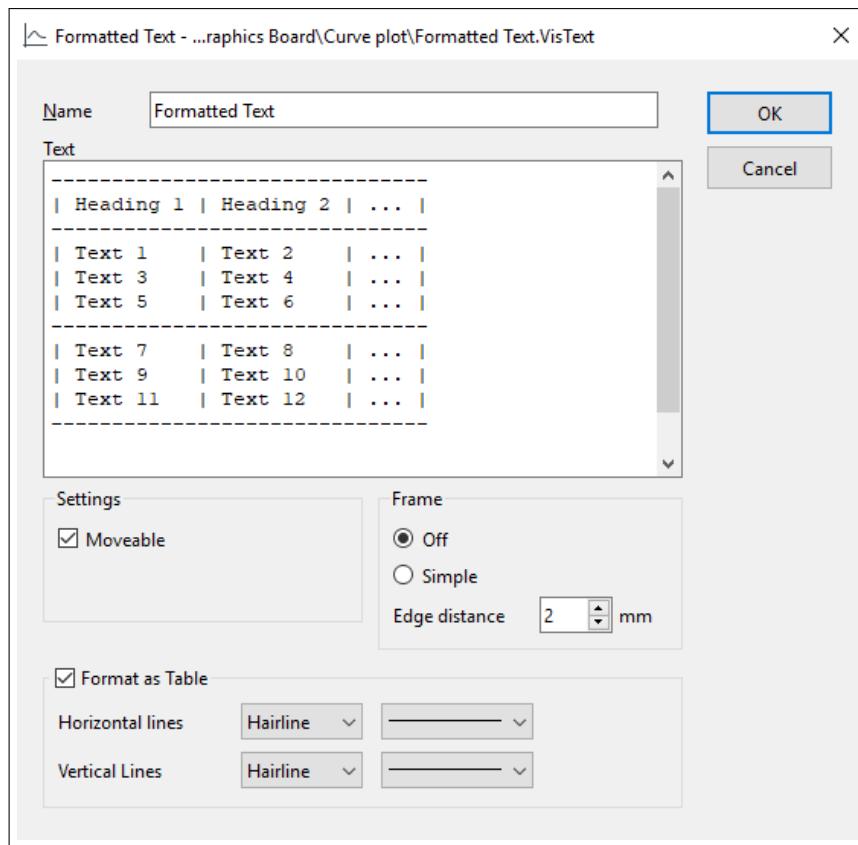


Figure 19.7.21: The text box dialog

- **Name:** name for the text box can be inserted, under which the object is saved.
- **Text:** the required text. A table will be created, if the option *Format as Table* is activated and the text is formatted.
- **Settings:** the *Moveable* option to allows/prohibits the movement of the text box.
- **Frame:** displays a simple frame (single solid line) around the text box and defines the distance between the edge of the frame and the text or hides the frame.
- **Format as Table:** the inserted text will be formatted as a table, if the *Format as Table* option is activated and the characters “-” and “|” are used in the form as shown in Figure 19.7.21. The horizontal and vertical line types, i.e. the line width and the line style, can then be specified.

### 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

To remove statistic labels from a curve, the button *Delete Statistic Labels*  can be used.

### 19.7.15.7 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 [29.13](#).

### 19.7.15.8 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.

### 19.7.15.9 Title Block

The icon  shows or hides the title block. The title can be defined or changed by double-clicking on it.

All plot pages in a Graphics Board show the same title by default. The only difference of the title blocks on the plots pages are the panel name and the page number which are unique for each plot page. A local title can be created by right-clicking on the title and selecting *Create local Title* from the context sensitive menu.

For details about the settings of the title object refer to Chapter [9](#): Network Graphics.

### 19.7.15.10 Edit Plot Page

Whenever a plot is inserted, a Plot Page is automatically created, the Plot Page being one of the possible page types on a Graphics Board. The edit dialog can be opened by clicking on the  icon and the pages are described below.

#### x-Axis

Holds the default x-Axis for plots without local axis stored in pages without local axis. The options are the same as described in section [19.7.1](#), on page [277](#).

#### Advanced

- *Arrangement*: this option has the same effect of using the icons described in section [19.7.15.4](#). If the option *User Defined* is selected, the plots can be resized and moved inside the panel as desired.
- *Plot style*: the style used for all the plots in the plot page can be selected in this field; more information is available in section [19.7.15.12](#).
- *Background*: the default background of the plot page is empty. The *Filename* defines the background file. If the selected file does not exist, or the filename not set, the background remains empty. *Graphics are transparent* must be activated to make all graphics transparent. If an opaque graphic fills the complete panel area the background will be invisible.

#### Results

On this page, it is possible 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.

### 19.7.15.11 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 graphics board.

In addition a local page format can be created for each Plot Page by selecting the option *Create local Page Format* from the context sensitive menu.

#### 19.7.15.12 Current Styles

Each plot page uses a style where line-widths, fonts, brushes and other graphical settings are defined. There are some predefined styles available in *PowerFactory*, which are:

- Default - Standard Text and Symbols
- Paper

The *Default* style uses smaller line-widths and smaller fonts than the *Paper* style. It was designed to get satisfactory printouts. The paper style was designed for reports and papers where plots are included in text-programs.

User-defined styles can also be created. This process is described in detail in section [19.7.18](#) later in this chapter.

#### 19.7.15.13 Export Diagram

The *Export Diagram* icon  opens a dialog offering options for exporting the graphic. A range of file formats is supported.

#### 19.7.15.14 Print Preview

The *Print Preview* icon  opens the print preview page. The printer and margins to be used can be selected in this dialog.

### 19.7.16 Context Sensitive Menu Tools

As well as the tools of the plot toolbar, the following additional tools are available via the context sensitive menu, displayed by right clicking on the plot.

#### Create Full Size

This option will create a new Plot Page containing only the selected plot.

#### Set Constant

The constant label is used to display a straight line. It can be used to display y-values for a constant x-quantity or x-values for a constant y-quantity.

The look of constant labels can be varied with the following settings:

- **Style:** 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.
- *Intersection Only*: shows only the intersection points with the curves.
- **Label:** 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.
- **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.
- **Linestyle and Width:** specifies the line style and line width for the line shown. Invisible if *Show Values* is set to *Intersections Only*.

### Straight Line

The *Straight Line → ...* option includes the following options:

- **Set Secant:** adds a line directly through the selected data point.
- **Through Point:** defines a graphic line through the selected data point with a defined gradient and gives back the function of the line.
- **User Defined:** defines a line independent from the curves shown with a defined gradient and y-offset. The function of the inserted line can also be seen, when holding the mouse arrow over the line for 1 second. The options of the line dialog is similar to the options for the constant value.

### 19.7.17 The Status Bar

In the status bar of *PowerFactory* on the bottom of the program window useful information regarding the data shown in the curves can be obtained.

- First the value of the mouse position in the diagram is displayed in the status bar, similar to the information shown with an open single line diagram.
- When a curve is clicked and marked with a cross, the cross value is displayed in the status bar and remains unchanged until the cross is set to a different position. If there is no cross on the active page the status bar value is reset and no longer displayed. Some plots have different scales on one axis; these plots can not display a value in the status bar.
- The option *Curve-Tracking* can be found in the status bar, normally in grey. Double-clicking on this enables the “Curve-Tracking” mode. In this mode a cross will appear if the mouse arrow is near a curve. Holding the mouse pointer still for one second will show a balloon window with the x- and y-value.

### 19.7.18 User-Defined Styles

The user-defined styles are stored in the settings folder element of the active project. There are two folders where the styles are stored, one for the Plot Page and one for the Plot object. A new style is created by right clicking on the plot or on the plot page and selecting *Style → Create New Style*. Once a user-defined style is created, the option *Style → Edit Style* of the context sensitive menu can be selected to open the dialog of the new style.

The user defined styles dialogs for the plot page and plot are described below.

#### 19.7.18.1 User-Defined Styles for the Plot Page

Figure 19.7.22 shows the dialog for editing the layout of the panel.

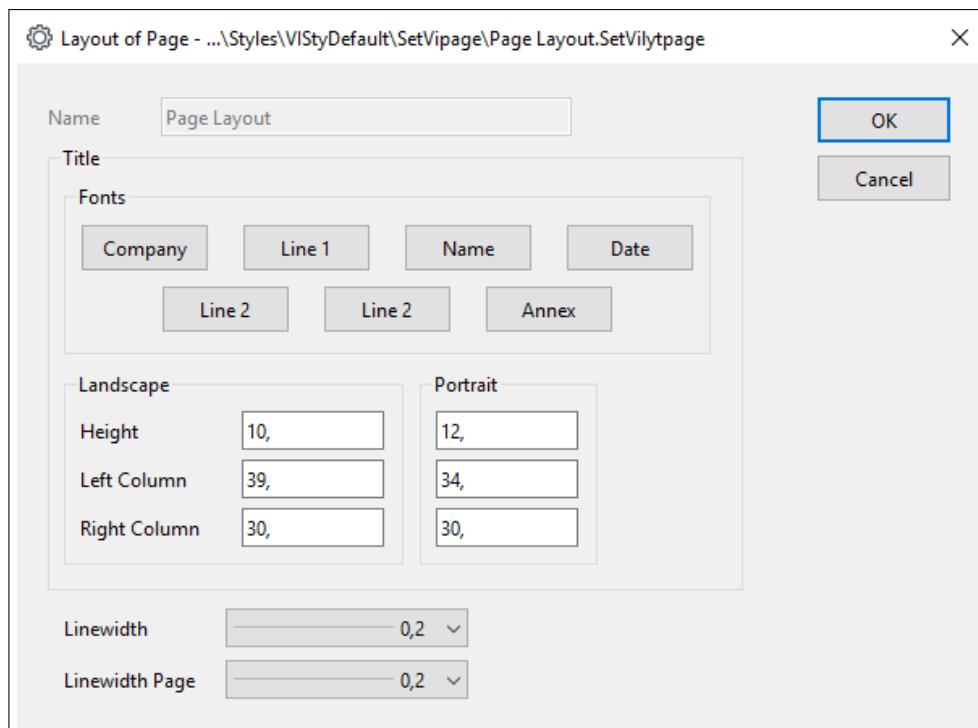


Figure 19.7.22: Editing the Plot Page style

In this dialog it is mainly the layout of the title block of the Plot Page that is edited. The following settings can be defined:

- the different font styles for the various entries of the block by clicking on the **buttons**
- the height and the width of the columns of the title block
- the line width of the title block and of the page frame

#### 19.7.18.2 User-Defined Styles for Plots

There is the possibility to define the x- and y-axis of the plots inside on one page. These settings are then valid for every plot on panels using this style. When selecting the option *Style → Edit Style* from the plot a window will be shown, containing the settings for:

- all x-axis of plots using this style

- all y-axis
- the selected plot object (*VisPlot*)

When double-clicking on the object which is to be changed, the dialog of the selected axis will be opened as shown in the following figure and can then be modified.

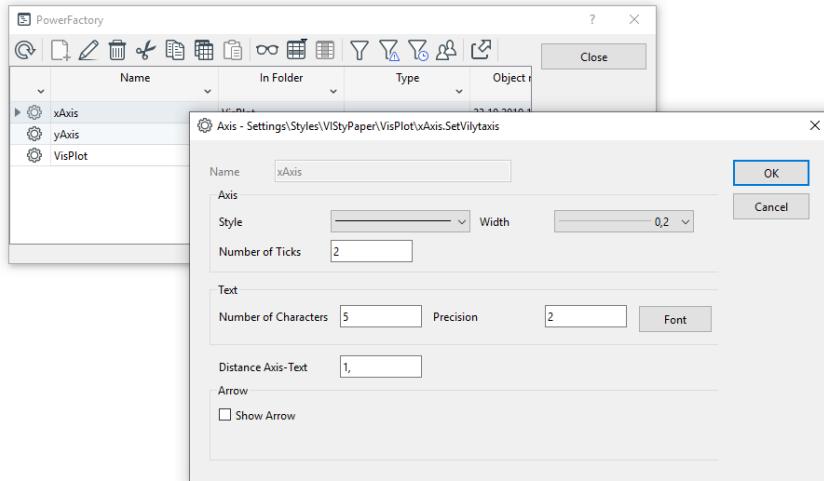


Figure 19.7.23: Editing the styles of X-axis

In the axes dialogs the following settings can be specified for the selected style:

- **Axis:** here the style and width of the axis itself can be changed. Also the number of small ticks shown between the divisions can be chosen.
- **Text:** the number of characters and the digits behind the decimal point as well as the font type and size can be specified.
- **Distance Axis-Text:** distance between the axis and the text
- **Arrow:** the representation can be altered between the normal style and a style with an arrow at the end of the axis with a certain width and length of its tip.

The presentation of the plot itself can be chosen by modifying the plot object (*VisPlot*). The settings available for the plot object are:

- **Grid:** options to alter the width, line style and colour of the main grid and the help grid.
- **Legend:** to edit the distances from the legend to axis and between the different legends.
- **Margins:** to set spaces between the diagram and the surroundings.

# Chapter 20

## Data Extensions

### 20.1 Introduction

Introduced in *PowerFactory* 2018, 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.
- 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; the use of characteristics with such parameters, however, is not possible.

To add a new attribute to a flexible data page (see [10.6](#) 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.

## 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 or snapshot export .dzs files, but will not be retained if the older .dz export is used.

# 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).

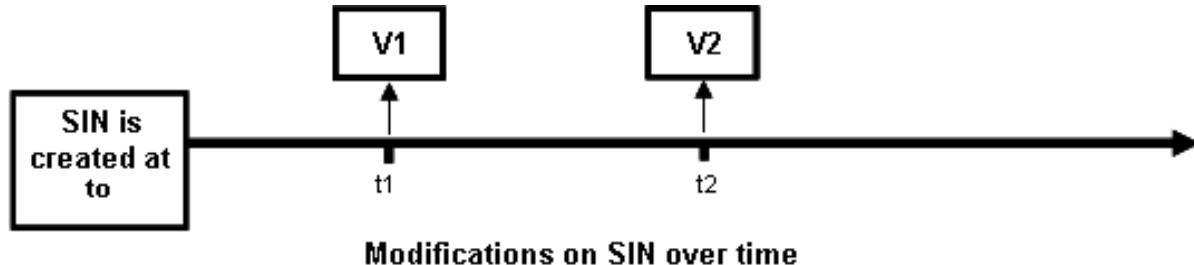


Figure 21.2.1: Project versions

## 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-sensitive 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 called 'V0', created at a point in time,  $t$ . If a *Rollback* to 'V0' is completed, the project returns to its state at the creation of 'V0'. After the *Rollback*, all changes implemented after 'V0' (i.e. after V0's point in time) are deleted. Also, all versions newer than 'V0' are removed. This concept is illustrated in Figure 21.2.2.

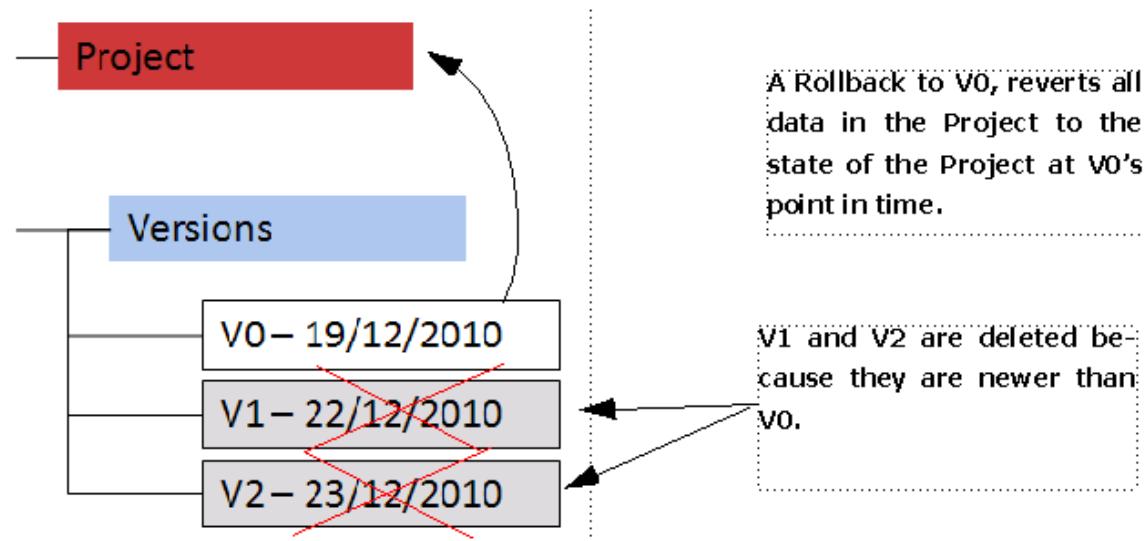


Figure 21.2.2: Example of a rollback

#### 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-sensitive menu.
3. Press **OK** when the confirmation message appears.

Note that a *Rollback* is not allowed (and therefore not enabled in the context-sensitive 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 *Output... → 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, *DIGSILENT* 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 8.1.4 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.

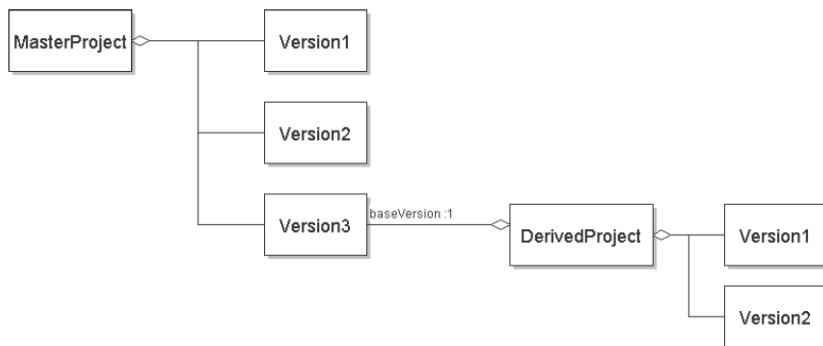


Figure 21.3.1: Principle of derived projects

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.

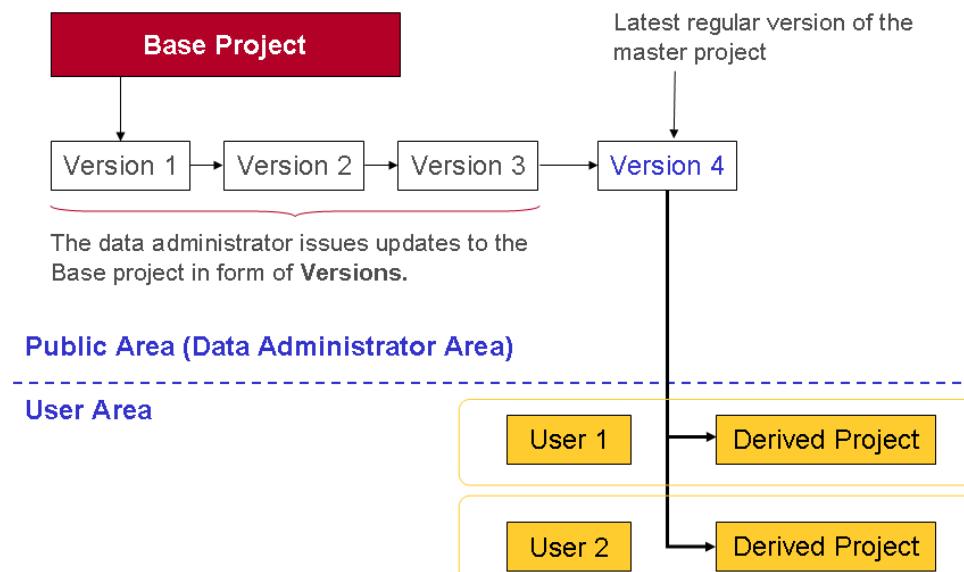


Figure 21.3.2: Derived projects in a multi-user database

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.

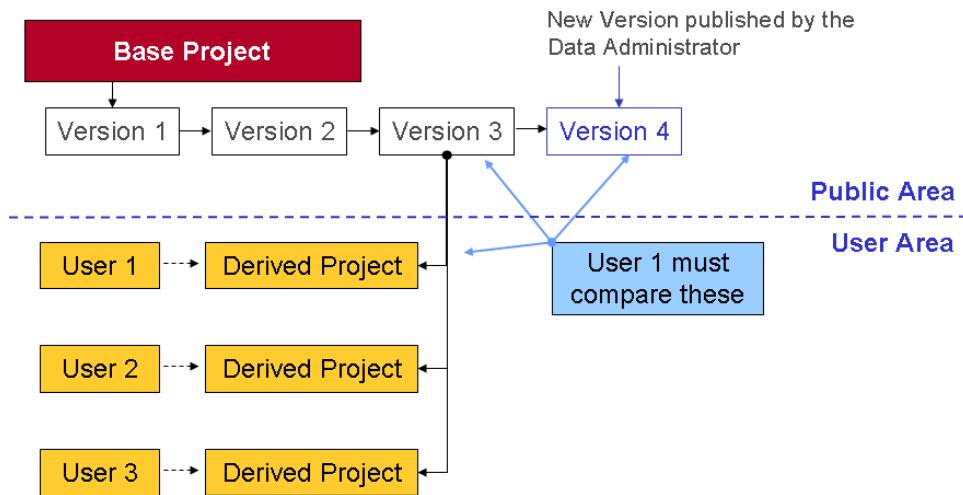


Figure 21.3.3: New version of base project in a multi-user database

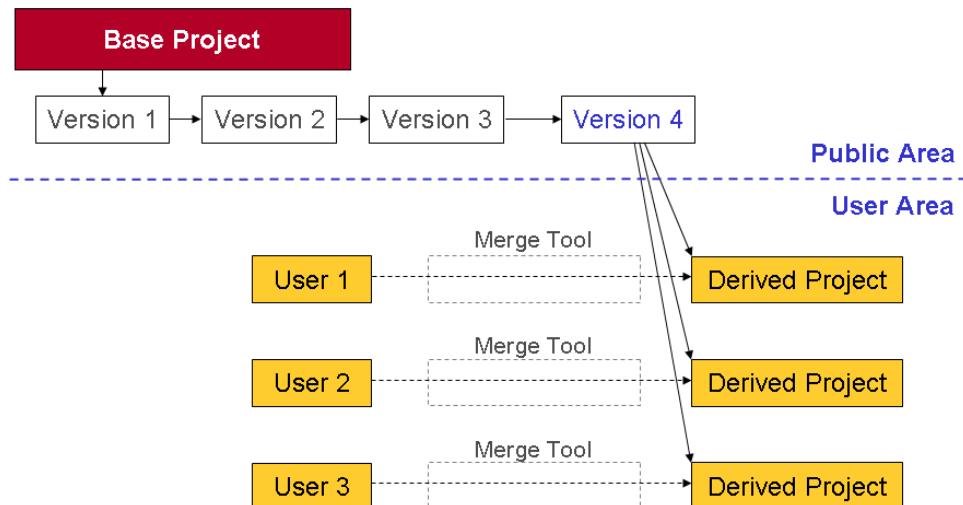


Figure 21.3.4: Merging the new version of the base project into derived projects

### 21.3.2 How to Create a Derived Project

A derived project is created using the Data Manager as follows:

1. Right-click on the desired folder in the *right pane* of the Data Manager where the derived project is to be created.
2. Select *New → Derived Project* from the context-sensitive menu.
3. Select the source version of the base project using the data browser that appears. This will likely be the last available version of a project in a public area, created by the data administrator.
4. Press **OK**.

- Note:**
- The base or master project has to have at least one version before other projects can be derived from it.
  - A project cannot be derived from a 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.
- 

After the derived project is created, it can be used as a normal project.

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.

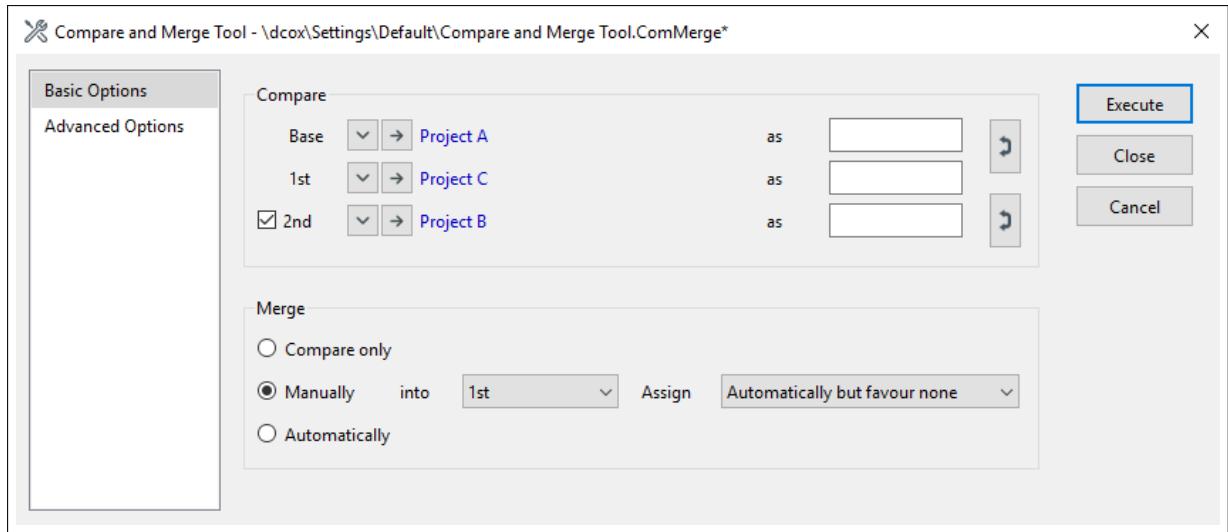


Figure 21.4.1: Compare and Merge Tool options dialog for a three-way merge

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

##### 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 *Identify correspondents always by name/rules* option is also enabled.

#### 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.

#### 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 <

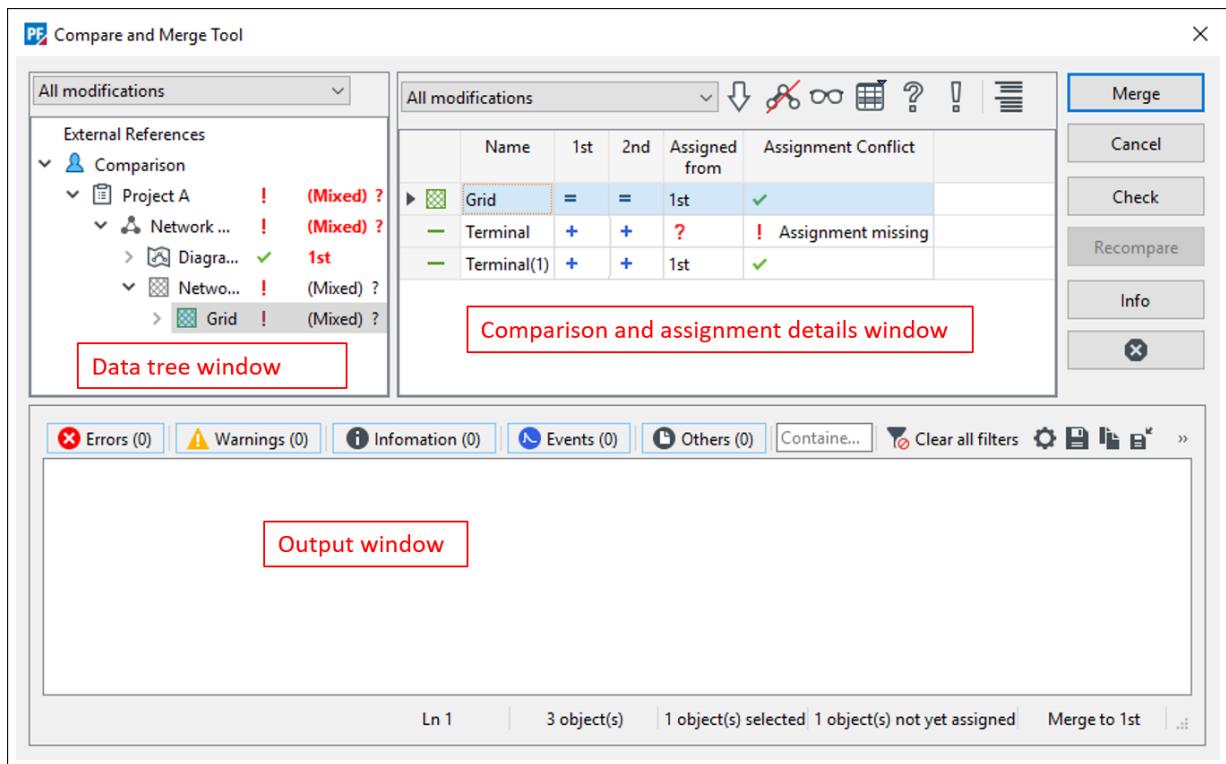
This field defines the tolerance of the comparison engine when comparing numerical parameters. If the difference between two numerical parameters is less than the value entered into this field, then the comparison will show the two values as equal, =.

### 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

### Output Window

The output window displays reports from the context-sensitive (right-click) menu, and other error information.

### How to use the Comparison and Assignment window

In the CMT 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-sensitive (right-click) menu.

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.

### 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

### 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** The functionality of these two buttons is identical to their function in the Data Manager.

**Show only not assigned** Filters the display to show only objects not yet assigned. This filter is only available when the merge option is used. By default all assigned and unassigned objects are displayed.

**Show only Objects with assignment conflicts** Only objects with assignment conflicts are displayed. This filter is only available when the merge option is used. By default objects with and without assignment conflicts are displayed.

**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).

#### Diff window right-click menu options

A context-sensitive 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.

### Diff window buttons

The various diff window buttons (as highlighted in Figure 21.4.4) will now be explained.

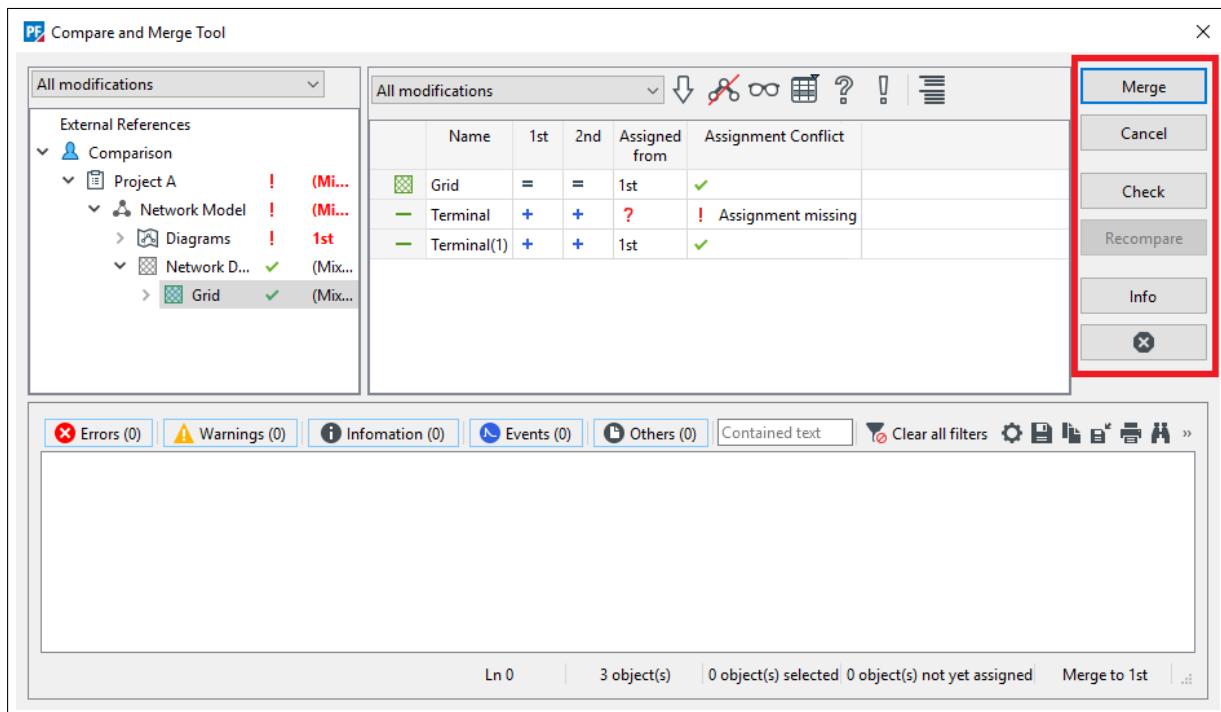


Figure 21.4.4: Compare and Merge Tool 'Diff window' with buttons highlighted

**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.

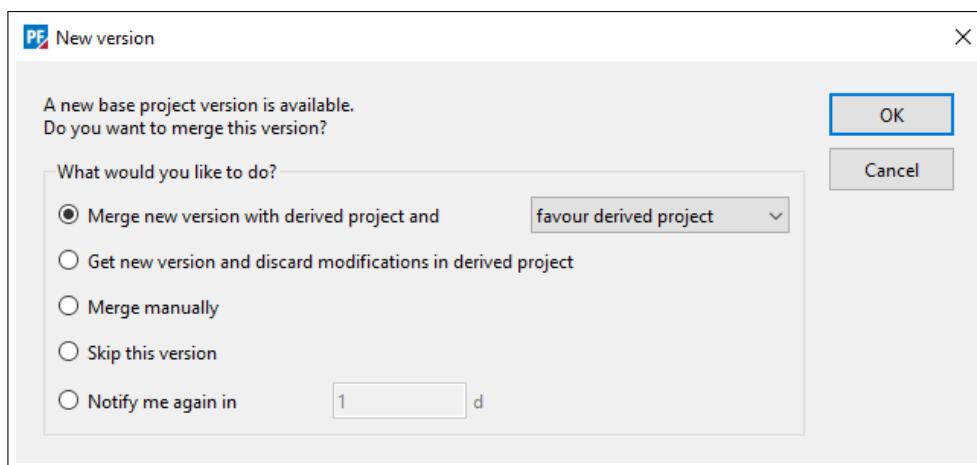


Figure 21.5.1: New version available dialog

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-sensitive 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-sensitive 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 and the data administrator, refer to Chapter 6 (User Accounts and User Groups).

## 21.7 Combining Projects

In version 2017 of *PowerFactory*, a new tool was introduced which enables two or more projects to be combined. It 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.

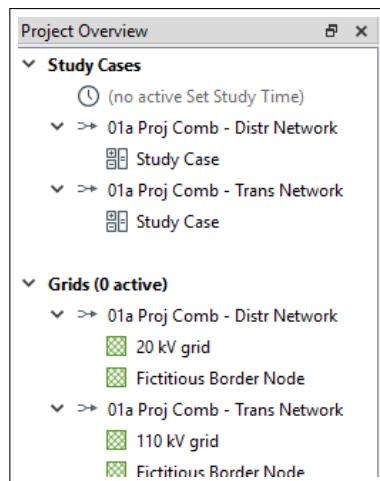


Figure 21.7.1: Structure of combined project

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 to “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.

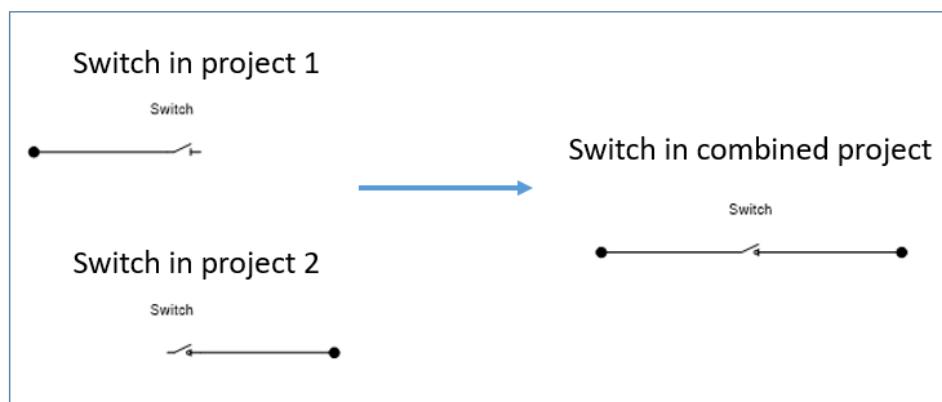


Figure 21.7.2: Network connection using paired switches

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-sensitive (right-click) menu 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 5 Program Administration, Section 5.6: 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 [22.2.2](#) 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 the *Parallel Computing Manager* object, refer to Section [22.4](#).

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* → *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 the elements category *Others* and type in “ComTasks”. Click 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:

- Basic Options
- Parallel Computing
- Output.

### 22.2.1 Basic Options 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.

**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 via the **Add** button. As a prerequisite, each selected command must be located within the referenced study case folder. The **Remove all** button removes all items within the current list. 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.

**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*.

---

## 22.2.2 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 ComTasks will be executed sequentially.

**Parallel Computing Manager:** A reference to the Parallel Computing Manager which administrates the parallelisation settings is added. The Parallel Computing Manager is described in Section 22.4. If the *Parallel Computation* checkbox is ticked, the number of processor cores that are practically used by *PowerFactory* (as configured in the Parallel Computing Manager and dependent on the local machine characteristics) is displayed. Clicking the *Edit* button (→) next to the *Parallel Computing Manager* reference will open the parallel computing configuration object. Further details on the settings used in this object are presented in Section 22.4.

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.

---

**Database changes of parallel processes:** the radio-button *Database changes of parallel processes* defines, which data shall be transferred from the parallel process to the master process and merged into the database:

- If *Merge all changes to master process* is selected, all changes, which 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.
- If *Transfer only results files to master process* 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.

---

**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.

---

### 22.2.3 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 22.2.3. 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\textbackslash Workspace.nnnnnnn\db-process-1\log*). These files provide further information on the execution details of each parallel process.

## 22.4 Parallel Computing Manager

When creating a new Task Automation command or any other command which supports parallel computation (e.g. *Contingency Analysis*, *Quasi Dynamic Simulation*, *Reliability Analysis*, etc.), *PowerFactory* links the specific command to a Parallel Computing Manager object (e.g. as seen in the *Parallel Computing* page of the Task Automation command dialog). This object contains the necessary settings for the parallel computation of tasks and by default it is located in within the *PowerFactory* database under “/System/Configuration/Parallel Computation”. This object has read-only rights for a non-administrator *PowerFactory* user account: it can be used by the Task Automation command (or any other command which supports parallel computation) but cannot be edited by the normal user. However, it is possible for a user to customise the Parallel Computing Manager settings in order to use fewer cores than the maximum defined by the Administrator. See Section 7.11 for details.

Alternatively, it is possible to create a user defined settings object by following the steps below:

- Log in *PowerFactory* using the administrator account;
- Using the Data Manager, verify under the “/Configuration” folder if there exists a subfolder named “Parallel Computation” whose key (parameter *loc\_name*) is “Parallel”. If it does not exist then create a new folder, give it a suitable name and assign the “Parallel” key to it;
- Create a new *Parallel Computing Manager* object (“\*.SetParalman”) under the newly created system folder (e.g. “Parallel Computation” folder) and give it a suitable name;
- Edit and modify corresponding settings by opening the Parallel Computing Manager Dialog;
- Log out of the *Administrator* account and log in with the normal *PowerFactory* user account;
- Go back to the “Parallel Computing” page of the specific command and notice the changed reference to the newly created *Parallel Computing Manager* object.

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

---

**Note:** The Parallel Computing Manager settings can be changed only by the Administrator account.

---

### 22.4.1 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.

### 22.4.2 Communication page

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

# 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.

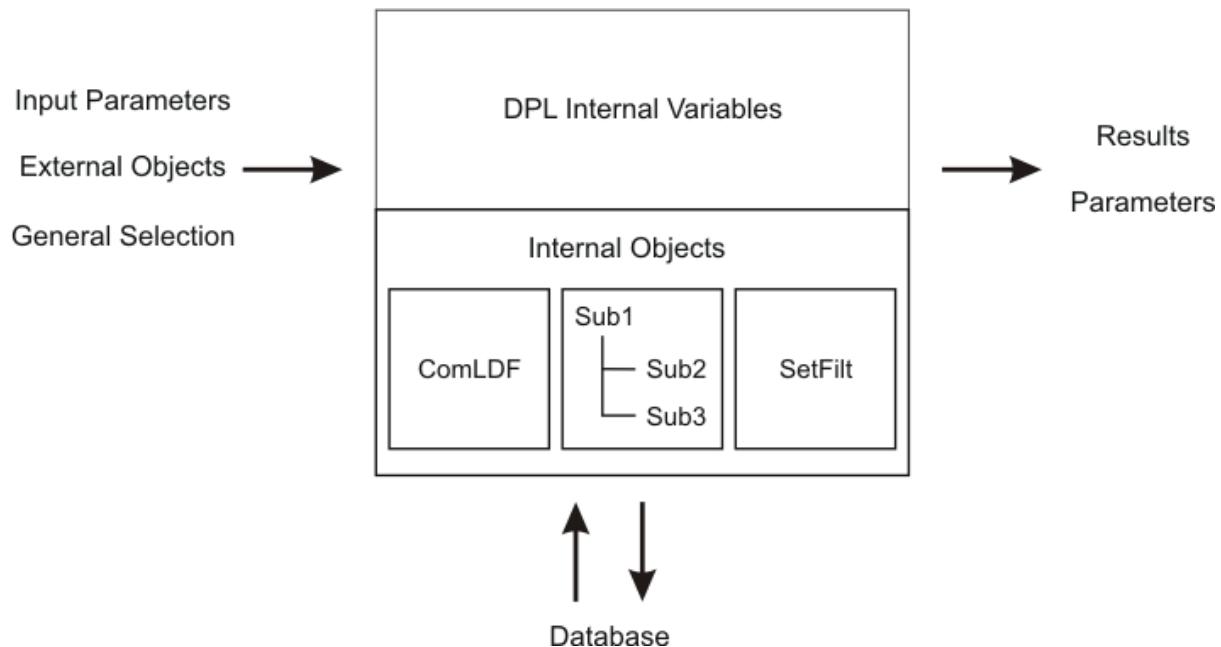


Figure 23.1.1: Principle structure of a DPL command

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.

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 selecting *DPL Command and more*. Then press **OK** and 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 sensitive 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 *Define... → DPL Commands Set...* from the context sensitive menu.
- It is also possible to select several elements in the Data Manager. Right-click the selection and choose the option *Define... → DPL Commands Set...* from the context sensitive 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 sensitive menu.
- Then select a DPL script from the list. This list will show DPL scripts from the global as well as from the local library.
- Select a DPL script, insert/change the variables and then press the button **Execute**

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 , selecting “other” and the element Set (*SetSelect*) and then selecting the set type.

---

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 encrypt-action affects the script for which it is executed and does not create an encrypted copy of the ComDpl-object.

---

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.

### 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:

```

1 statement1;
2 statement2;
3 if (condition){
4     groupstatement1;
5     groupstatement2;
6 }
```

#### 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:

```

1 double Losses, Length, Pgen;
2 int NrOfBreakers, i, j;
3 string txt1, nm1, nm2;
4 object O1, O2, BestSwitchToOpen;
5 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:

```
1 variable = expression;
2 variable += expression;
3 variable -= expression;
```

The add-assignment “`+=`” adds the right side value to the variable and the subtract-assignment “`-=`” subtracts the right-side value.

Examples:

```
1 double x,y;
2 x = 0.5*pi();      ! x now equals 1.5708
3 y = sin(x);        ! y now equals 1.0
4 x += y;            ! x now equals 2.5708
5 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:**

```

1 if (a<3) {
2     b = a*2;
3 }
4 else {
5     b = a/2;
6 }
7 while (sin(a)>=b*c) {
8     a = 0:dline;
9     c = c + delta;
10}
11if ({.not.a}.and.{b<>3}) {
12    err = Ldf.Execute();
13    if (err) {
14        Ldf:iopt_lev = 1;
15        err = Ldf.Execute();
16        Ldf:iopt_lev = 0;
17    }
18}
19for (i = 0; i < 10; i = i+1){
20    x = x + i;
21}
22for (o=s.First(); o; o=s.Next()) {
23    o.ShowFullName();
24}

```

### Break and Continue

The loop statements “do-while” and “while-do” 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” or “while-do” 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” or “while-do” 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:**

```

1 O1 = S1.First();
2 while (O1) {
3     O1.Open();
4     err = Ldf.Execute();
5     if (err) {
6         ! skip this one
7         O1 = S1.Next;
8         continue;
9     }
10    O2 = S2.First();
11    AllOk = 1;
12    DoReport(0); !reset
13    while (O2) {
14        err = Ldf.Execute();
15        if (err) {
16            ! do not continue
17            AllOk = 0;

```

```

18         break;
19     }
20     else {
21         DoReport(1); ! add
22     }
23     O2 = S2.Next();
24 }
25 if (AllOk) {
26     DoReport(2); ! report
27 }
28 O1 = S1.Next();
29 }
```

### 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:

```

1 double diameter;
2 input(diameter, 'enter diameter');
3 printf('the entered value = %f',diameter);
```

The example results in the pop up of a window as depicted in Figure 23.1.2.

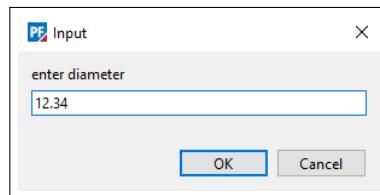


Figure 23.1.2: The input window

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 DPL 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:

```
1 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

```
1 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

```
1 00. int error;
2 01. double loading;
3 02. Ldf:iOpt_net = 1; ! force unbalanced
4 03. error = Ldf.Execute(); ! execute load-flow
5 04. if (error) {
6 05.   exit();
7 06. } else {
8 07.   loading = Line:c:loading; ! get line loading
9 08.   printf('loading=%f', loading); ! report line loading
10 }
```

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.

```

1 00. int error;
2 01. error = Ldf.Execute();
3 02. if (error) {
4 03.   printf('Load-flow command returns an error');
5 04.   exit();
6 05. }
```

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

```

1 00. int error;
2 01. double maxi;
3 02. object O, Omax;
4 03. set S;
5 04.
6 05. error = Ldf.Execute();      ! execute a load-flow
7 06. if (error) exit();        ! exit on error
8 07.
9 08. S = SEL.AllLines();       ! get all selected lines
10 09. Omax = S.First();        ! get first line
11 10. if (Omax) {
12 11.   maxi = Omax:c:loading; ! initialise maximum
13 12. } else {
```

```

14.     printf('No lines found in selection');
15.     exit();                                ! no lines: exit
16. }
17. O = S.Next();                            ! get next line
18. while (O) {                             ! while more lines
19.     if (O:c:loading>maxi) {
20.         maxi = O:c:loading;      ! update maximum
21.         Omax = O;                ! update max loaded line
22.     }
23.     O = S.Next(); 
24. }
25. printf('max loading=%f', maxi); !print results
26. 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. Good examples are load-flow and short-circuit commands, or the vector and matrix objects that the DPL command uses for its computations.

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.

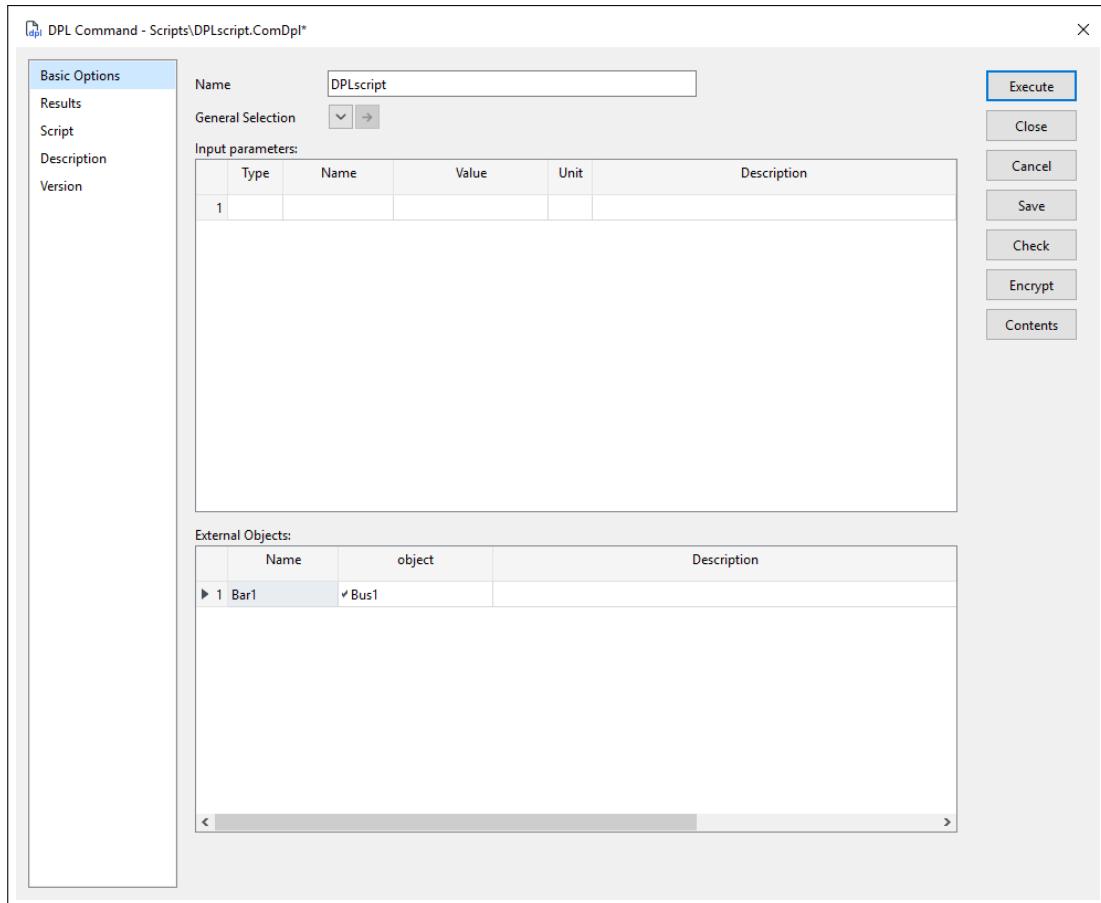


Figure 23.1.3: DPL external object table

Example:

```
1 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. 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.

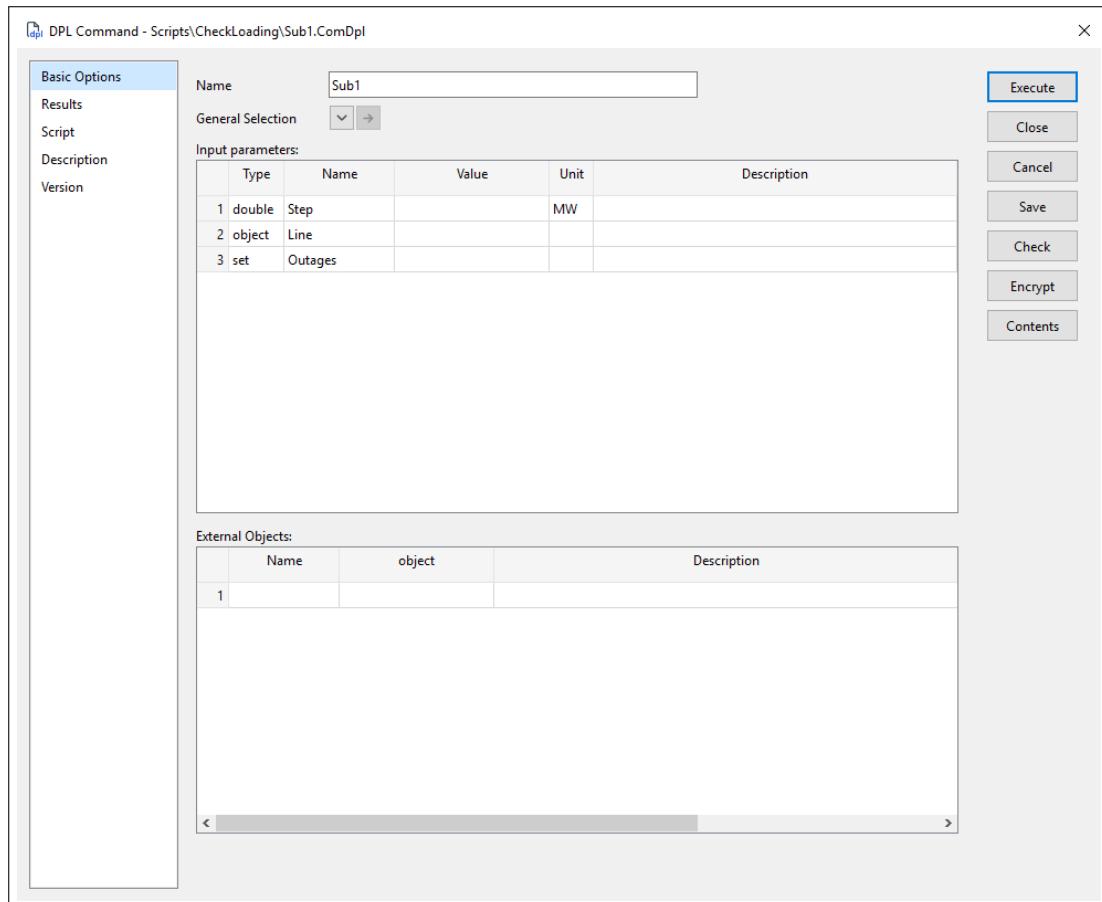


Figure 23.1.4: Interface section of subroutine

The calling command may then use, for example:

```

1 ! set the parameters:
2 Sub1:step      = 5.0;
3 Sub1:Line      = MyLine;
4 Sub1:Outages   = MySelection;
5 ! execute the subroutine:
6 error = Sub1.Execute();

```

However, using calling arguments, we may also write:

```

1 ! execute the subroutine:
2 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.

## 23.2 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 Table 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](#).

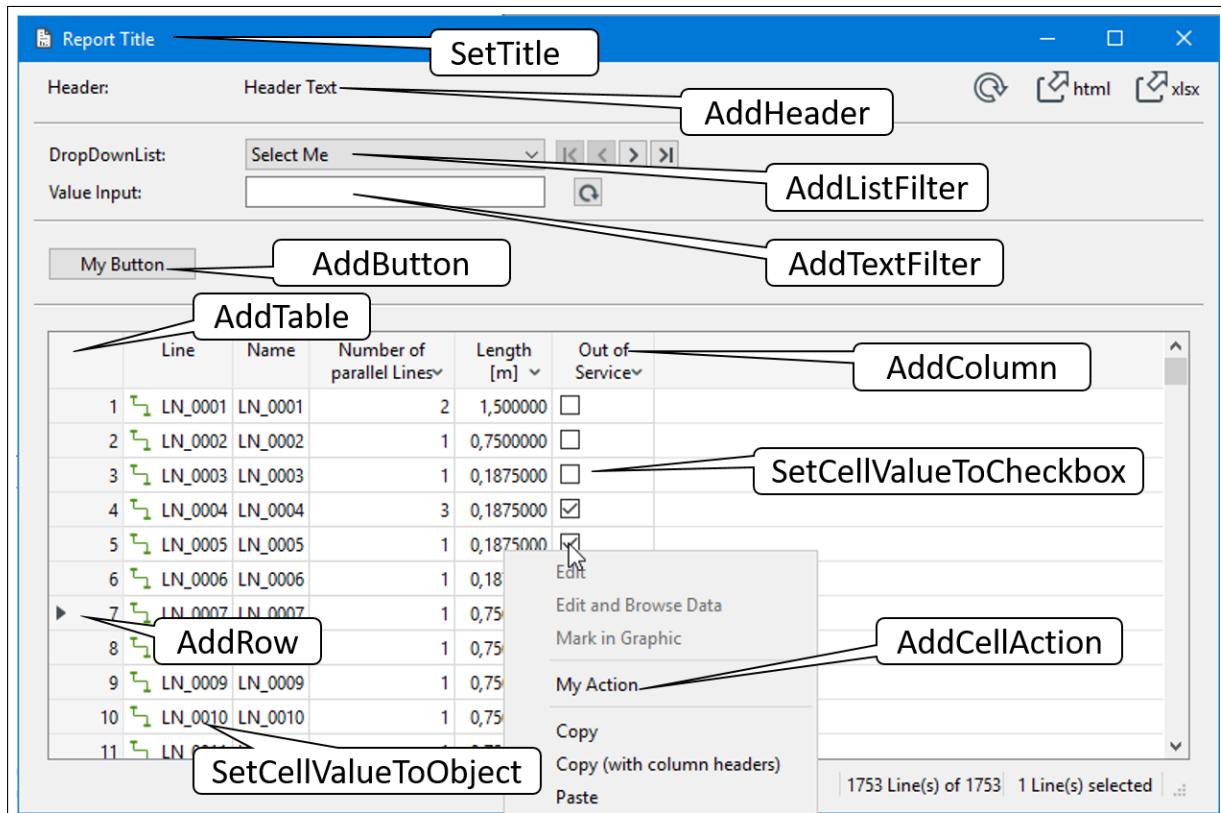


Figure 23.2.1: ComTablereport functions

### 23.2.2.1 Creating a new Table Report Command

A tabular report command element *ComTablereport* can be created by using the *New Object* ( icon) in the toolbar of the Data Manager and selecting *Others*. If the *Filter* property is set to *Commands (Com\*)*, it is possible to select *Table Report (ComTablereport)* from the *Element* list.

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 [23.2.3](#) for an example. Other *ComDpl* functions (see [23.2.6](#)) may be added optionally.

### 23.2.2.2 Executing a Table 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 “Execute Table 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:**

```

1 object oReport;
2 oReport = this.GetParent();
3 oReport.AddTable('Table ID');
4 oReport.AddColumn('Table ID', 'Column 1 ID', 'First Column');
5 oReport.AddColumn('Table ID', 'Column 2 ID', 'Second Column');
6 oReport.addRow('Table ID', 'Row 1 ID');
7 oReport.SetCellValueToString('Table ID', 'Column 1 ID', 'Row 1 ID', 'Hello');
8 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 [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:**

```

1 set      aLines;
2 object   oReport,
3         oLine;
4 string   sRowId;
5
6 oReport = this.GetParent();
7 oReport.AddTable('Table ID');
8 oReport.AddColumn('Table ID', 'line column', 'Line');
9 oReport.AddColumn('Table ID', 'name column', 'Name');
10 oReport.AddColumn('Table ID', 'parallel column', 'Number of\nparallel Lines');
11 oReport.AddColumn('Table ID', 'length column', 'Length\n[m]');
12 oReport.AddColumn('Table ID', 'out column', 'Out of\nService');
13
14 aLines = GetCalcRelevantObjects('*.*.ElmLne');
15 for(oLine = aLines.First(); oLine; oLine = aLines.Next()) {
16     sRowId = oLine.GetFullName();
17     oReport.addRow('Table ID', sRowId);
18     oReport.setCellValueToObject('Table ID', 'line column', sRowId, oLine);
19     oReport.setCellValueToString('Table ID', 'name column', sRowId, oLine:loc_name);
20     oReport.setCellValueToInt('Table ID', 'parallel column', sRowId, oLine:nlnum);
21     oReport.setCellValueToDouble('Table ID', 'length column', sRowId, oLine:dline);
22     oReport.setCellValueToCheckbox('Table ID', 'out column', sRowId, oLine:outserv);
23 }
```

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:**

```

1 object   oReport;
2 oReport = this.GetParent();
```

```

3 oReport.AddListFilter('UFilt', 'Usage');
4 oReport.AddListFilterEntries('UFilt', 'Busbar');
5 oReport.AddListFilterEntries('UFilt', 'Junction');
6 oReport.AddListFilterEntries('UFilt', 'Internal');
7 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:**

```

1 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:**

```

1 object oReport,
2 oReport = this.GetParent();
3 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:**

```

1 object oReport,
2 string sCurrentValue,
3 int iNumber, iRes;
4
5 oReport = this.GetParent();
6 iNumber = 0;
7 iRes = sscanf(sNfilt, '%d', iNumber);
8 sCurrentValue = sprintf('%d', iNumber);
9 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:**

```

1 object oReport,
2 string sCurrentValue,
3 int iNumber, iRes;
4
5 oReport = this.GetParent();
6 iNumber = 0;

```

```

7 iRes = sscanf(sOldValue, '%d', iNumber);
8 iRes = sscanf(sNFilt, '%d', iNumber);
9 sCurrentValue = sprintf('%d', iNumber);
10 oReport.AddInvisibleFilter('OldValue', sCurrentValue, NULL);
11 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:

```
1 set aEditSet;
```

Add the following code to the *Create* script before line 23.

#### Example 2 “Lines report” *Create* script extension:

```

1 aEditSet.Clear();
2 aEditSet.Add(oLine);
3 oReport.SetCellEdit('Table ID', 'name column', sRowId, aEditSet);
4 oReport.SetCellEdit('Table ID', 'parallel column', sRowId, aEditSet);
5 oReport.SetCellEdit('Table ID', 'length column', sRowId, aEditSet);
6 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:**

```

1 object oLine;
2 int iColumnCmp;
3 oLine = oEditObject1;
4 iColumnCmp = strcmp(sColumnId, 'name column');
5 if (iColumnCmp == 0){
6     oLine:loc_name = sNewValue;
7 }
8 iColumnCmp = strcmp(sColumnId, 'parallel column');
9 if (iColumnCmp == 0){
10    oLine:nlnum = iNewValue;
11 }
12 iColumnCmp = strcmp(sColumnId, 'length column');
13 if (iColumnCmp == 0){
14    oLine:dline = dNewValue;
15 }
16 iColumnCmp = strcmp(sColumnId, 'out column');
17 if (iColumnCmp == 0){
18    oLine:outserv = iNewValue;
19 }

```

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 table 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 chapter [23.2.2](#) for a description of the *ComTableReport* attributes. In addition, the report must be stored in the report folder *Library\TableReport* 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 [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 Table 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 dialogue (see Figure 23.3.1) here is also where a preferred Python version can be selected.

The *PowerFactory* architecture (32- or 64-bit) determines the Python interpreter as shown below:

- *PowerFactory* 32-bit requires a Python interpreter for 32-bit
- *PowerFactory* 64-bit requires a Python interpreter for 64-bit

To check which *PowerFactory* architecture is installed, press **Alt-H** to open the Help menu and select *About PowerFactory*. If the name of *PowerFactory* includes "(x86)", then a 32-bit version is installed; if the name of *PowerFactory* instead includes "(x64)", then a 64-bit version is installed. 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.

### 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:

```
1 import powerfactory
```

To gain access to the *PowerFactory* environment the following command must be added:

```
1 app = powerfactory.GetApplication()
```

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:

```
1 user = app.GetCurrentUser()
2 project = app.GetActiveProject()
3 script = app.GetCurrentScript()
4 objects = app.GetCalcRelevantObjects()
5 lines = app.GetCalcRelevantObjects("*.ElmLine")
6 sel = app.GetDiagramSelection()
7 sel = app.GetBrowserSelection()
8 project = app.CreateProject("MyProject", "MyGrid")
9 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:

```
1 project = app.GetActiveProject()
2 projectName = project.loc_name
3 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.

```
1 lines = app.GetCalcRelevantObjects("*.ElmLine")
2 line = lines[0]
3 currLoading = line.GetAttribute("c:loading")
```

For printing to the *PowerFactory* output window, the following application object (e.g. "app" object) methods are provided:

```
1 app.PrintPlain("Hello world!")
2 app.PrintInfo("An info!")
3 app.PrintWarn("A warning!")
4 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):

```
1 project = app.GetActiveProject()
2 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:

```

1 project = app.GetActiveProject()
2 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.

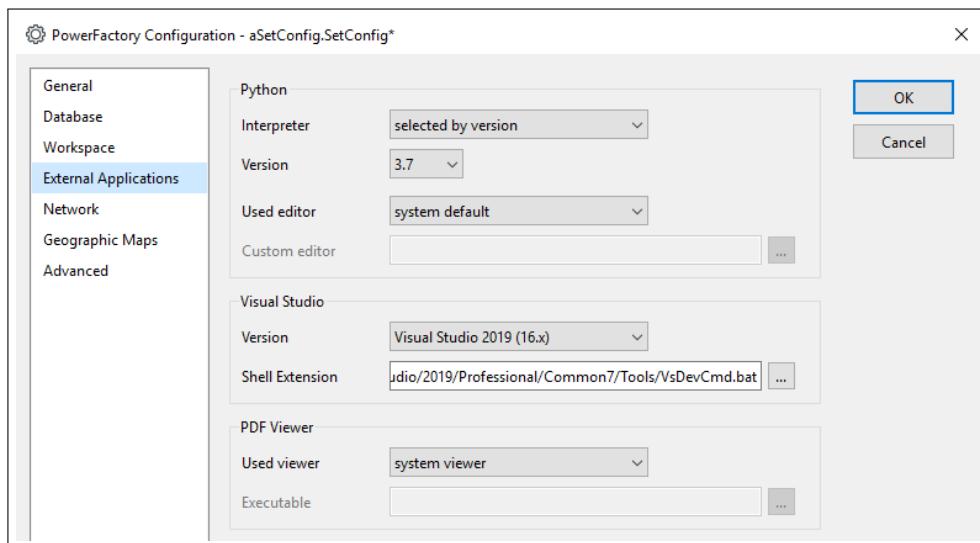


Figure 23.3.1: Selection of a preferred Python editor program

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:

```
1 script=app.GetCurrentScript()    #to call the current script
2 extObj=script.NameOfExternObj #to call the external object
3 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] :

```
1 list(DataObject, ...) SetDesktop::GetPage(str,[int])
```

Version 2- “new interface version”. Methods like the method GetPage() returns just the result, without input parameters.

```
1 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* pop-up 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

```
1 script=app.GetCurrentScript()
2 ContainedObejct=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 select *DPL Command and more*. From the *Element* drop-down list select 'Python Script

(*ComPython*’). 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-sensitive 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-sensitive menu.
  - Right-click in a blank area and select *Execute Script* from the context-sensitive 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-sensitive 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**.

### 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:

```

1 # Add powerfactory.pyd path to python path.
2 import sys
3 sys.path.append("C:\\Program Files\\DIgSILENT\\PowerFactory 2020\\Python\\3.7")
4
5 #import PowerFactory module
6 import powerfactory
7
8 #start PowerFactory in non-interactive mode
9 app = powerfactory.GetApplication()
10
11 #run Python code below
12 .....
```

The *PowerFactory* environment can be accessed directly from the Python shell as shown in Figure 23.3.2

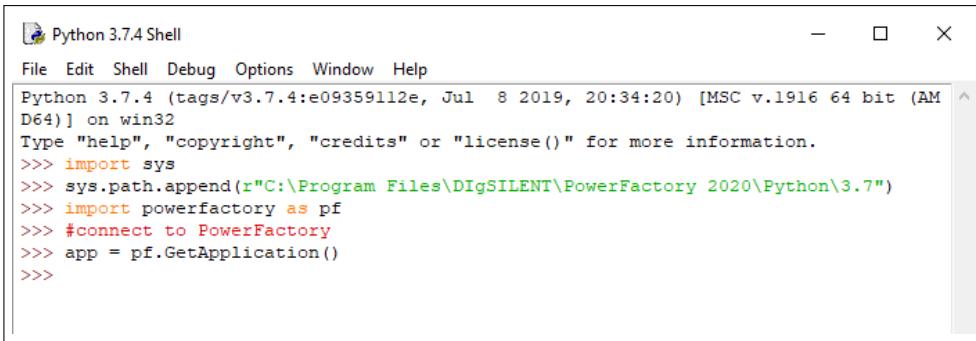


Figure 23.3.2: Python shell

**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:

```

1 #prepare debug
2 import sys
3 sys.path.append ("C:\\Program Files\\eclipse\\plugins\\
4                 org.python.pydev_2.8.2.2013090511\\pysrc")
5 import pydevd
6
7 #start debug
8 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*.

```

1 if __name__ == "__main__":
2     #connect to PowerFactory
3     import powerfactory as pf
4     app = pf.GetApplication()
5     if app is None:
6         raise Exception("getting PowerFactory application failed")
7
8     #print to PowerFactory output window
9     app.PrintInfo("Python Script started..")
10
11    #get active project
12    prj = app.GetActiveProject()
13    if prj is None:
14        raise Exception("No project activated. Python Script stopped.")
15
16    #retrieve load-flow object
17    ldf = app.GetFromStudyCase("ComLdf")
18
19    #force balanced load flow
20    ldf.iopt_net = 0
21
22    #execute load flow
23    ldf.Execute()
24
25    #collect all relevant terminals
26    app.PrintInfo("Collecting all calculation relevant terminals..")
27    terminals = app.GetCalcRelevantObjects("*.ElmTerm")
28    if not terminals:
29        raise Exception("No calculation relevant terminals found")
30    app.PrintPlain("Number of terminals found: %d" % len(terminals))

```

```

31
32     for terminal in terminals:
33         voltage = terminal.GetAttribute("m:u")
34         app.PrintPlain("Voltage at terminal %s is %f p.u." % (terminal , voltage))
35
36     #print to PowerFactory output window
37     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”. When the editor is opened in an additional window, 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 sensitive 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.

In order to close an editor window click on the “close button” (“X”) on the top of the right side of the window directly underneath corresponding buttons on the title bar.

## 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.

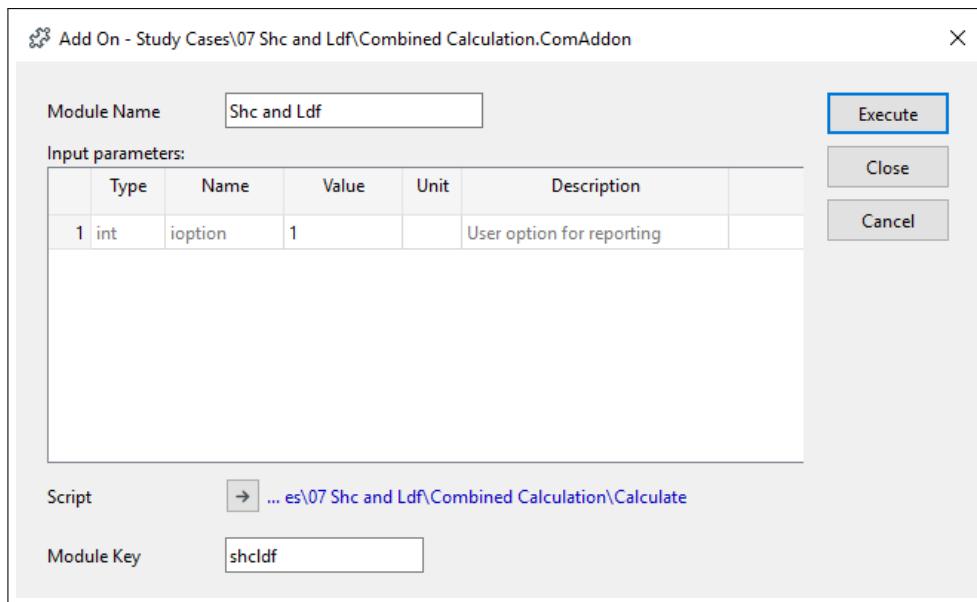


Figure 23.5.1: Add On Command object

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
- Select from the list *Others* and as *Filter* “Commands (Com\*)”. Choose from the *Element* drop down list “Add On (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 (*SetFoldflex*), 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.

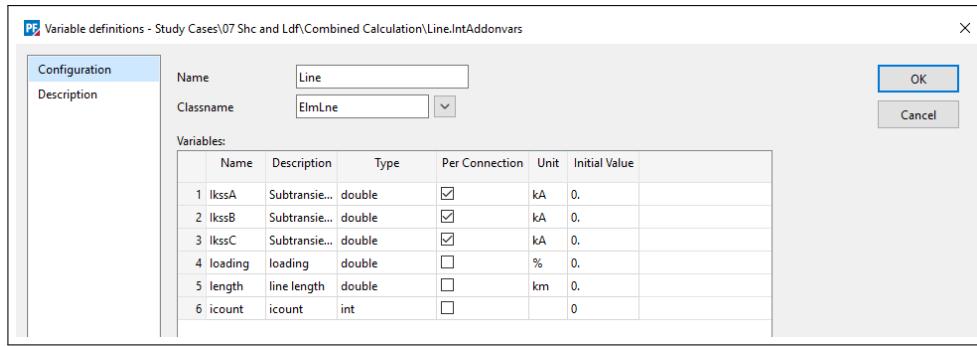


Figure 23.5.2: Example of user-defined variable definitions for line elements

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.

Name	Grid	Subtransient Short-Circuit Level MVA	Subtransient Short-Circuit Current kA	Voltage at Short Circuit p.u.
Bus 1	Nine-bus System	192,387484	6,731816	0,999365
Bus 2	Nine-bus System	223,011312	7,153091	0,973412
Bus 3	Nine-bus System	134,760262	5,637962	1,000734
Bus 4	Nine-bus System	1189,497039	2,985898	0,0318
Bus 5	Nine-bus System	883,785822	2,218496	0,023627
Bus 6	Nine-bus System	841,202581	2,111602	0,02488
Bus 7	Nine-bus System	1200,604325	3,013779	0,032097
Bus 8	Nine-bus System	960,800861	2,41182	0,025686
Bus 9	Nine-bus System	1081,917664	2,71585	0,028924

Name	Type	Object modified
ElmLne		21.06.2017 15:52:36
ElmTerm		21.06.2017 15:52:36
ElmTr2		21.06.2017 15:52:36

Figure 23.5.3: Definition of additional flexible data page and corresponding variable sets 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).

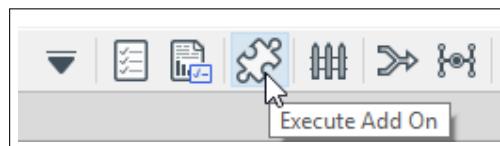


Figure 23.5.4: Running Add On modules from the Additional Tools toolbar

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 left-hand side together with the standard *PowerFactory* calculation functions.

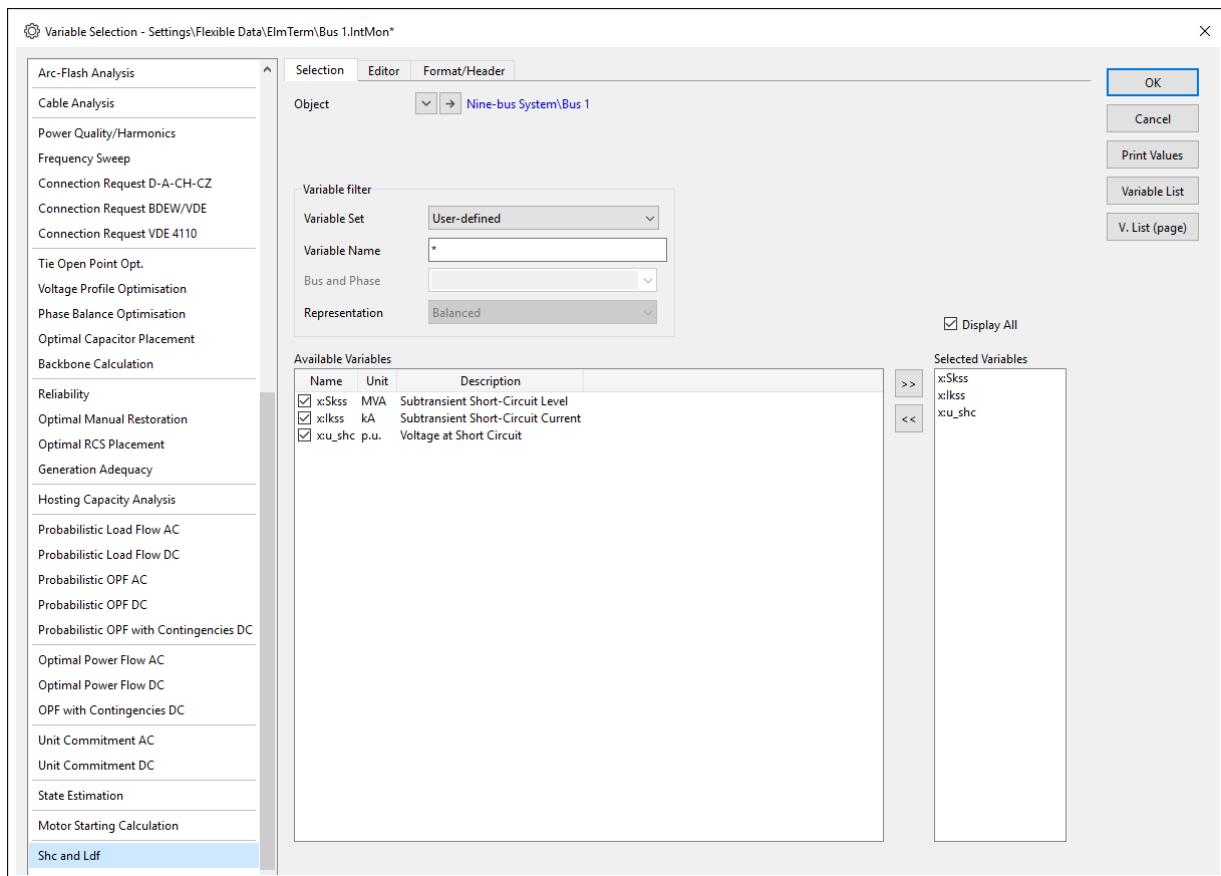


Figure 23.5.5: Selecting user-defined variables for a flexible data page

#### 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.

# 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)
  - ELEKTRA
  - MATLAB (The MathWorks, Inc trademark)
  - INTEGRAL
  - PSS/SINCAL (Siemens/PTI 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.

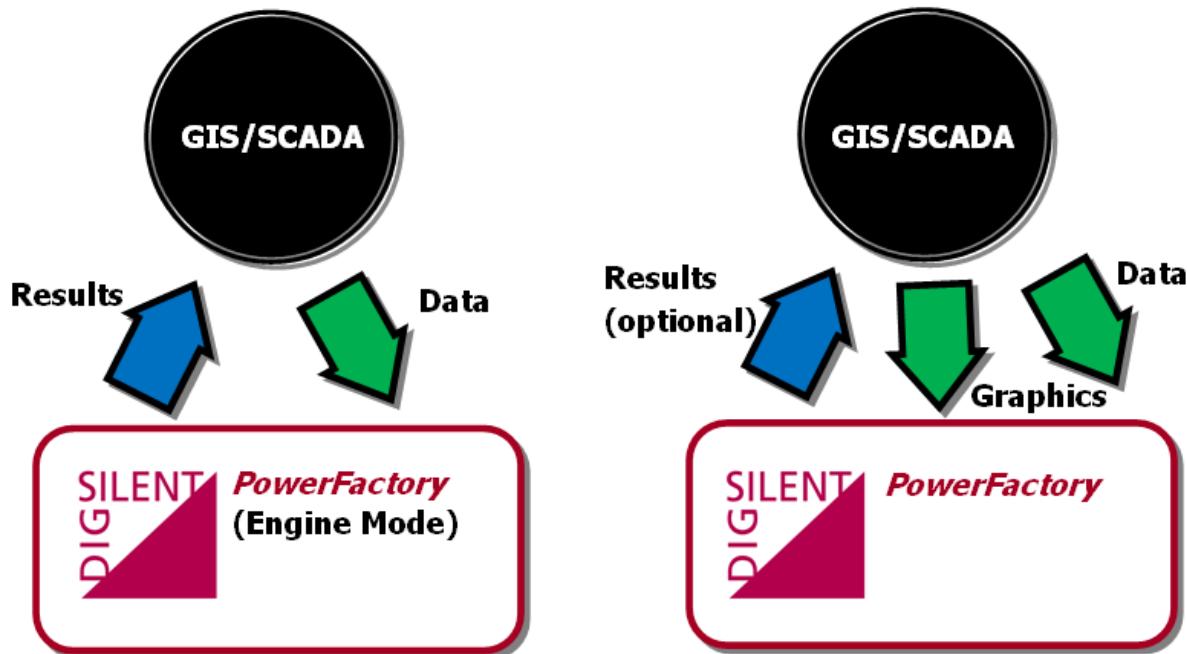


Figure 24.2.1: DGS - GIS/SCADA Integration

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.

The following sections describe each of these options.

### 24.2.3.1 General Page

**Import into New Project** By choosing this option, a project will be created where all the DGS data will be stored. The user will have the option of specifying a specific name and location (other than the default).

**Import into Existing Project** By choosing this option, the DGS data will be imported into an already existing project. Here, the data can be selective and it's not required that the imported data must be complete. In some cases, most of the objects already exist and only an update is required for some of them.

**Import from** The source of the data to be imported is specified with this option. If a *File Format* source is selected then the location and type of data (DGS, XML, MDB or XLS) must be specified. If a *Database* source is selected, then a service, User and Password information is required (the SQL server option will require an extra Database information).

### 24.2.3.2 Options Page

The visible options depend on the DGS version being used, and on the users choice of the *Import Format*.

#### Options for DGS version 4.x

##### **Predefined Library**

A predefined library located somewhere else in the database can be selected. The option of copying the library into the project is available.

##### **Create Switch inside Cubicle**

In cases where the source data has no switches defined inside the cubicles, the enabling of this option will create the switches automatically during the import. If switches already exist in a certain cubicle, the creation of switches in that particular cubicle is ignored.

##### **Replace non-printable characters**

If the source data contains not allowed characters (~, ?, etc.), they are replaced by an underscore character.

#### Options for DGS version 5.x

##### **Open single line diagram(s)**

If the DGS source contains graphics objects for single line diagrams, these will be opened automatically after import.

##### **Dataset Import (only available if a Database Schema is selected as Import Format)**

For DGS version 5 or higher, a labelled version of the data in the source data base can be selected for import. The labelled versions are mainly used to chose a time-dependent state of the data. The label name is input in the field *Label*.

#### Options for DGS version 6.x

The options for DGS version 5.x are available for DGS version 6.x as well. In addition, the options described below can be configured.

##### **Global type library**

The DGS import in earlier versions only allows for references to existing objects within the active project. With the DGS version 6.x, a global type library can be selected. Elements in the DGS source can refer to the *foreign key* of types in the selected global library.

##### **Partial Import (only available if a Database Schema is selected as Import Format)**

For DGS version 6 or higher, labelled regions can be selected for import in the source data base.

The labelled regions can be used to select specific voltage levels or sub-grids from the bulk data set. The label names are input in the field *Labels* separated by commas.

The option pages for all DGS versions contain the field

#### **Additional Parameters**

This field is specified for internal use only. No extra information is required by the user.

More detailed information on DGS and examples can be accessed by selecting from the main menu *Help → Additional Packages → DGS Data Exchange Format*

### **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
- 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.

The following sections describe each of these options.

#### **24.2.4.1 General Page**

##### **DGS Version**

Version of the DGS structure.

##### **Format**

Output format. Either as ASCII, XML, MS Excel or MS Access file (for Excel or Access, Microsoft Office must be installed on the computer) or as Oracle, MS SQL Server, ODBC DSN or generic ODBC databases (respective data base drivers must be installed.).

##### **File Name or Data Base Service**

Depending on the *Output Format*, a file name for the output file or the data base service and user access information are required.

##### **Insert Description of Variables**

If checked, a description of the column headers is included in the output file (only available for ASCII, XML and MS Excel).

### Variable Sets

Select the variable set definition for export. The data exported will be according to the variable definition specified (see the explanation at the beginning of the section). It is required to select a folder that contains the monitor variable objects (*IntMon*) related to each class that is to be exported.

#### 24.2.4.2 Options Page

The visible options mostly depend on the DGS version 4.x, 5.x or 6.x and on the users choice of the *Export Format*.

##### DPL Script

Independent of the DGS version, the user can select a DPL script. This DPL script is automatically executed before DGS export.

#### Options for DGS version 5.x

##### ***Allow hierarchy and references of exported objects to be incomplete***

If this option is set, error messages because of references to external objects (e.g. types in the global library) that will not be exported are omitted. Furthermore, the export of data classes is possible although their parent-folder class is not contained in the *variable definition set*.

##### ***Allow user-defined table names***

With this option, a prefix and suffix can be added to all table names on DGS export. The prefix or suffix is defined in the field *Additional Parameters*.

##### ***Export as dataset (available for export to data base formats only)***

This option is used to write the exported data to a specifically labelled data set (session state) into the data base. A label identifier must be given. Optional, a description can be added.

#### Options for DGS version 6.x

All options for DGS version 5.x are available for DGS version 6.x. In addition, the following options exist:

##### ***Export as Update***

DGS 6.0 supports an explicit marker *OP* to mark a data record to be created (C), updated (U) or deleted (D) on DGS import. On export, the user can chose to mark all data as *Update* by means of this option. Otherwise the exported data are marked for *Create*.

##### ***Categorisation of data for partial import (available for export to data base formats only)***

This option is used to define for each grid (ElmNet) in the active project a *data part* in the data base. In addition, certain data types and elements are labelled with respect to their meaning in the context of this partial export: global types, local types, boundary set (branches connecting elements that belong to different grids), other elements (e.g. characteristics). The names of these labels are input in the respective fields (all fields should be filled in). In addition, a global library folder can be selected to include referenced types from this library on DGS export.

#### Options for DGS version 4.x

##### ***Export Grid Name***

If this option is set, a column "Grid" is added to all tables containing network elements.

##### ***Export Cubicles***

With this option, cubicles are exported. Cubicles describe the connectivity of nodes and branches. The export of cubicles can be omitted if the grid topology is not needed (e.g. for result export).

##### ***Export Graphical Data***

The user can select one of three options for the export of graphical data:

- No** No graphical data are exported.

**Yes, with Graphic (*IntGrfnet*) Names** Graphical data are exported. All graphic object tables contain a column for the name of the graphic scheme (*IntGrfnet*).

**Yes, without Graphic (*IntGrfnet*) Names** Graphical data are exported. The graphic scheme is not referenced by the exported graphic objects.

The option page contains independent of the DGS version always the field

#### **Additional Parameters**

This field is specified for internal use only. No extra information is required by the user.

More detailed information on Variable Sets definitions (*IntMon*) can be accessed by selecting from the main menu *Help* → *Additional Packages*→ *DGS Data Exchange Format*.

## **24.3 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 23 to 34 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 is supported up to and including version 33.

### **24.3.1 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 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.

PSS/E files from versions v35.x cannot be converted/imported to *PowerFactory* at present. They first have to be converted to the PSS/E version v34 in PSS/E.

### **24.3.2 Importing PSS/E Steady-State Data**

*PowerFactory* is able to convert both steady-state data (for load-flow and short-circuit analysis) and dynamic data files. It is good practise to first import the steady-state data (described in this section), then to add the dynamic models (described in Section [24.3.3: Import of PSS/E file \(Dynamic Data\)](#)).

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*. By doing so, the *Convert PSS/E Files* command dialog will be displayed, asking the user to specify various options.

#### 24.3.2.1 General Page

**Nominal Frequency** Nominal frequency of the file to be Converted/Imported.

##### PSS/E File Type

**PSS/E Raw data** Location on the hard disk of the PSS/E raw data file. By default the program searches for \*.raw extensions.

**Sequence Data** Location of the PSS/E sequence data file. By default the program searches for \*.seq extensions.

---

**Note:** From *PowerFactory* 2020 onwards the drw file is no longer converted/imported. Instead, network graphics can be generated after import into *PowerFactory*, using the *Diagram Layout Tool*.

---

##### Save converted data in

**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 will be stored.

The following topics:

- **Dyn. Models Data,**
- **Composite Frame Path,**
- **DSL - Model Path,**
- **Parameter Mapping**

are not used for the import of steady-state data and will be explained in the dynamic import Section [24.3.3](#).

#### 24.3.2.2 Options Page

- **Convert only sequence data file** - With this option enabled, the converter will only add the sequence data to an existing project.
- **Convert only dynamic models file** - With this option enabled, the converter will only add the dynamic data file to an existing project (only for dynamic data import).
- **Only convert file (no DB action)** - Internal option used for syntax check and error messages during conversion. Normally this box should be left unchecked.
- **Output only used dynamic models** - Displays a list of used dynamic models (only for dynamic data import).
- **Unit of 'LEN' for lines in miles instead of km** - With this option enabled, all lengths will be interpreted in miles in the PSS/E raw files.
- **Consider transformer phase shift** - With this option enabled, transformer phase shifts will be considered. This option is recommended and activated by default.
- **Convert Induction Machines (Generators: P<0)** - With this option enabled, all generators in the raw data file that have negative active power will be converted to asynchronous machines. For transmission grids the option should be disabled for proper modelling of phase shift generators.

- **Automatic 3-W. Transformer detection/conversion** - In versions <27, PSS/E does not handle 3-winding transformers as a dedicated model. In such cases, the 3-winding transformer is modelled with three 2-winding transformers connected to a busbar. If this option is selected, the converter will try to detect the existence of three 2-Winding Transformers connected to a busbar. If any candidates are available, *PowerFactory* will replace them by a 3-Winding Transformer. The detection algorithm uses the impedances and the voltage control of the transformers as reference. From version 27 onwards PSS/E supports the 3W-transformer model, so that *PowerFactory* does not start an automatic detection of 3W-Trf modelled as 2W-Trfs.
- **Convert capacitive line shunts to line susceptance B'** - If a line has line shunts the converter adds automatically the line shunt capacitance to the C1' (B1') in the *PowerFactory* line type.
- **Convert Common Impedance as Transformer** - If this option is selected, the Common Impedance in PSS/E may be converted to a *PowerFactory* common impedance or to a transformer.
- **Convert Series Capacitance as Common Impedance** - Older versions of PSS/E do not handle series capacitances as a dedicated model. These elements therefore are represented by lines with negative reactances. During the conversion, *PowerFactory* detects these branches and converts them to series capacitances (by default) or to common impedances (when this option is active).
- **Convert off-nominal turn ratio to transformer tap** - Transformer ratios different from the rated ratio are automatically converted to a transformer type using taps, including the correct tap position.
- **Busbar naming: 'PSSE\_NAME'** - With this option enabled, the busbars are named similar to the PSS/E raw data file (without bus number).
- **Branch naming: 'BUSNAME1\_BUSNAME2\_ID'** - With this option enabled, the branches are named as the name of the busbars + ID.

**Additional Parameters** - This field is specified for internal use only. No extra information is required by the user.

### 24.3.3 Import of PSS/E file (Dynamic Data)

As explained in Section 24.3.2 it is good practice first to import the steady-state data and then to add the dynamic model data.

Some dynamic models used in PSS/E are available in the Global Library. User defined dynamic models should be modelled in *PowerFactory* before importing the program. In this case, an important condition for successful file conversion is that all DSL models used during the conversion process should be stored in the same model library folder.

If the original library should use specific folders for the different types of controllers (AVR,PCO,PSS, etc.), the user should copy all of the models into the same library folder, in this case the recommendation is to copy the dynamic models from the global library into the library where the rest of the user defined models are located. After the conversion, the user may re-arrange the models.

The procedure to start the import of dynamic network data is very similar to the import of steady-state data. Some parameter adjustments have to be made.

#### 24.3.3.1 General Page - Dynamic Models

On the *General* page of the import dialog the following topics have to be specified:

**Dyn. Models Data** - Location of the PSS/E Dynamic Models data file. By default the program searches for \*.dyn and \*.dyr extensions.

**Use Standard Models from global library** - If this option is enabled, *PowerFactory* will automatically point to the Standard Models library located in the Global library. There will be no need of selecting the composite Frame Path and DSL Model Path.

**Composite Frame Path** - Location in the *PowerFactory* data base where the composite frames are stored (Standard Models/Composite Models Frames...).

**DSL - Model Path** - Location in the *PowerFactory* data base where the DSL models are stored (Standard Models...).

**Parameter Mapping** - Location of the *PowerFactory* mapping file. This is an option that normally will not have to be defined by the user. By default *PowerFactory* will automatically set up its own internal mapping file. This file defines how to translate the PSS/E internal models into *PowerFactory* models, including the mapping of controller parameters. For automated conversion of user-defined PSS/E controllers the mapping file may be customised.

#### 24.3.3.2 Import Options Page - Dynamic Model Import

On the *Options* page of the import dialog the following options should be considered:

**Convert only dynamic models file** - With this option enabled, the converter will only add the dynamic data file to an existing project.

**Output only used dynamic models** - Displays a list of used dynamic models.

#### 24.3.4 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 standard IEEE models. 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.

##### 24.3.4.1 Export General Page

**RAW Conversion 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.

**SEQ Conversion 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.

**DYN Conversion File** - Path and file name for the PSS/E DYN/DYR file, containing the dynamic models of the project. By default the program selects the \*.dyr extension.

**PSS/E Version** - Version of the exported PSS/E file (25 to 33).

##### 24.3.4.2 Export Options Page

**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:

- **Grids:** the exported file will have the areas defined according to the Grids defined in the *Power-Factory* model.
- **Areas:** the exported file will have the areas defined according to the Areas defined in the *Power-Factory* model.

**Additional Parameters** - This field is specified for internal use only. No extra information is required by the user.

## 24.4 ELEKTRA Interface

*PowerFactory* offers the user the possibility to import different types of ELEKTRA files. The files supported for import are as follows:

- **Elektra network models**
  - Element data (\*.esd) from Elektra Version 3.60 to 3.98, which contain the topological and electrical data of the elements in the grid.
  - Network diagrams (\*.enp) from Elektra Version 3.92 to 3.98, which contain the graphical representation of grids.
- **Elektra equipment type library**
  - Type data (\*.dat), which contains equipment types.

### 24.4.1 Import of Elektra Data

The general way to import data via the Elektra interface is as follows:

- From the main menu, select: *File* → *Import* → *Elektra....* The Elektra-Import dialog will be displayed.
- Select the desired options and click on the **Execute** button.

---

**Note:** The Elektra import cannot be executed if Elektra is open. Close the software before executing the import.

---

The import will be executed regardless of whether a project is activated or not. At the end of the import, the project will be activated. If there is another project activated while importing the Elektra data, *PowerFactory* will deactivate the active project, and activate the newly-created or selected project (according to the settings).

The options available in the Elektra import dialog are described in the following section.

## 24.4.2 General Settings

### Import into

**New project** A new project will be created in which all of the Elektra data will be stored. The user can select a name and a storage location. Different versions of the same network model should be stored in new projects.

**Existing project** Elektra data will be imported into an existing project. Use this option if grids from different regions will be connected and should be calculated together in one project.

### Files

**Kind of data** Within the Elektra import, *Element/graphic data* (data type \*.esd and \*.enp) or *Type data* (data type \*.dat) can be imported, according to the selection.

**Element data** If *Element/graphic data* is selected, set the storage location of the Elektra element data by clicking the “...” icon.

**Graphical data** Add graphical data for the element. Select *Delete* to remove the data from the list.

**Type data** If *Kind of Data: Type data* is selected, click on *Add* to select the Elektra type library (\*.dat) for import. Repeat this step if more type libraries should be added to the import. Select *Delete* to delete single files from the selection.

## 24.4.3 Advanced Settings

On the *Advanced* settings page, the following options can be used to simplify the imported network. In addition, there are two options to activate the import of coupling impedances and active/reactive power characteristics (Q(P) curves).

### General Options

**consider graphical node representation** If a node is set to *Internal Node* in the Elektra element data, *PowerFactory* will also set the node to *Internal Node*. That is, the *usage* of the node in *PowerFactory* is set according to the *usage* in Elektra element data.

**create detailed busbar systems for single busbars** By default, a detailed representation of substations is generated for all Elektra busbars in a *PowerFactory* substation. This is done regardless of whether it is a single or double busbar. This option should be chosen to set locations where only single busbars exist, to single busbars in *PowerFactory*.

**create auxiliary graphic objects in annotation layer** Objects in the Elektra open graphic (open texts, memos, rectangles, pictures, ...) will be transformed into the *annotation layer* of *PowerFactory* by default. These layers can be scaled and changed in *PowerFactory*. As an alternative, graphical objects can be split into parts in the import process. This leads to limited options in later adaptations of the objects.

**create element names with reference to the node name** In *PowerFactory*, every element must have a unique name. To ensure this uniqueness for the Elektra import, the names are comprised of the following parts: *Elektra element name - Elektra name of terminal 1 - Elektra name of other terminal*. If this name has more than 40 letters it will be shortened.

**coupling impedances** Coupling impedances between adjacent overhead lines in Elektra network data are converted into corresponding tower elements (*ElmTow*) and tower types *TypTow* in *PowerFactory*.

**convert Q(P) curves** The reactive power behaviour of generator units or synchronous machines in Elektra data can be given as an active/reactive power characteristic. These curves are converted into a Q(P) characteristic in *PowerFactory* and assigned to the corresponding static generator/s or synchronous machine/s.

#### Individual scaling factors at Elektra node elements

Active and reactive power can be modified through scaling factors in Elektra on different layers. These factors are transformed into scalar *PowerFactory* characteristics, upon import of Elektra element data. If there are many individual scaling factors for Elektra node elements, one of the following options can be chosen. These options may assist in reducing the number of characteristics in *PowerFactory*.

**Ignore all scaling factors** The factors for active and reactive power for Elektra node elements are ignored within the data import. The results of the load flow calculation are influenced by this option.

**Calculate resulting power quantities** The multiplication of the active and reactive power by the Elektra node element factor is transferred into *PowerFactory*.

**Create individual scale factor objects** For all factors for Elektra node elements that are set to a value different to '1', corresponding scalar characteristics are created in *PowerFactory*. This is the default option.

**Additional Parameter** This field is for internal use. No additional information is required from the user.

### 24.4.4 Importing Elektra Network Data

To import Elektra network data, choose *Kind of data: Element/graphic data*. The following combinations of element and graphic data exist:

1. Selection of Elektra element data (\*.esd) without graphic data  
The element and topological data from the \*.esd file will be imported. Type data for the element data will be created. There is no creation of a network diagram.
2. Selection of Elektra element data (\*.esd) and one or more corresponding graphic files (\*.enp)  
The included topological and type data from the \*.esd file will be imported. Type data for the element data will be created. Additionally, a network diagram for every selected Elektra graphical data will be created and elements are linked to the graphical objects (if present in both files).
3. Selection of Elektra graphical data (\*.enp), without element data.  
If only graphical data has been selected, for each graphic file one network diagram will be created. From the topological information in the \*.enp file, network data will be created. This network data does not contain technical parameters or type references.

### 24.4.5 Importing Elektra Type Data

To import Elektra type data, select one or more \*.dat files.

In the folder *Library/Equipment Type Library* from the import project, a new Equipment Library will be created for each file and relevant kind of element.

If the successfully imported type data should be used in *PowerFactory*'s global library, continue as follows:

1. Change the user to Administrator by selecting *Tools* → *Switch User...* → *Administrator* via the main menu.
2. Open the *PowerFactory* Data Manager, and create a new folder of type *Library* within the directory *Database*.
3. Copy the Equipment Library from the import project into this folder.

#### 24.4.6 Output Window

During the import the following information is provided in the output window:

- Network elements which do not coexist in the Elektra element and in the Elektra graphical data (multiple entries while importing multiple graphical files are possible).
- Network elements which are generated from power ratings in Elektra nodes.
- Coupling objects between different locations, which cannot be converted.
- Graphical objects whose names are adapted during import.
- Inconsistent or incomplete element parameters.

### 24.5 NEPLAN Interface

*PowerFactory* offers to the user the option of importing different types of NEPLAN files. The files supported for importing are the following:

- **NEPLAN 4**
  - Project File Data (\*.mcb) containing the topological, electrical and graphical data.
  - Line Data Type (\*.ldb) containing the line type information.
- **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.

#### 24.5.1 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 left activated upon completion.

Independent of the NEPLAN file version (4 or 5), 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 4 Files

When importing NEPLAN 4 files, the user has basically two options:

1. Selection of a \*.mcb file.

If the user selects this type of file and if a corresponding \*.ldb file is present (should be in the same directory where the \*.mcb is stored), then the information of both files gets imported. If only the \*.mcb file exists, then only the information regarding this file is imported (which can also contain line data).

2. Selection of a \*.ldb.

If the user selects this type of file only the information regarding this file (line data) is imported.

## 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 directory! As a recommendation, create a separate folder and place all the files there.

The following section describes each of the NEPLAN import dialog options.

### 24.5.1.1 General Settings

#### File Type

**Neplan Data** Location on the hard disk of the NEPLAN data file. Three types of files are available:  
\*.mcb, \*.ldb and \*.ndt.

#### Save converted data in

**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 will be stored.

#### Common Conversion Settings

##### Automatic busbar system detection

**Import Graphic Information** If this option is enabled then the graphical information is imported and the single line diagram is generated. In case of NEPLAN 5 import the \*.cde file is required.

##### Graphic Import Options (only for NEPLAN 5 import)

**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

##### Additional Parameters

This field is specified for internal use only. No extra information is required by the user.

## 24.6 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.

### 24.6.1 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 '**Save converted data in**' 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.

### 24.6.2 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.

## 24.7 PSS SINCAL Interface

*PowerFactory* offers the user the option to import MS Access database files from PSS SINCAL for Load Flow and Short Circuit analysis. The following files are supported:

- \*.mdb

### 24.7.1 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 MS Access database file of the SINCAL project (usually named *database.mdb*) in the field *Database name*.
- In the *Save converted data in* field, the user can enter a project name, and the *PowerFactory* user for this project can be selected.

The PSS SINCAL data import will always create a new *PowerFactory* project.

The SINCAL \*.mdb database files contain graphical information. This information is converted into a *PowerFactory* network diagram.

---

**Note:** A SINCAL import error message appears when the *PowerFactory* installation type (32bit or 64bit) is different from the Microsoft Office installation type (32bit or 64bit). To open the SINCAL database (\*.mdb) during the import to *PowerFactory*, MS Access is required.

During the installation of Office / Access 32bit, Microsoft only installs the 32bit ODBC driver by default. *PowerFactory* (64bit) needs the 64bit ODBC driver. The same goes for the other way around: Office / Access 64bit is installed by Microsoft by default only with the 64bit ODBC driver. *PowerFactory* (32bit) needs the 32bit ODBC driver.

The user has to 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.

---

## 24.8 UCTE-DEF Interface

In *PowerFactory*, both export and import of **UCTE-DEF** (**U**nion for the **C**o-ordination of **T**ransmission of **E**lectricity - **D**ata **E**xchange **F**ormat) 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.

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>

### 24.8.1 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.

#### 24.8.1.1 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 on the hard disk 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.

## 24.8.2 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.

### 24.8.2.1 General Settings

#### File Type

**UCTE Data** Location on the hard disk where the UCTE files will be stored. Two types of files are available: \*.uct and \*.ucte.

**Grids** Selection of which grids to export.

#### 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.

**Additional Parameters** This field is specified for internal use only. No extra information is required by the user.

## 24.9 CIM Interface

In *PowerFactory*, both export and import of **CIM** (Common Information Model) is supported. The CIM interface is currently intended for importing/exporting the following profile:

- ENTSO-E 2009

(Options “ENTSO-E 2010” and “ENTSO-E 2009 Dynamic Models” in the drop-down menu relate to profiles which were never formally released. Therefore, although these options are available to the user, they are not supported.)

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 information relating to CGMES, please see section [24.10](#) below.

### 24.9.1 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.

### 24.9.2 General Page

#### 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.

#### Import from

**Profile** Currently the profile ENTSO-E 2009 is supported.

**separated Files** With this setting the user has the option to import the equipment, topology and solved state files separately.

**CIM File** Location on the hard disk of the CIM files. Two types of files are supported: \*.zip and \*.xml.

**Additional Parameters** This field is specified for internal use only. No extra information is required by the user.

### 24.9.3 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.

#### 24.9.3.1 General Page

##### Export to

**Profile** Currently the profile ENTSO-E 2009 is supported.

**separated Files** With this setting the user has the option to export the equipment, topology, and solved state files separately.

**CIM File** Location on the hard disk where the CIM files will be stored. Two types of files are supported:  
\*.zip and \*.xml.

##### Export Selection

**Grids** Selection of which grids to export.

**Border Nodes Grid** Selection of the grid which contains the X-nodes.

## 24.10 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*. 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 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*).

#### 24.10.1 CIM Data Import

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 existing project** imports the data into a new archive with the given name; it requires an 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.

### 24.10.2 CIM Data Export

The CIM Data Export is accessible via *Tools → CGMES Tools → CIM Data Export* or *right-click → CIM Data Export*. 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 both 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.

### 24.10.3 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*

To convert all data contained in an imported archive, select “Source Archives” 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 “Convert selected profiles” and specify those 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 “Additional Archives”.

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 “Additional Archives”.

For both selections “Additional Archives” as well as “Source Archives” 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.

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).

---

#### 24.10.4 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.
- **Additional Archives** is used as a reference (e.g. original imports) to ensure the persistence of RDF IDs for CIM objects not represented in the *PowerFactory* model (e.g. cim:PowerTransformerEnd).

For “Additional Archives” the user may either select a single or multiple CIM archives for example to get dependencies on the correct Boundary grid as well as keeping IDs persistent.

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).

The option **Create difference models** creates a difference model, when selected. Note that difference models can only be created if *Existing archive* is selected as the destination. This archive should contain the base models for the difference models.

In order to create a bus-branch model, the option **Create bus-branch model** can be selected. When ticked, an internal reduction from a node-breaker model to a bus-branch model is done automatically. The resulting CIM archive will be a bus-branch model according to CGMES.

##### Convert network selection

The *PowerFactory* networks to be converted can be selected by ticking the checkbox. Only activated networks can be converted. If one of the networks is to be treated as boundary grid, the corresponding checkbox must be ticked as well. Each network to be converted must have a Modelling Authority Set URI.

- 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.

##### Advanced Options

Further information for the model to be converted can be altered under the *Advanced Options* page:

- **Version** is an optional parameter to define the model version

- **Description** is an optional parameter to describe the model
- **Additional options** are inputs causing specific behaviour and do not need to be used.

## 24.10.5 CIM Data Validation

The CIM Data Validation is accessible under *Tools → CGMES Tools → CIM Data Validation* or *right-click → CIM Data Validation*.

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.

---

## 24.10.6 Import and Export of the EIC as additional parameter

### Grid to CIM Conversion

The “IdentifiedObject.energyIdentCodeEic” 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.energyIdentCodeEic” is set in the corresponding CIM object.

### CIM to Grid Conversion

The “IdentifiedObject.energyIdentCodeEic” 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.).

## 24.11 MATLAB Interface

For a detailed description on the MATLAB interface refer to Chapters Stability and EMT Simulation and Modal Analysis, Sections [30.6.3: MATLAB Interface for DSL models](#) and [32.2.3: Output Options Modal Analysis](#).

## 24.12 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 24.12.1 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.

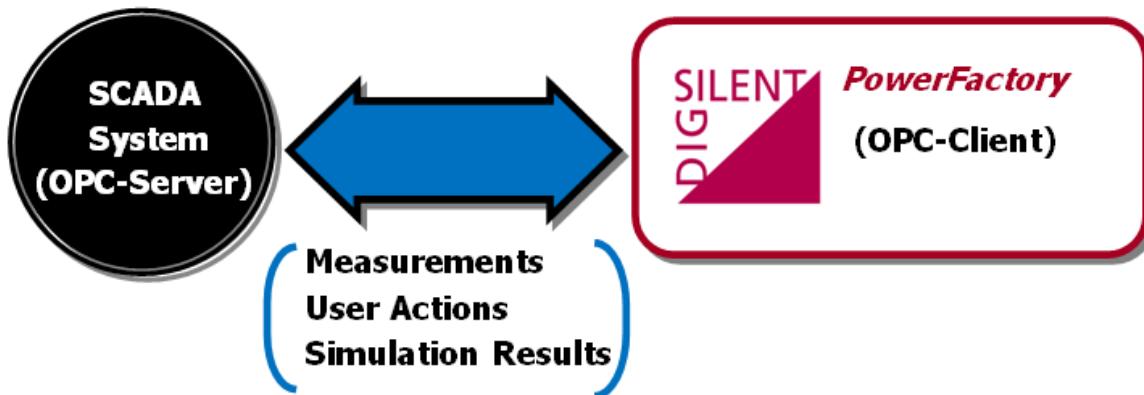


Figure 24.12.1: SCADA -*PowerFactory* integration via the OPC interface.

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.

### 24.12.1 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.

## 24.13 StationWare Interface

This chapter describes the *StationWare* interface. An introduction into *StationWare* is provided in Section [24.13.1](#).

The following two sections describe the overall *StationWare* architecture (Section [24.13.2](#)) and the conceptual differences between *PowerFactory* and *StationWare* (Section [24.13.3](#)).

Both *PowerFactory* and *StationWare* have to be configured before they can be used together (Section [24.13.4](#)).

The *Getting Started* section (Section [24.13.5](#)) provides an introduction to the most important features. The complete documentation can be found in the section 'Description of the Menu and Dialogs' (Section [24.13.6](#)).

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*.

### 24.13.1 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*.

### 24.13.2 Component Architecture

*DIGSILENT StationWare* is a so-called *Client-Server Application*: the functionality is distributed over at least two computers: client and server. Figure [24.13.1](#) gives an overview of the components.

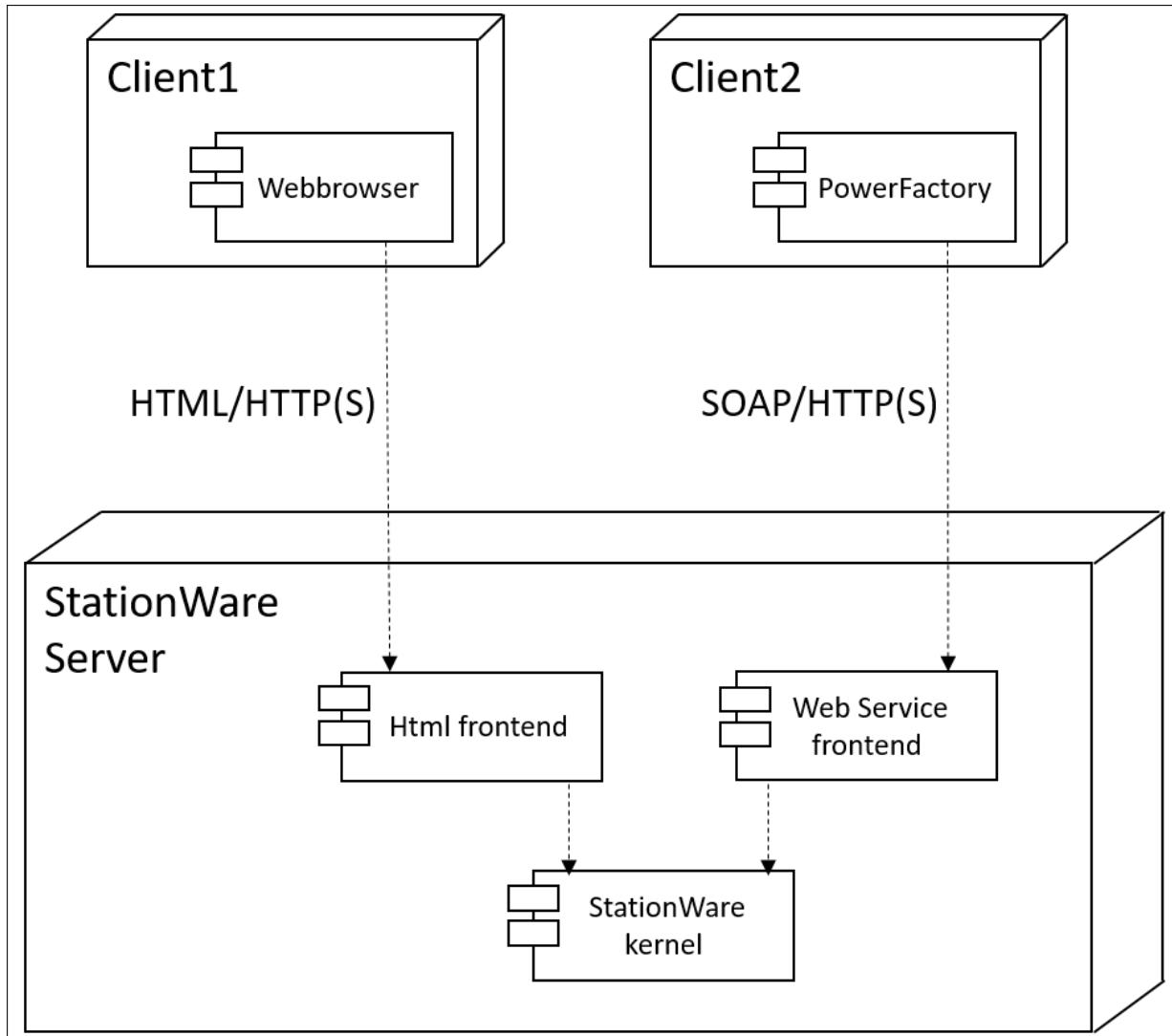


Figure 24.13.1: Architecture overview

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.

*DlgSILENT 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.

### 24.13.3 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.

#### 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 24.13.2.

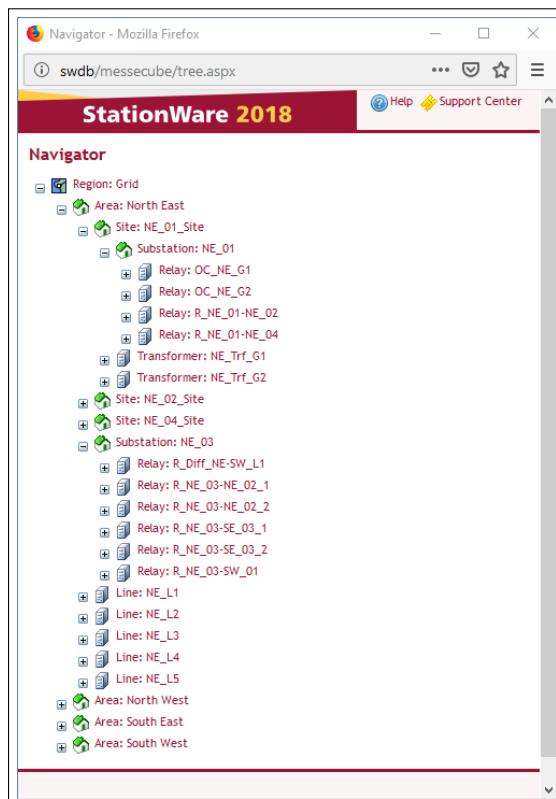


Figure 24.13.2: *StationWare* locations

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 24.13.3 for a typical *PowerFactory* project.

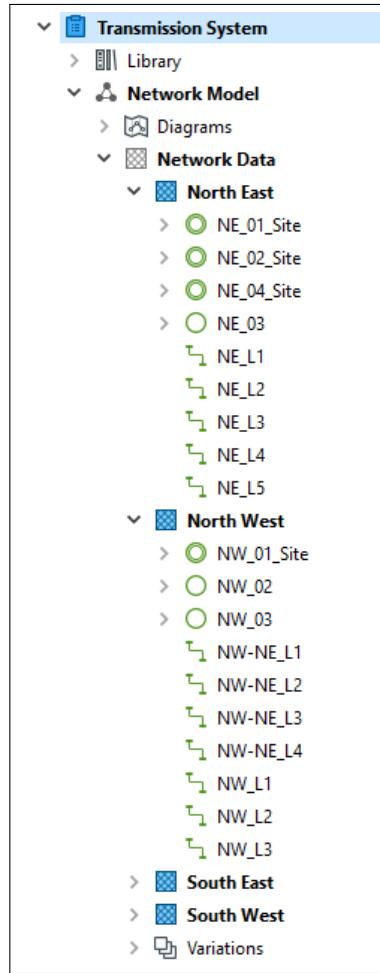


Figure 24.13.3: *PowerFactory* project

*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.

## 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 24.13.4 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.

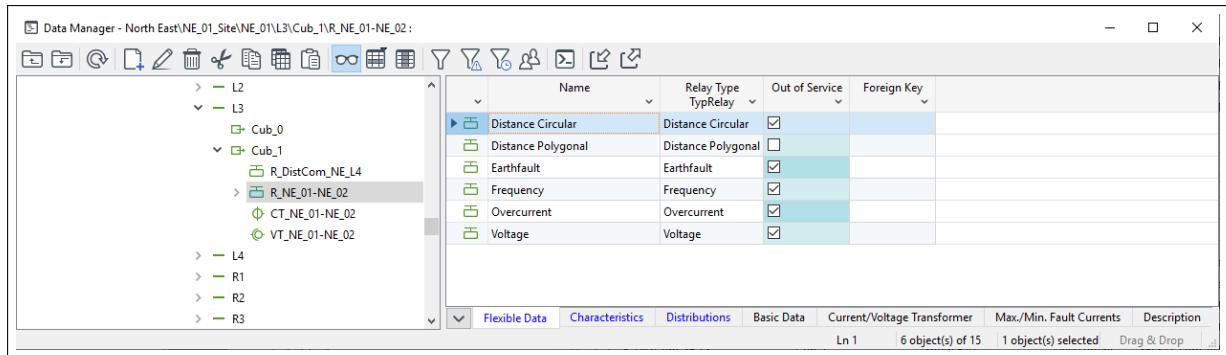


Figure 24.13.4: PowerFactory relay

### 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. A 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 24.13.1 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	I/s
	c	float in [0.03, 4.65]	1.0	
H	d	string	'DEFAULT'	
	e	enum 'yes', 'no', 'maybe'	'yes'	

Table 24.13.1: Settings Definition

According to this parameter definition a device can have settings as shown in tables 24.13.2 or 24.13.3.

Group, Name	Value
G,a	7
G,b	23.43
G,c	1.1
H,d	'abc'
H,e	'maybe'

Table 24.13.2: Settings Example 1

Group, Name	Value
G,a	8
G,b	0
G,c	1.1
H,d	'abcdef'
H,e	'yes'

Table 24.13.3: 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 24.13.4 for a definition and Table 24.13.5 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 24.13.4: Parameter Definition

Some relays support *multiple setting groups* (MSG) also called *parameter sets*. Such relays have the same group many times (c.f. table 24.13.5). 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*.

### 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 24.13.5: 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 24.13.6: 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.)

## 24.13.4 Configuration

In order to transfer data between *PowerFactory* and *StationWare* both systems must be configured.

### 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 fulfills these requirements, and there exists a setting of this phase.

### 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 *TypRelays* in the Library must of course support *StationWare - PowerFactory* mapping.

## 24.13.5 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
      |   +- PsmssExport.ComDpl
      |   +- PsmssImport.ComDpl
      |
      +- Operational Library.IntPrjfilder
  +- Scripts.IntPrjfilder
  +- StationWare.IntPrjfilder
    +- Attributes.IntFolder
      | |
      | +- ElmLne.IntFolder
      |   | +- PsmssExport.ComDpl
      |   | +- PsmssImport.ComDpl
      |   +- ElmTr2.IntFolder
      |     +- PsmssExport.ComDpl
      |     +- PsmssImport.ComDpl
      |
      +- Results.IntFolder
        | |
        | +- arcflash.IntFolder
        |   | +- ElmTerm.IntFolder
        |   |   | +- PsmssExport.ComDpl
        |   |   +- ElmLne.IntFolder
        |   |     +- PsmssExport.ComDpl
        |   |
        |   +- shc.IntFolder
        |     +- ElmTerm.IntFolder
        |       | +- PsmssExport.ComDpl
        |       +- ElmLne.IntFolder
        |         +- PsmssExport.ComDpl
        |
        +- Settings.IntFolder
          | |
          | +- ElmLne.IntFolder
          |   | +- PsmssExport.ComDpl
          |   | +- PsmssImport.ComDpl
          |   +- ElmTr2.IntFolder
          |     +- PsmssExport.ComDpl
          |     +- PsmssImport.ComDpl
  
```

Figure 24.13.5: 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](#).

### 24.13.5.1 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 [24.13.6](#)).

#### 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 [24.13.6](#).

- Go to the detail page of the *Getting Started Relay 1* (Figure [24.13.7](#)).

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.

Name	Manufacturer	Usage	Type	Category	Description	Foreign Key
Getting Started Relay 1	Siemens	7SA61_generic	Relay			
Getting Started Relay 2	Siemens	7SA61_generic	Relay			

Figure 24.13.6: Substation

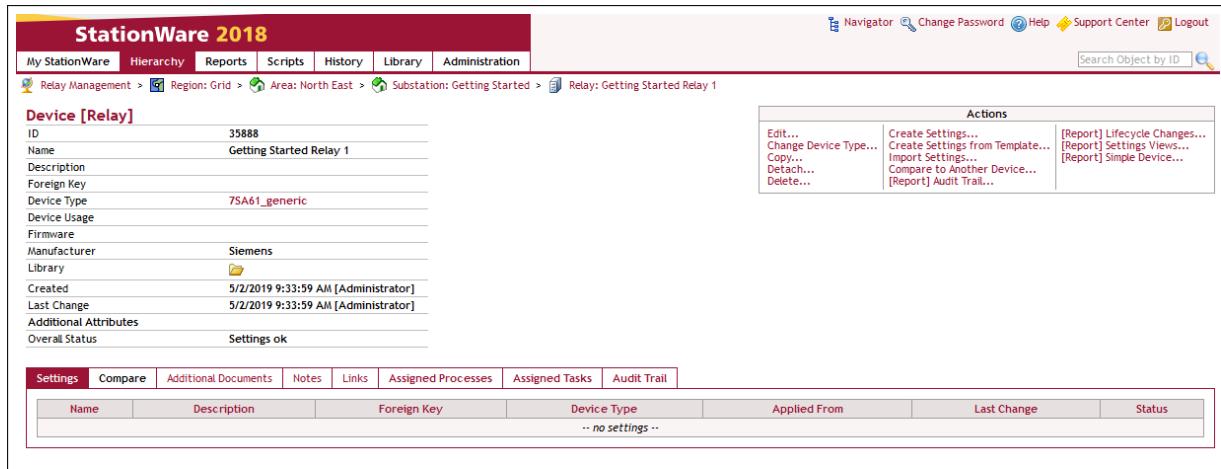


Figure 24.13.7: Device

### 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 24.13.8.

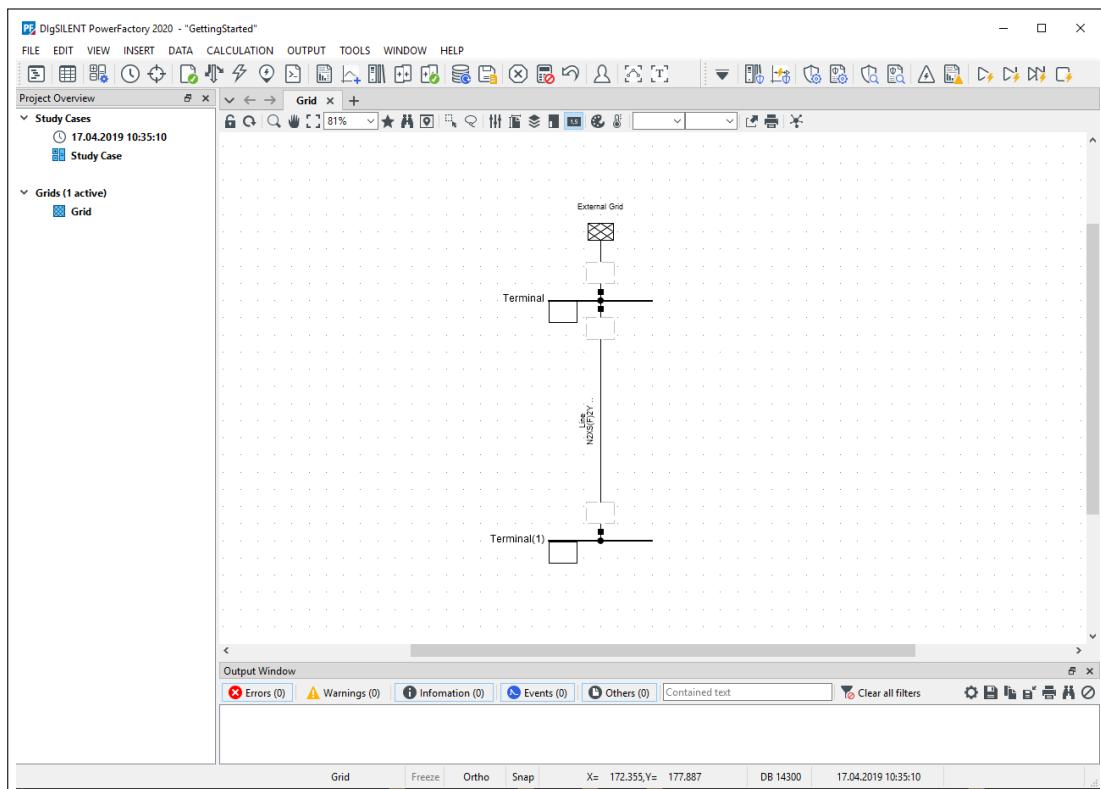


Figure 24.13.8: Grid

Now add a relay to the upper terminal:

- Right-click the cubicle quadrangle with the mouse. A context menu pops up.
- Select *New Devices.../Relay Model...* as shown in Figure 24.13.9.

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 24.13.10),
- 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.

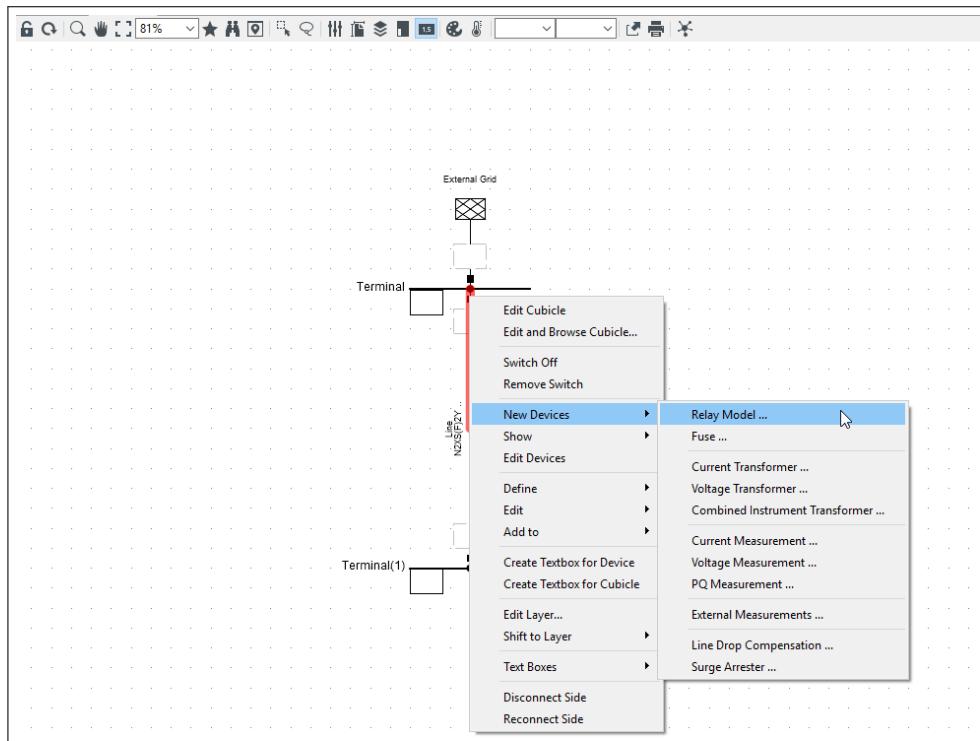


Figure 24.13.9: Cubicle context menu

- 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 24.13.11).

#### 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 24.13.12.

- 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.

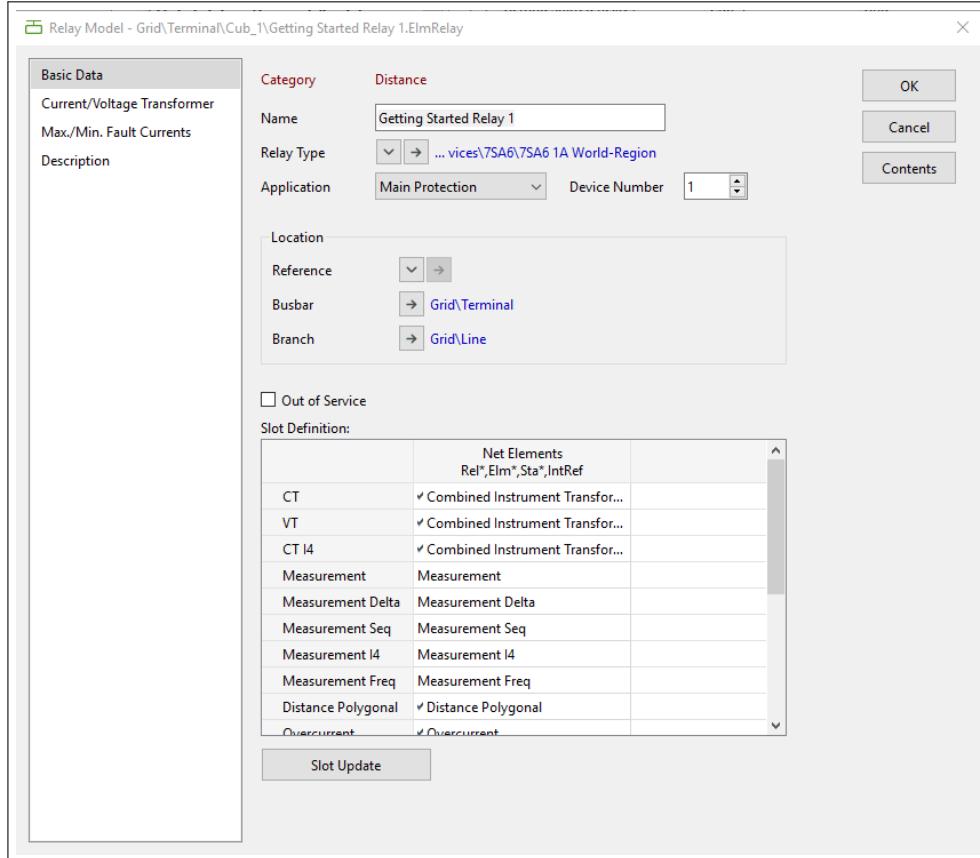


Figure 24.13.10: Relay dialog

- Enter the Server Endpoint URL of the *StationWare* server. The URL should have a format similar to  
`http(s)://192.168.1.53/pmsws/PSMSService.asmx`.
- Enter *Username* and *Password* of a valid *StationWare* user account.

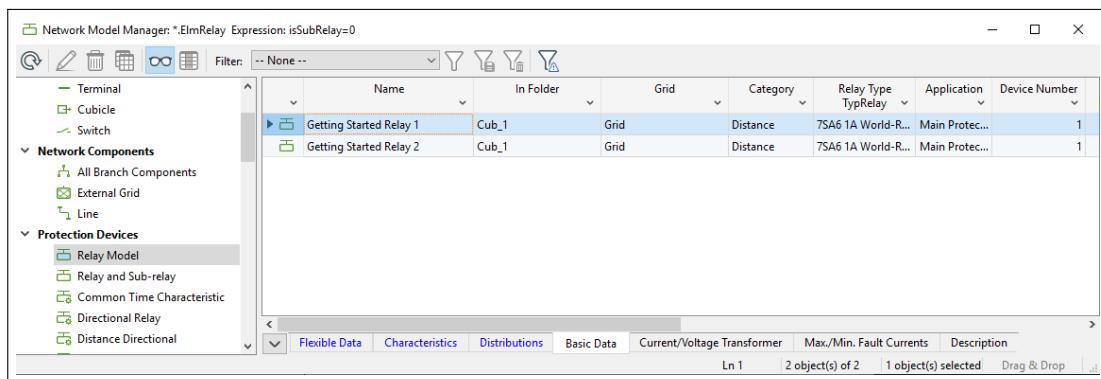


Figure 24.13.11: Relay display

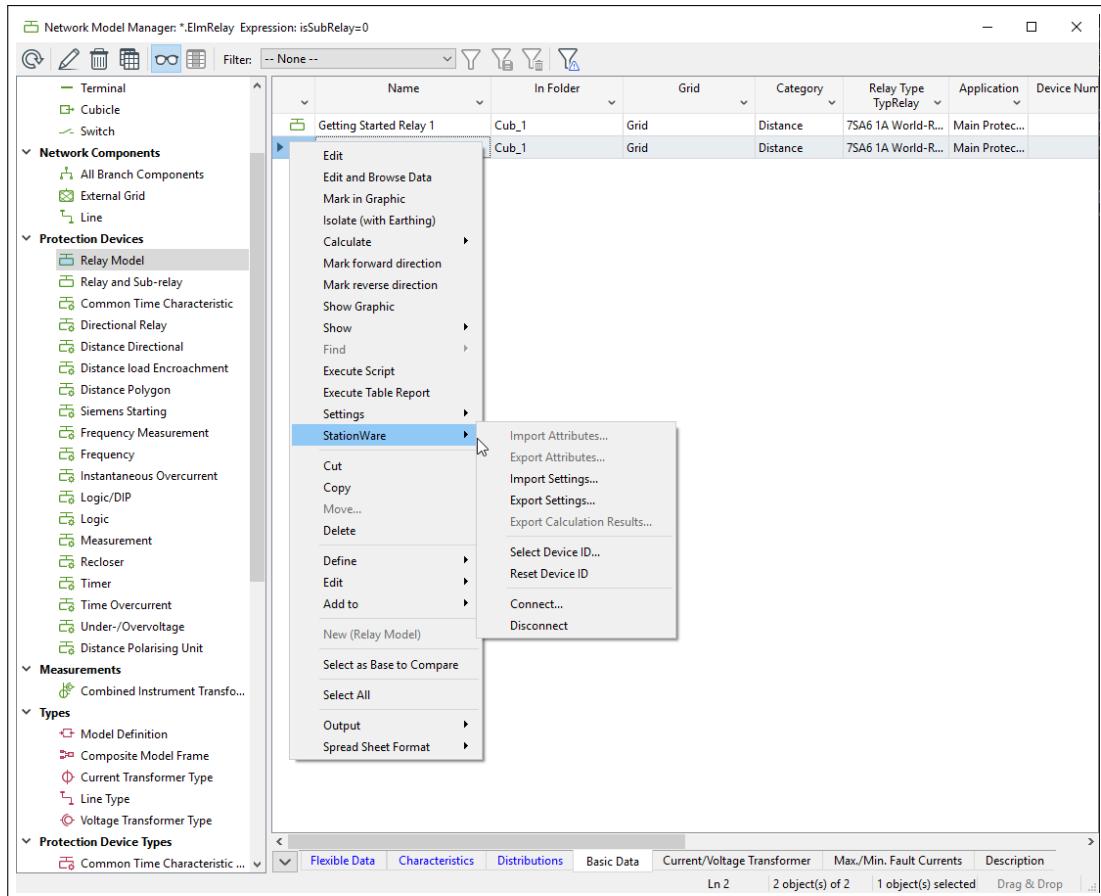


Figure 24.13.12: Device context menu

Figure 24.13.21 shows the dialog settings.

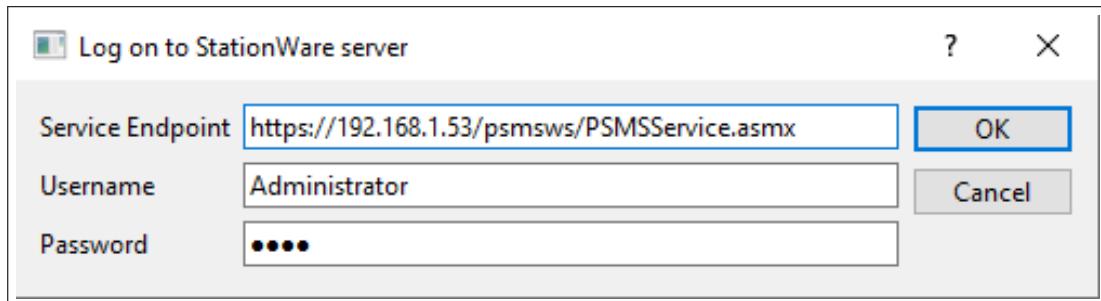


Figure 24.13.13: Log on dialog

- 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/psmssws/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 24.13.6) 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 24.13.14.

- Navigate to the *Getting Started* substation,
- select the *Getting Started Relay 1* device,
- press **OK**.

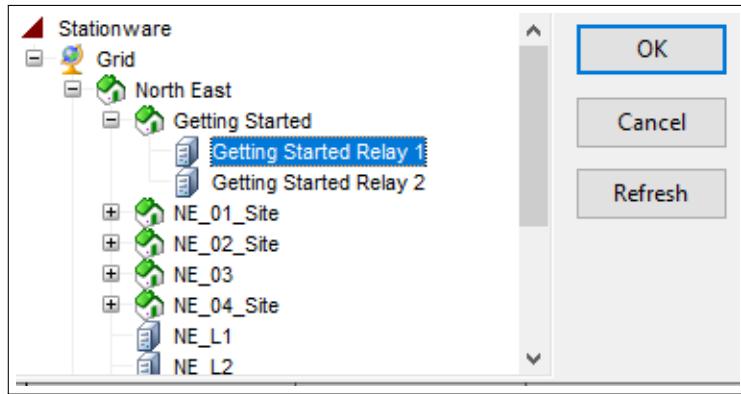


Figure 24.13.14: Browser dialog

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 24.13.12.
- 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 24.13.6 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.

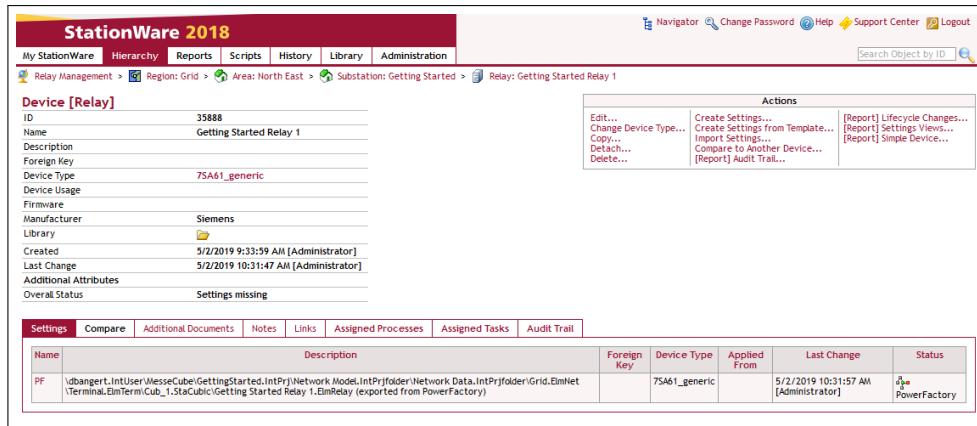


Figure 24.13.15: Device detail page

- Navigate to the relay detail view of the *Getting Started Relay 1* relay (see Fig. 24.13.15)

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. 24.13.16).

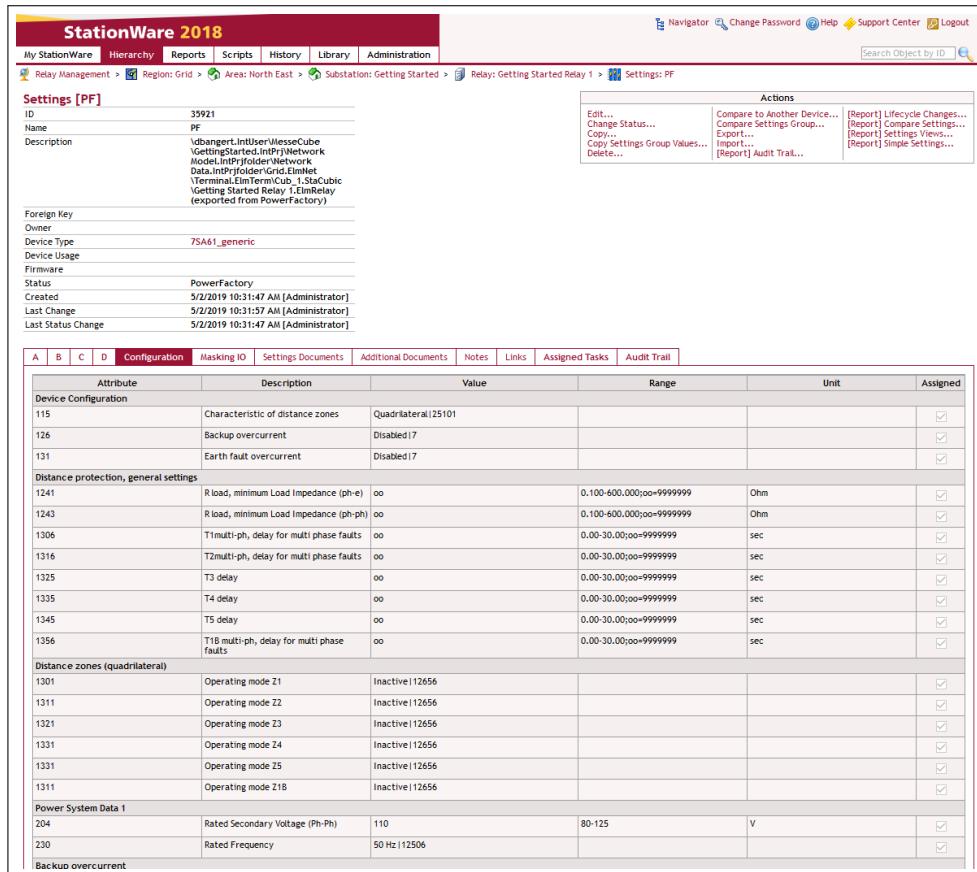


Figure 24.13.16: Setting detail page

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 24.13.12.
- 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 24.13.6: Import/Export Options.

#### 24.13.5.2 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 'StationWare 2018' interface with the 'Device [Line]' page open. The main area displays the device details for 'NE\_L1'. A red box highlights the 'Additional Attributes' section, which lists various parameters like Overall Status, Terminal I, Terminal J, Length, Laying, Substation I, and Substation J. Below this is a table of additional attributes with columns for Name, Value, and Unit. A red box also highlights this table. At the bottom of the page, there are tabs for Settings, Compare, Additional Documents, Notes, Links, Assigned Processes, Assigned Tasks, Audit Trail, and Type Data, with 'Type Data' being the active tab.

Name	Value	Unit
Rated Voltage	400	
Rated Current	1	
Nominal Frequency	50	
PhaseNR	3	
NeutralLines	0	
Resistance R(20°C)	0.023	
Reactance X	0.25	
Resistance R0	0.2	
Reactance X0	1	
Conductor Material	Aluminium	
Max Operational Temperature	80	
Susceptance B	3.14159	
Susceptance B0	0.7	

Figure 24.13.17: 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 24.13.5). All actions are similar to those described for settings (see Section 24.13.5.1).

### 24.13.5.3 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 24.13.18).

**Important:** Process lifecycle must posses a phase named “*PowerFactory*” of type “Planning”.

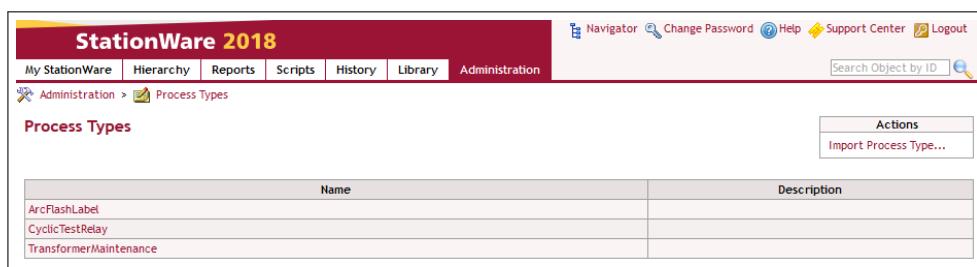


Figure 24.13.18: Process types page in *StationWare*

After being defined, the process should be created and have a device assigned to it.

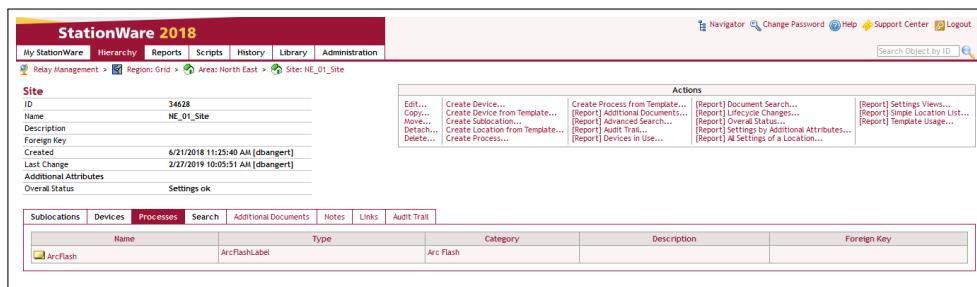


Figure 24.13.19: Location page where process is created

#### 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 24.13.5). 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 24.13.5.1.

#### Export of results

Refer to similar section 24.13.5.1.

### 24.13.6 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 24.13.12)
- in the Data Manager as shown in Figure 24.13.20.

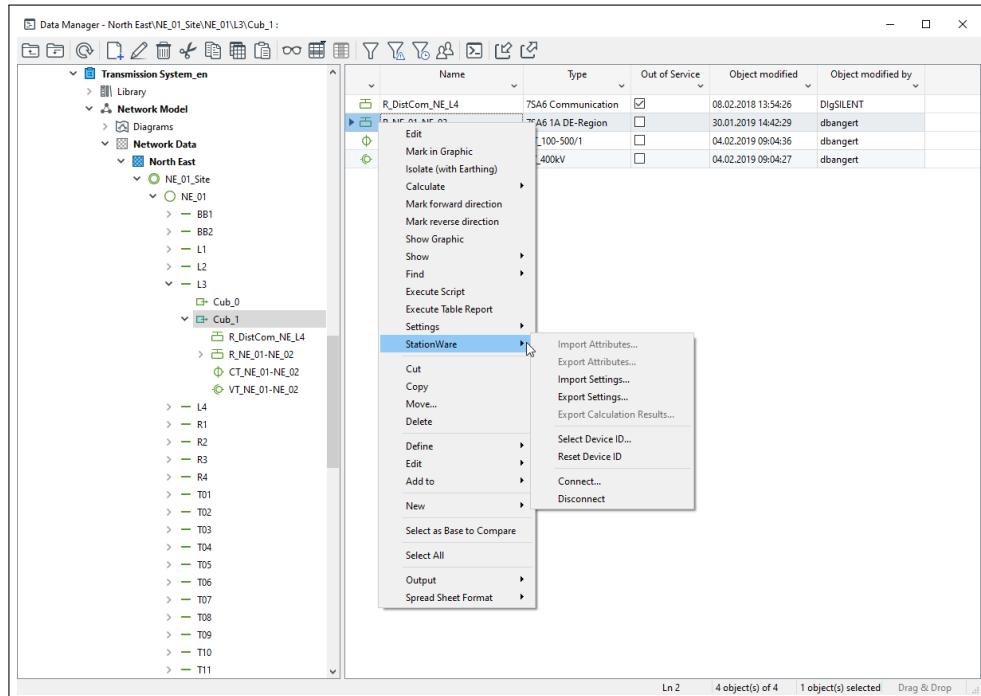


Figure 24.13.20: Device context menu

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 24.13.6: The *ComStationware* Object.

**Export X...** does the same for the export direction.

**Select Device ID...** starts the Browser dialog (Figure 24.13.24) to link this device to a *StationWare* device. The dialog is subject of Section 24.13.6 : 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 24.13.6. 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.

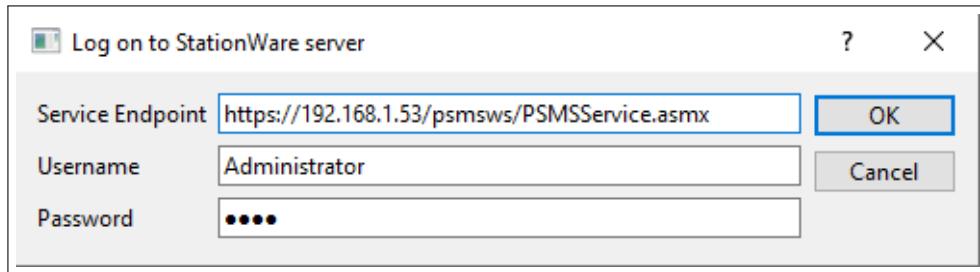


Figure 24.13.21: Log on dialog

Working with *PowerFactory* for the first time, the *StationWare* server is required, and the Logon dialog is as shown in Figure 24.13.21.

The *StationWare* connection options are stored in the user settings (Figure 24.13.22). After each successful logon the user settings are updated.

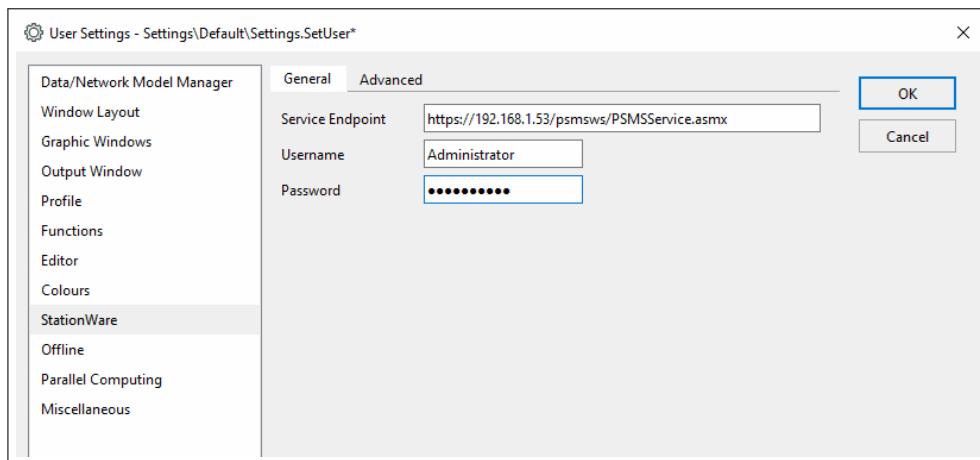


Figure 24.13.22: Log on dialog

As mentioned in the Architecture section (Section 24.13.2) *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 24.13.3: Device) the *StationWare* device ID is stored as Foreign Key in the e.g. *ElmRelay* object dialog (Description page) as shown in Figure 24.13.23.

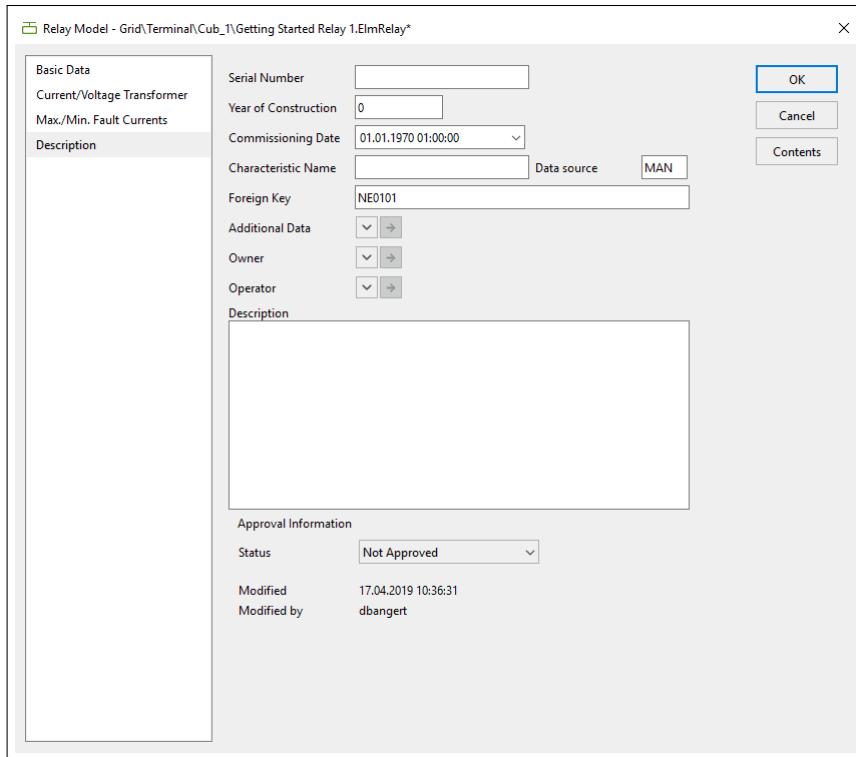


Figure 24.13.23: *ElmRelay* dialog

A more convenient way is to use the Browser dialog shown in Figure 24.13.24. The dialog allows to browse through the *StationWare* location hierarchy and select a device. The hierarchy data is cached to minimise network accesses. Due to 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.

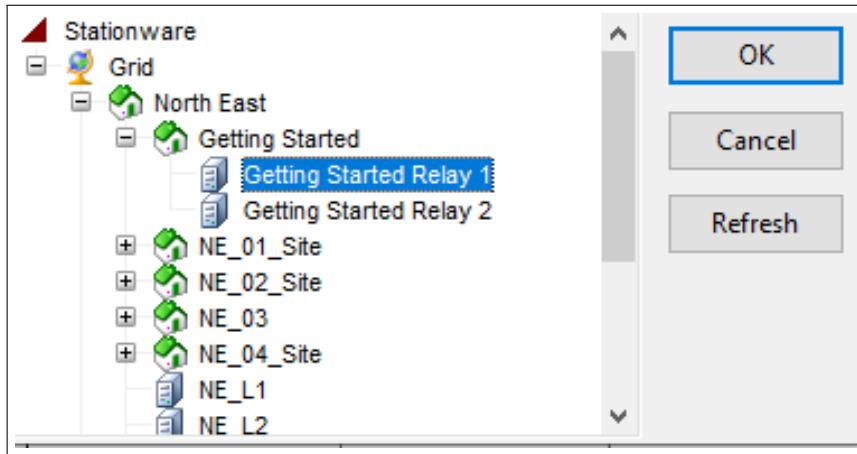


Figure 24.13.24: Browser dialog

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 parametrised 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 24.13.6: 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**.

## 24.14 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).

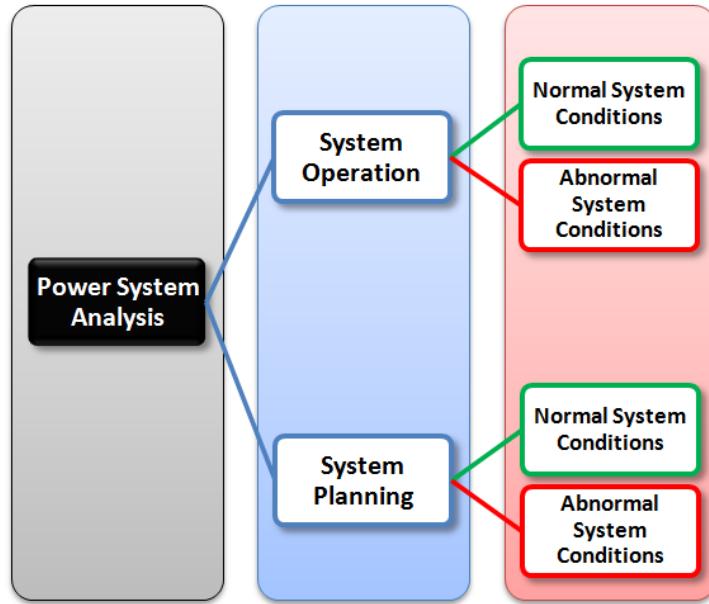


Figure 25.1.1: Power System Analysis: System Operation and System Planning

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.

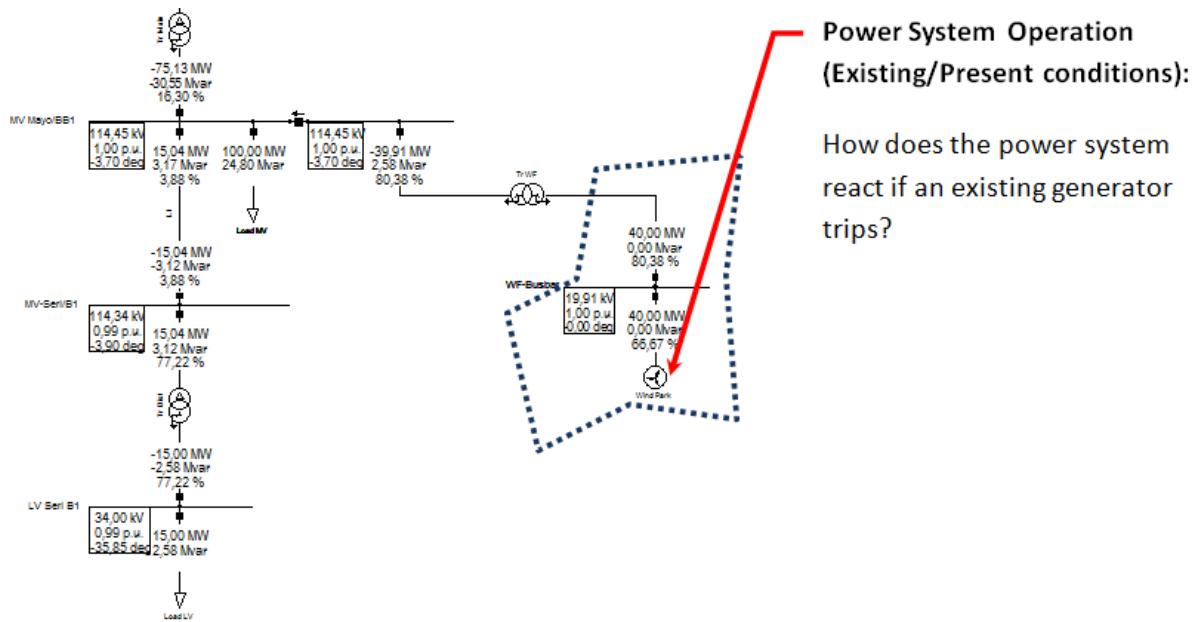


Figure 25.1.2: Power system operation example

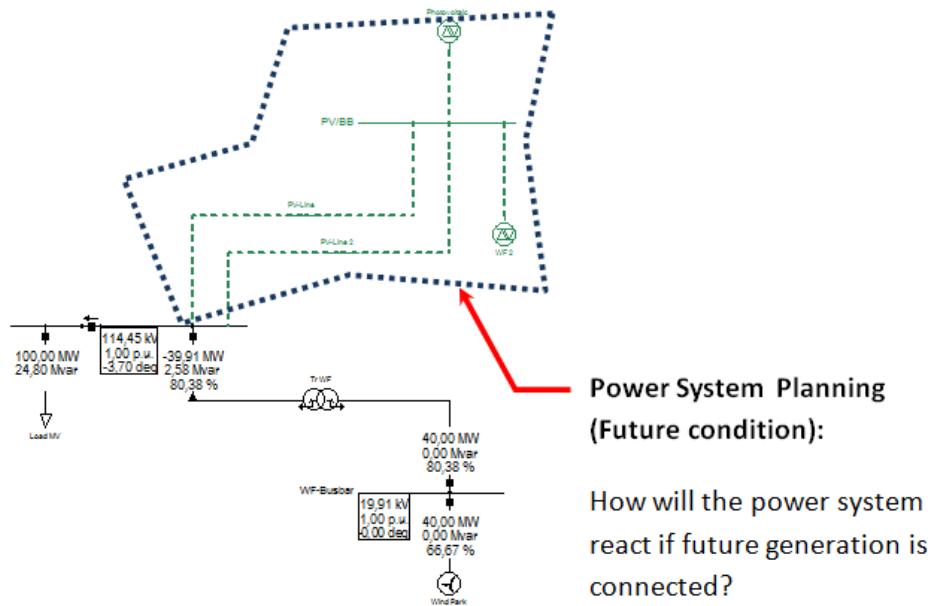


Figure 25.1.3: Power system planning example

## 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).

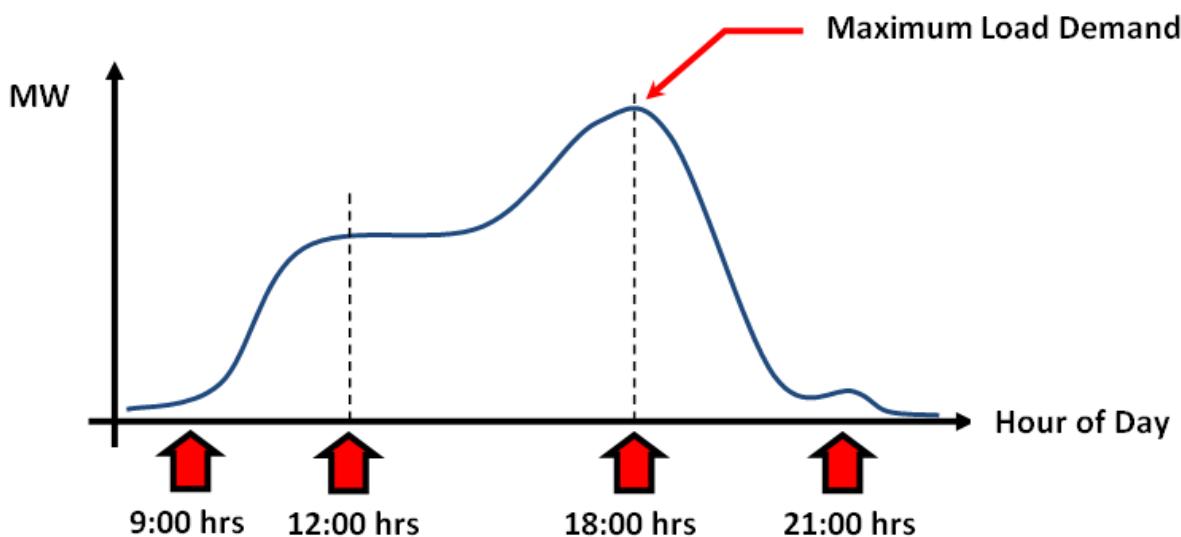


Figure 25.2.1: Example of a Load Demand Curve

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 48.4.6 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 [48.4.6](#) 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 (25.25) described in Section [25.4.5](#) 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 bus bar 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 bus bar has been defined. The static generator with the highest nominal 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.

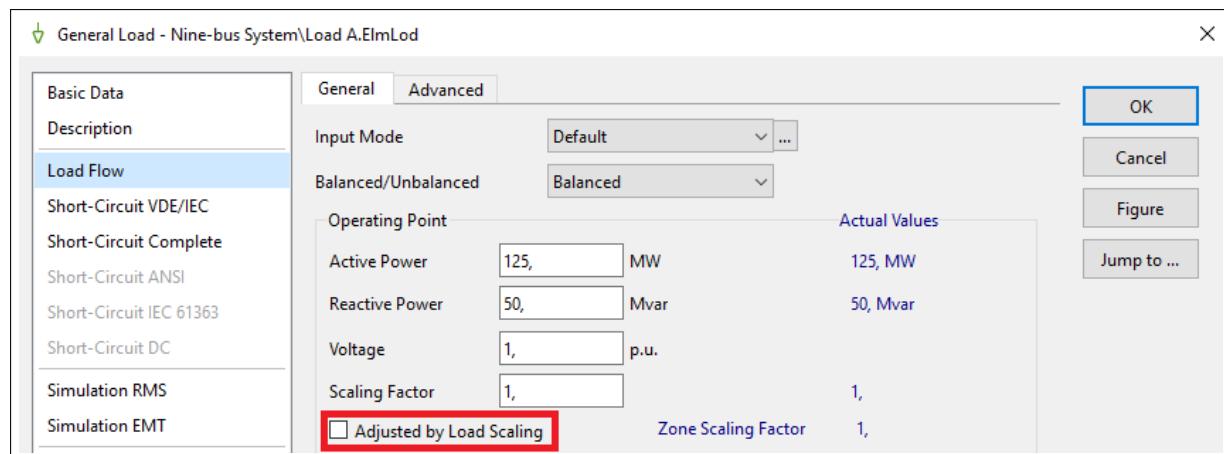


Figure 25.3.1: *Adjusted by Load Scaling* option in the Load Flow page of the Load element (*ElmLod*)

**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.

#### 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.

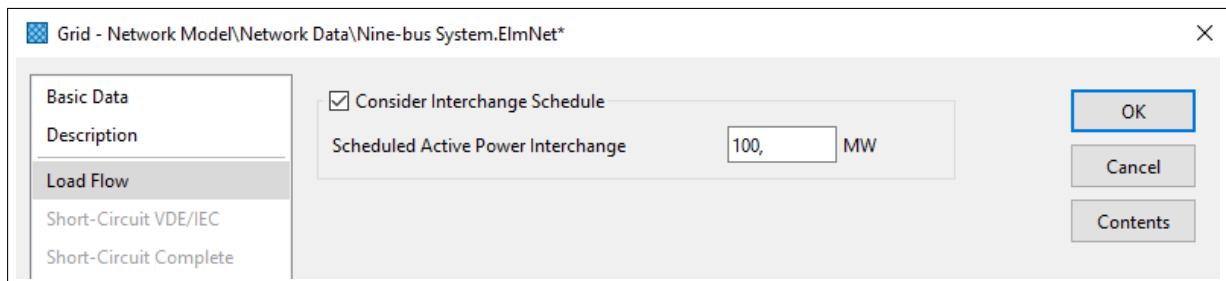


Figure 25.3.2: *Consider Interchange Schedule* option in the Load Flow page of the Grid element (*ElmNet*)

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.

## Reference Machine Selection

The user is able to select which machine is used as the reference (slack) machine, using the checkbox *Reference Machine* on the Load Flow page of the element (*ip\_ctrl*). However, in cases where there is no reference machine defined in the network, or an isolated part of the network, *PowerFactory* can select which element should act as the reference machine, according to certain criteria. 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 8.1.2.3.

### Automatic Determination of Reference Machine if none selected

If *PowerFactory* is required to designate a reference machine, the following network elements are considered: *ElmXnet*, *ElmPvsys*, *ElmSym*, *ElmVsc* and *ElmRec*. The logic used is designed to ensure that a large machine is selected, so scaling factors are applied to the size (Snom, or Unom for Voltage source or Pgen for ward equivalent voltage source), according to the table below. This means that generally speaking the choice of reference machine is determined according to element class. (Note that i\_spin=1 means that *Spinning if circuit breaker is open* is selected.)

Element	Factor used for selection
Extended Ward equivalent voltage source	Pgen +0.001
<i>ElmXnet</i> , <i>ElmSym</i> with i_spin=1	Snom
Ward equivalent voltage source	0.1*(Pgen+1)
<i>ElmSym</i> with i_spin=0 Method 2, <i>ElmVsc</i> with controlled AC busbar	0.01*Snom
Voltage source Normal operation	0.001*Unom*usetp
<i>ElmVsc</i> if not Vac/phi control active	0.00001*Snom
<i>ElmRec</i> , <i>ElmPvsys</i> , <i>ElmSym</i> with i_spin=0 Method 2	-1

Table 25.3.1: Reference machine prioritisation factors (no machine selected by user)

### Automatic Determination of Reference Machine if more than one selected

If several machines have been designated as reference machine, a similar prioritisation according to element class and size takes place as that described above. The scaling factors below are used:

Element	Factor used for selection
<i>ElmXnet</i>	10000*Snom
<i>ElmGenstat</i> , <i>ElmPvsys</i> , <i>ElmSym</i>	100*Snom
<i>ElmSym</i> with i_spin=0 Method 2, <i>ElmVsc</i> with controlled AC busbar	0.01*Snom
<i>ElmVac</i>	Unom*usetp
<i>ElmVsc</i>	0.01*Snom

Table 25.3.2: Reference machine prioritisation factors (several machines selected by user)

**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: Pmax-Pmin is used instead of Snom

- Active Power Reserve: Pmax-Pgini is used instead of Snom
- 

### 25.3.3 Advanced Options

#### 25.3.3.1 Tap Adjustment

##### Method

The *direct* 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. The *stepped* 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.

##### 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 36.2.1.

##### Considered Models for Active Power Limits

The following element classes can be independently selected:

- *ElmSym*
- *ElmStat*
- *ElmAsm*

- *ElmVsc*

#### 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:

- *ElmSym*
- *ElmStat*
- *ElmAsm*
- *ElmVsc*
- *ElmSvs*
- *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.)

#### 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 improved 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 ( $I_{max}$ ). To configure a project such that new substations are created with Bays, the user should go to *Project Settings* → *Single Line Graphics*→ *Insert substations with bays*.

## 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 %).

#### Convergence Options

- **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.

#### 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.

#### **Settings - Automatic step size**

This 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.

#### **25.3.4.3 Initialisation**

##### **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.

##### **No Initialisation (no flat-start)**

Initialises a load flow from a previously convergent solution (no flat-start).

##### **Consideration of transformer winding ratio**

Sets the manner in which voltage initialisation takes place at nodes. This option, enabled by default, means that *PowerFactory* will automatically consider off nominal transformer ratios as part of the initialisation process.

##### **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°), it will be broken down into two or more steps, in successive outer loops.

#### **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.

#### Show Verification Report

Produces a table in the output window with a list of overloaded power system elements and voltage violations, according to the following values:

- **Max. Loading of Edge Element:** reference value of the maximum loading used by the *Verification Report*.
- **Lower Limit of Allowed Voltage:** reference value for the minimum allowed voltage used by the *Verification Report*.
- **Upper Limit of Allowed Voltage:** reference value for the maximum allowed voltage used by the *Verification Report*.
- **Output:** displays the report format definition that will be used. The arrow button → can be pressed to edit or inspect the report settings.

#### Check Control Conditions

This option is selected by default and lists all elements in the output window whose control conditions (e.g. reactive power limits, etc) have not been fulfilled. The arrow button → allows the user to select which control devices should be checked:

- 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 and motors that will be used only during the execution of the Load Flow Calculation.

#### Load Scaling Factor

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*)

#### Generation Scaling Factor

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

### **Motor Scaling Factor**

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

### **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.

## **25.3.7 Low Voltage Analysis**

As explained in Sections [25.4.2](#) and [36.2.1](#), low voltage loads (*ElmLodlv* and *ElmLodvp*) are modelled in *PowerFactory* with fixed and variable (stochastic) components. The parameters which define these fixed and variable components are set in both the load flow command dialog (i.e. globally), and in the load types' dialogs (i.e. locally) according to the settings defined below.

### **Consider Coincidence of Low-Voltage Loads**

Calculates a 'low voltage load flow' as described in Sections [25.4.2](#) and [25.3.7](#), where load coincidence factors are considered, so as to produce maximum branch currents and maximum voltage drops. Since coincidence factors are used, the result of low voltage analysis will not obey Kirchhoff's current law. After the load flow has been successfully executed, maximum currents (*Imax*), maximum voltage drops (*dumax*) and minimum voltages (*umin*, *Umin*) are displayed in every branch element and at every busbar. The usual currents and voltages represent here average values of voltages and currents. Losses are calculated based on average values, and maximum circuit loading is calculated using maximum currents.

### **Scaling Factor for Night Storage Heaters**

This is the factor by which the night storage heater power (as found in *Low Voltage Load* elements) is multiplied for all low voltage loads.

### **Definition of Fixed Load per Customer**

The fixed load is the non-stochastic component of the load, which is not subject to coincidence factors. The active and reactive power defined in this field, multiplied by the number of customers (defined in the load element itself), are added to the fixed load component defined for each low voltage load (*ElmLodlv* and *ElmLodvp*). For further information about LV loads refer to the [Technical References Document](#).

### **Definition of Variable Load per Customer**

The variable component of low voltage loads can be globally defined using the parameters in this section or by specifically defining LV load types for the target loads.

The *Max. Power per Customer* is the independent maximum kVA per customer. This value, multiplied by the *Coincidence Factor* (*ginf*) (see Section [25.4.2](#)), gives the "Average Power" per customer, which is used in load flow calculations.

The 'total' maximum variable power per load is calculated using the *Max. Power per Customer*, the *Coincidence Factor* (*ginf*), and the number of customers (defined in the load element itself) as described in Section [25.4.2](#).

---

**Note:** The factors defined in the section Definition of *Variable Load per Customer* are used as global data for the load flow calculation. If specific LV load types are defined, the locally-defined data in the type is used by the corresponding loads. For all other LV loads with no type assigned, the global data from the load flow command is used.

---

### Voltage Drop Analysis

For the consideration of the stochastic nature of loads, *PowerFactory* offers two calculation methods:

- Stochastic Evaluation
- Maximum Current Estimation

The *Stochastic Evaluation* method is the more theoretical approach, and can also be applied to meshed network topologies. The *Maximum Current Estimation* method 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.

## 25.4 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 ( $K_{pf}$ ) of these generators, those with a higher  $K_{pf}$  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:

$$P_i = P_{i-dispatch} + \Delta P_i \quad (25.3)$$

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.

$$\Delta P_i = K_{pf-i} \cdot \Delta f \quad (25.4)$$

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:

$$\Delta f = \frac{\Delta P_{Tot}}{\sum K_{pf}} \quad (25.5)$$

where  $\Delta P_{Tot}$  corresponds to the active power change sum of every generator:

$$\Delta P_{Tot} = \sum_{j=1}^n \Delta P_j \quad (25.6)$$

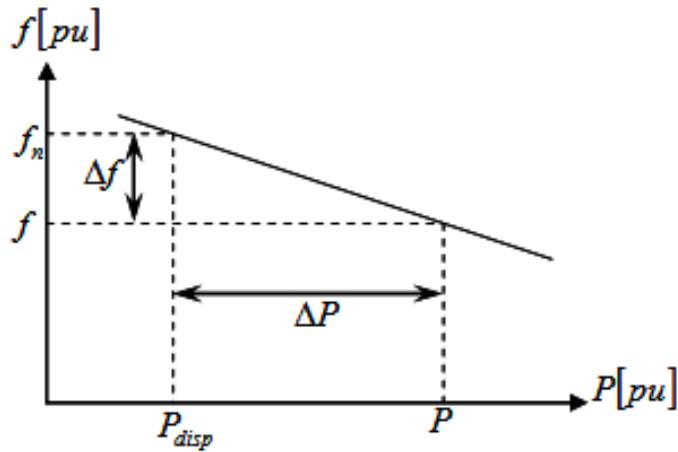
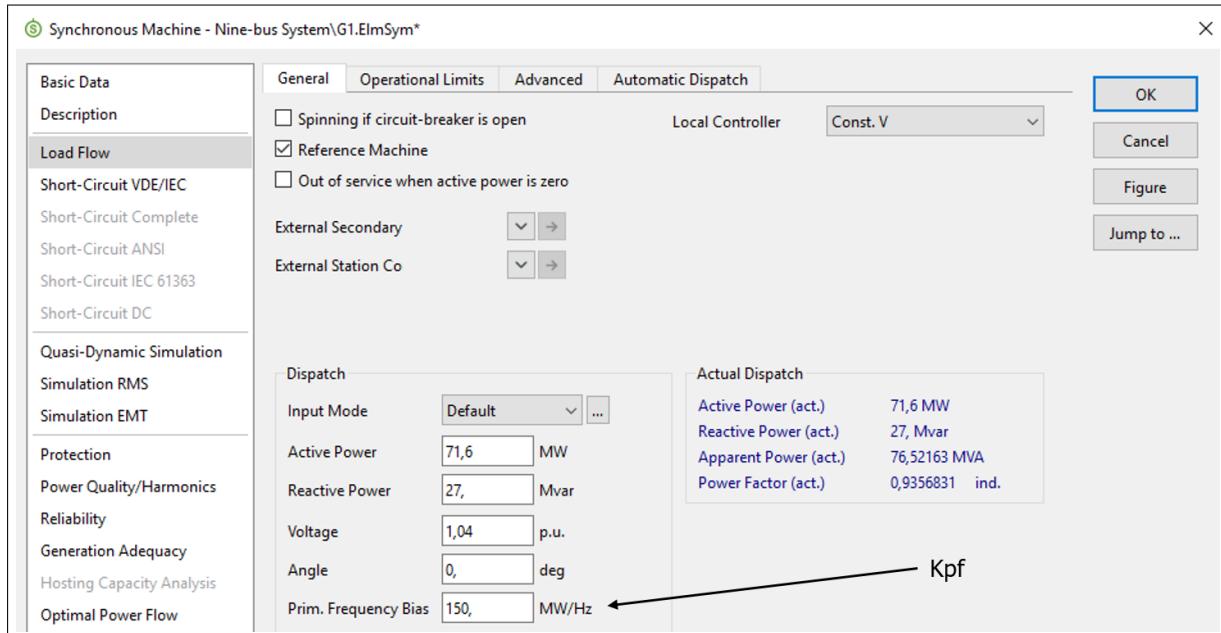


Figure 25.4.1: Primary Frequency Bias


 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.

$$\Delta P_{Tot} = 5 \text{ MW} - 7 \text{ MW} = -2 \text{ MW} \quad (25.7)$$

The total frequency deviation is therefore:

$$\Delta f = \frac{-2 \text{ MW}}{5 \text{ MW}/\text{Hz}} = -0.4 \text{ Hz} \quad (25.8)$$

The active power deviation between the dispatch setpoint of each generator and its primary controlled output is therefore:

$$\Delta P_G1 = 2 \text{ MW}/\text{Hz} \cdot -0.4 \text{ Hz} = -0.8 \text{ MW} \quad (25.9)$$

$$\Delta P_G2 = 2 \text{ MW}/\text{Hz} \cdot -0.4 \text{ Hz} = -0.8 \text{ MW} \quad (25.10)$$

$$\Delta P_G3 = 1 \text{ MW}/\text{Hz} \cdot -0.4 \text{ Hz} = -0.4 \text{ MW} \quad (25.11)$$

Finally, The primary controlled output of each generator is therefore:

$$P_G1 = 2 \text{ MW} - 0.8 \text{ MW} = 1.2 \text{ MW} \quad (25.12)$$

$$P_G2 = 2 \text{ MW} - 0.8 \text{ MW} = 1.2 \text{ MW} \quad (25.13)$$

$$P_G3 = 3 \text{ MW} - 0.4 \text{ MW} = 2.6 \text{ MW} \quad (25.14)$$

**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 decrease/increase in the system frequency. The contribution of each individual generator towards the total additional power required is proportional to its inertia. If the *Active Power Control* option *According to Inertias* is selected in *PowerFactory*'s load flow command, the power balance is established by all generators. 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:

$$P_i = P_{i-dispatch} + \Delta P_i \quad (25.15)$$

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 inertia gain ( $J \cdot \omega_n \cdot 2\pi$ ) and the total frequency deviation, as follows:

$$\Delta P_i = J \cdot \omega_n \cdot 2\pi \cdot \Delta f \quad (25.16)$$

where

$\Delta f$  is the total frequency deviation and  $J$  is the moment of inertia, calculated as

$$J = S_n \cdot \frac{T_{ags}}{\omega_n^2} \quad (25.17)$$

where

$\omega_n$  is the rated angular velocity,

$S_n$  is the generator nominal apparent power and

$T_{ags}$  is the acceleration time constant rated to  $S_n$

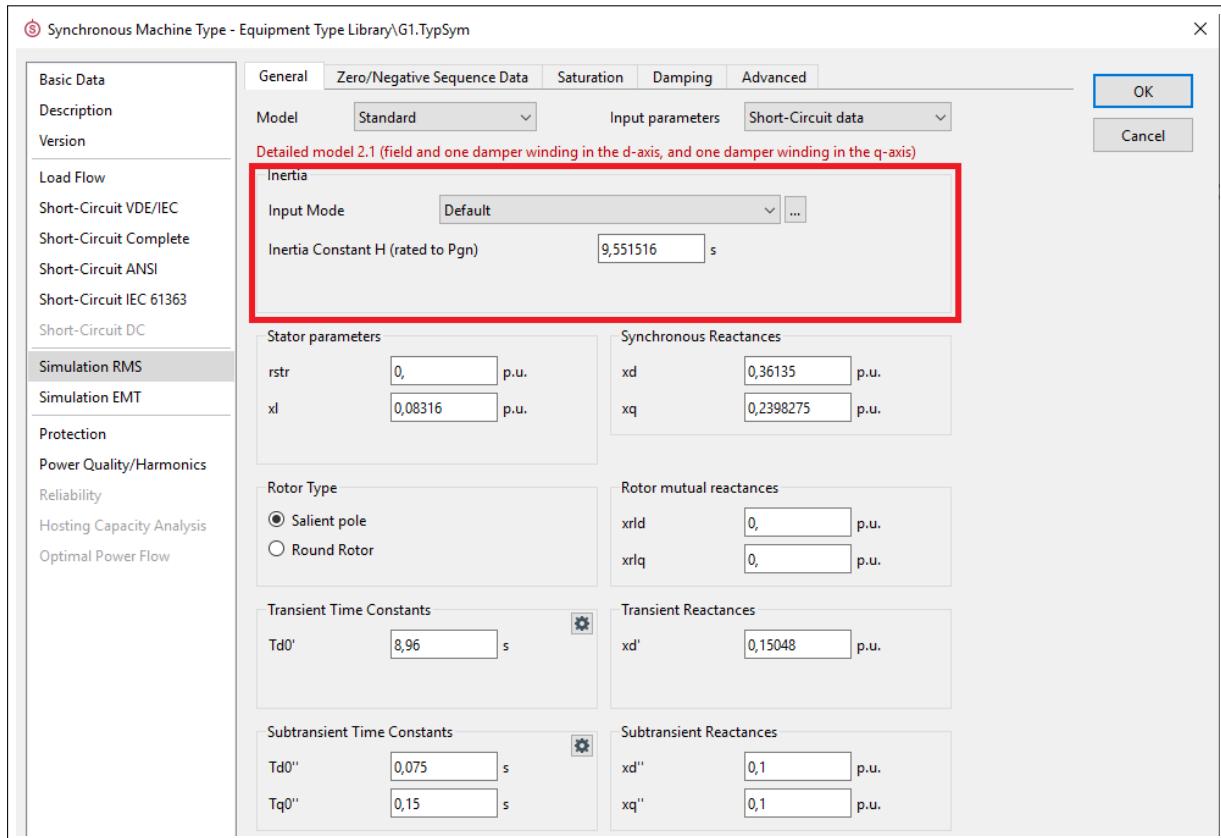


Figure 25.4.3: Inertia/Acceleration Time Constant Parameter of the Synchronous Machine Type (TypSym). *Simulation RMS* Page

Figure 25.4.4 illustrates the different types of active power control.

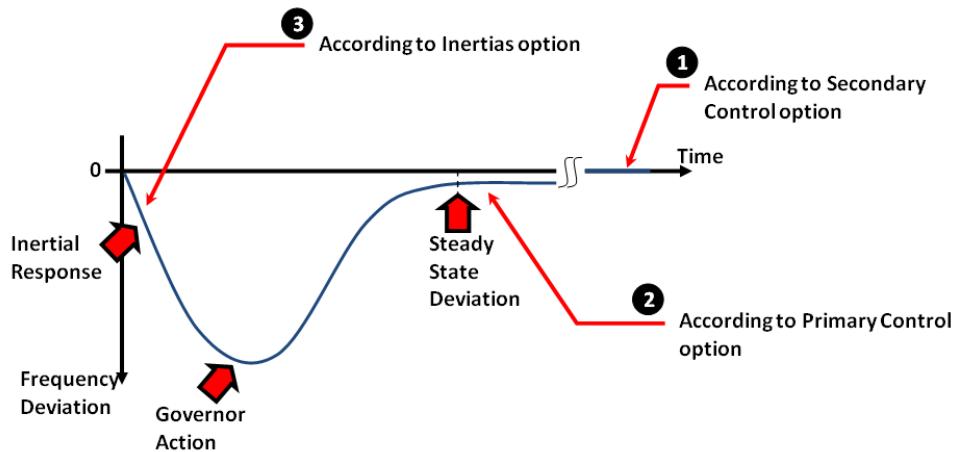


Figure 25.4.4: Frequency Deviation Following an Unbalance in Active Power

**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).

#### 25.4.1.2 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 (25.18) and (25.19). In these equations, the subscript 0 indicates the initial operating condition as defined in the input dialog box of the Load Type.

$$P = P_0 \left( aP \cdot \left( \frac{v}{v_0} \right)^{e_{-aP}} + bP \cdot \left( \frac{v}{v_0} \right)^{e_{-bP}} + (1 - aP - bP) \cdot \left( \frac{v}{v_0} \right)^{e_{-cP}} \right) \quad (25.18)$$

where,

$$cP = (1 - aP - bP)$$

$$Q = Q_0 \left( aQ \cdot \left( \frac{v}{v_0} \right)^{e_{-aQ}} + bQ \cdot \left( \frac{v}{v_0} \right)^{e_{-bQ}} + (1 - aQ - bQ) \cdot \left( \frac{v}{v_0} \right)^{e_{-cQ}} \right) \quad (25.19)$$

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 nominal 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.

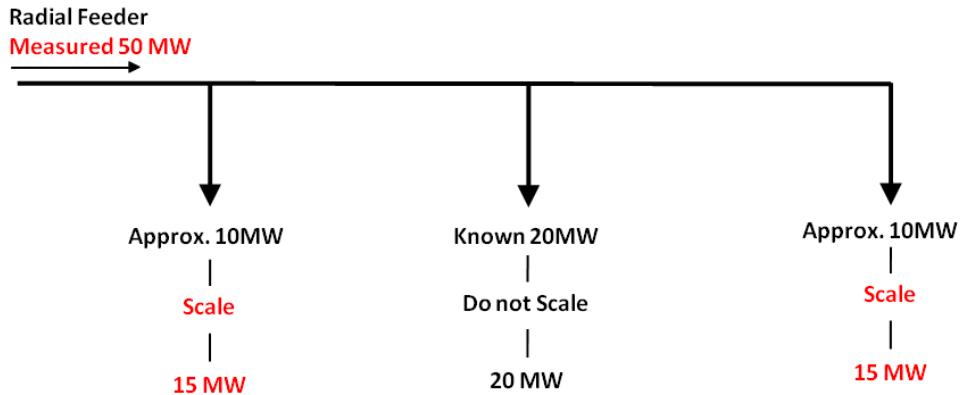


Figure 25.4.5: Radial Feeder. Feeder Load Scaling Option

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.

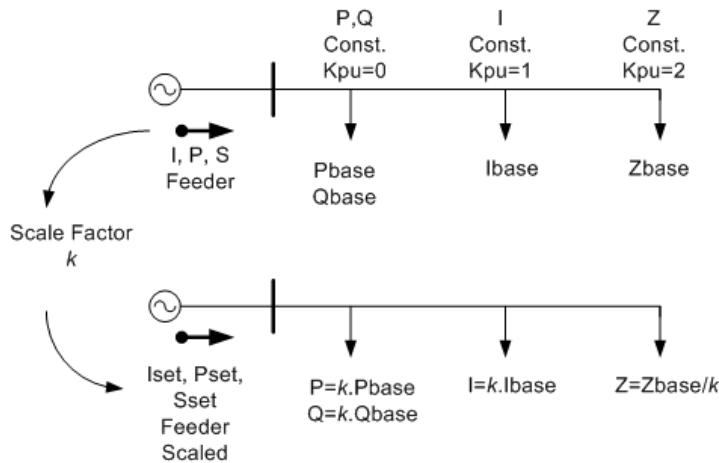


Figure 25.4.6: Feeder Load Scaling Factor Considering Different Behaviour of Loads

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 L2. 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 (L1, L2, L3 and L4), 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 L1 and L2 according to its setting, while Feeder 2 will scale loads L3 and L4. If Feeder 2 is placed *Out of Service*, then Feeder 1 will scale all the loads along the radial path (L1, L2, L3 and L4).

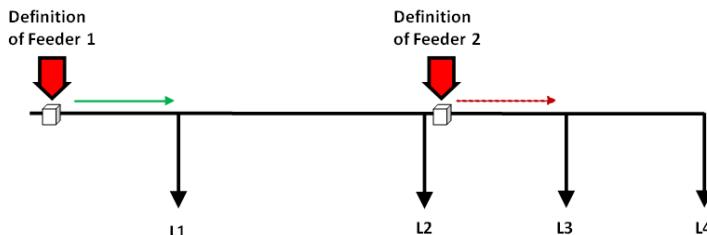


Figure 25.4.7: Nested Feeder Definition

**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 15.5 (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 cosphiSet. Each scalable load inside the feeder is then set to the same powerfactor *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  $\text{angleset}$  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  $\text{angleset} = \text{Arg}(u * \bar{cur})$ .

*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:

$$S_n = S_o * scale \quad (25.20)$$

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*:

$$\begin{aligned} P_n &= S_o * \cos(scalephi) \\ Q_n &= S_o * \sin(scalephi), \end{aligned} \quad (25.21)$$

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 Coincidence of Low Voltage Loads

In a low voltage system every load may consist of a fixed component with a deterministic amount of power demand plus a variable component comprising many different, small loads, such as lights, refrigerators, televisions, etc., whose power varies stochastically between zero and a maximum value. Under such conditions, *PowerFactory* uses a probabilistic load flow calculation, which is able to calculate both maximum and average currents as well as the average losses and maximum voltage drops. The probabilistic load flow calculation used by *PowerFactory* can be applied to any system topology, including meshed low-voltage systems.

*PowerFactory*'s probabilistic load flow calculation uses low voltage loads comprised of several customers with fixed and variable (stochastic) demand components. The maximum value of the variable component (which is dependent upon the number of customers,  $n$ ) is described by the following formula:

$$S_{max}(n) = n \cdot g(n) \cdot S_{max} \quad (25.22)$$

Where  $S_{max}$  is the maximum variable 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:

$$g(n) = g_\infty + \frac{1 - g_\infty}{\sqrt{n}} \quad (25.23)$$

The average value of the variable component is:

$$g(n) = g_\infty \cdot S_{max} \quad (25.24)$$

---

**Note:** Low voltage loads can be represented in *PowerFactory* by Low Voltage Load (*ElmLodlv*) elements which can be directly connected to terminals or by Partial Low Voltage Loads (*ElmLodlvp*) which are defined along transmission lines/cables (see the Definition of Line Loads section on the Load Flow page of transmission line/cable elements - *ElmLne*).

---

### 25.4.5 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:

$$R = R_{20}[1 + \alpha(T - 20^\circ C)] \quad (25.25)$$

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  $T$

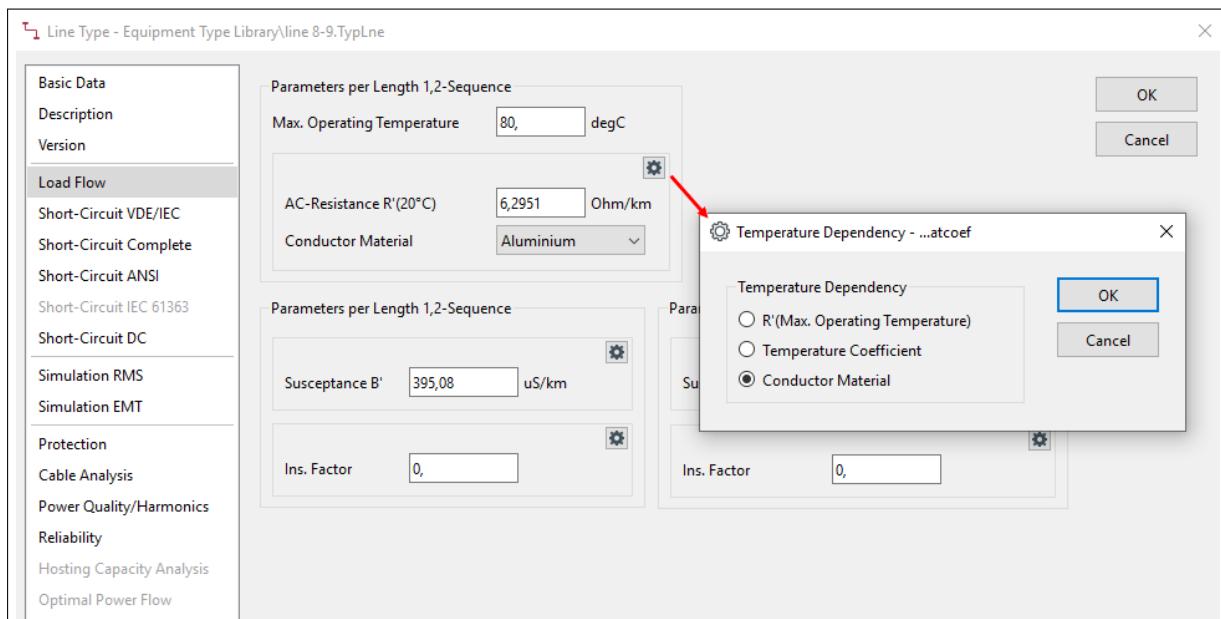


Figure 25.4.8: Temperature Dependency Option Setting in the Load Flow page of the line type (*TypLne*)

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 25.4.1 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 25.4.1: Electrical Resistivities and Temperature coefficients of Aluminium and Copper

## 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 9: Network Graphics, section 9.5. 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 (grid icon) 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 Predefined Report Formats (ASCII Reports)

In *PowerFactory* there are predefined report formats also called ASCII reports, available to the user. These ASCII reports can be created by pressing the *Output Calculation Analysis* button (grid icon) located on the main menu (a load flow must be calculated first). Output reports are described in Chapter 19: Reporting and Visualising Results, section 19.4.

A *Verification Report* can be also printed out automatically each time a load flow calculation is executed (see Section 48.4.5).

#### 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. 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 percentaged 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

$$s_b := (1/\bar{S}) \max_{i=1, \dots, N} \{|S_i - \hat{S}|\}.$$

---

**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

In *PowerFactory* input (data that has been entered by the user) and output (parameters that have been calculated) data is kept separate. Output data, such as the new tap positions following an automatic tap adjustment calculation, does not overwrite the settings that the user originally entered, unless the user specifically commands this, using the  icon on the main toolbar.

---

**Note:** The corresponding input parameters of the database will be overwritten by the calculated values.

---

Updating the database may be performed for:

- Scaling factor of loads

- Transformer taps
- Capacitive Steps of Shunts/Filter
- P,Q of Loads
- P,Q of Asynchronous machines
- P,Q,V of Synchronous machines and Static Generators

**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 may be seen in the single line diagram, but it will be noticed that the input data parameter in the element data dialog is as originally entered. If the icon  is clicked, 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 *edit and browse*, 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 48.4.5).

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 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 [18], 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 [5] 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.

Engineering Recommendation G74 was used for the implementation of *Complete Method*.

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.

---

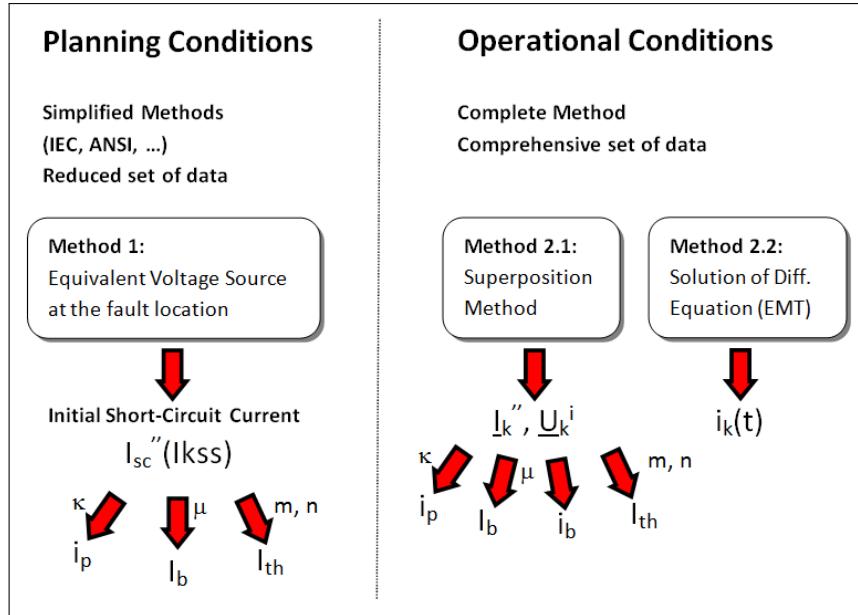


Figure 26.2.1: Areas of Application of Short-Circuit Calculations

According to IEC 60909 [18] the definition of the currents and multiplication factors shown in Figure 26.2.1 are as follows:

- $Ikss$  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).

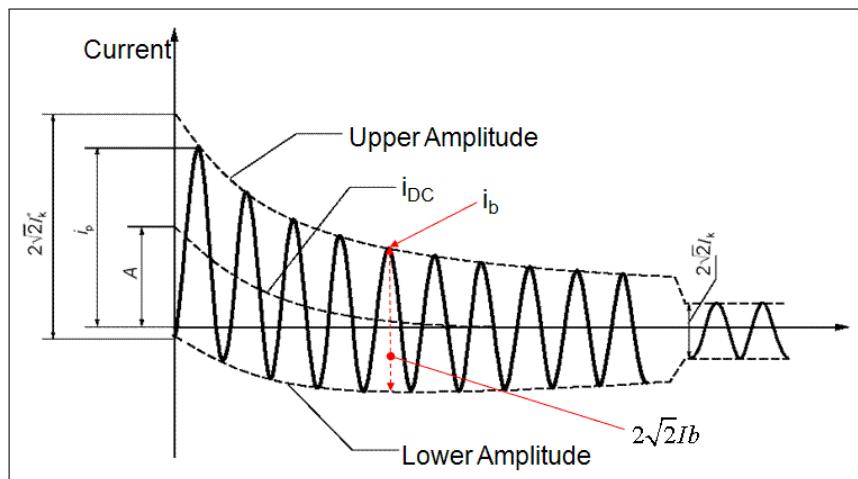


Figure 26.2.2: Short-Circuit Current Time Function

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 [18] (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).

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 [5]: ANSI/IEEE standard for the calculation of short-circuit currents in DC auxiliary systems in power plants and substations.

For AC and DC 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 [18] 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.

### 26.2.1 The IEC 60909/VDE 0102 Method

The IEC 60909/VDE 0102 [18] method uses an equivalent voltage source at the faulted bus and is a simplification of the superposition method (Complete Method). It is illustrated in Figure 26.2.3. The goal of this method is to accomplish a close-to-reality short-circuit calculation without the need for the preceding load-flow calculation and the associated definition of actual operating conditions. Figure 26.2.3 illustrates how 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.  $Ui = Un,i$
- Load currents are neglected, i.e.  $IOp = 0$ .
- 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 a network.

The short-circuit calculation based on these simplifications may be insufficient for some practical applications. Therefore, additional impedance correction factors are applied to the physical impedances of the network elements. This method is described in detail in the following section. Please note in

addition that both IEC 60909 [18] and VDE 0102 do not deal with single phase elements (except single phase elements in the neutral conductor).

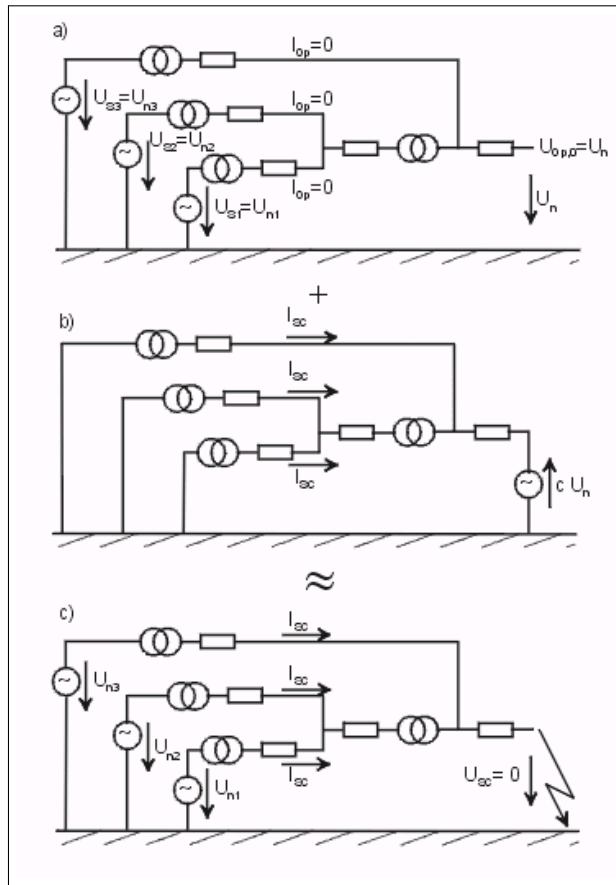


Figure 26.2.3: Illustration of the IEC 60909/VDE 0102 Method

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 1.5 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  $f_c = 20Hz$  (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 [26.4.4](#)

### IEC Impedance Correction Factors

The IEC 60909 method uses only the rated parameters of network elements. This is advantageous in that only little information is necessary to perform a short-circuit calculation. However, considering that, for example, 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.

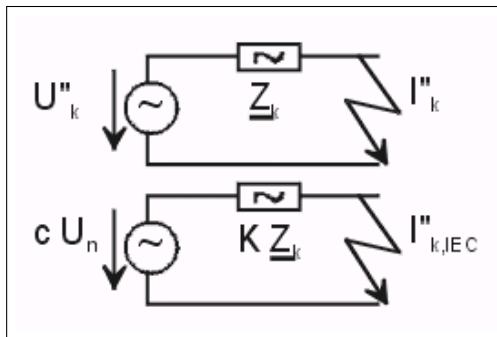


Figure 26.2.4: Principle of Impedance Correction (IEC/VDE Method)

As 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 (*TypLine*), 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** *kappaWD* can be entered, which is used to calculate the peak short-circuit current.

**Doubly Fed Induction Generator** *iWDmax* is the maximum instantaneous short-circuit current

**Doubly Fed Induction Generator** ratio RWD/XWD

**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 [18] 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.

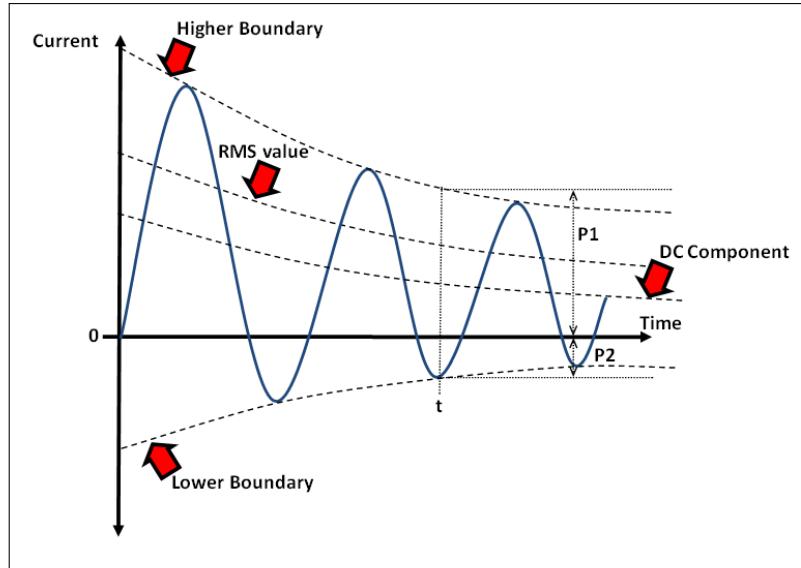


Figure 26.2.5: Asymmetrical Short-Circuit Current

The DC component “DC” is calculated according to the following equation:

$$DC = \frac{P1 - P2}{2} \quad (26.1)$$

The RMS value of the ac component (RMS) is then calculated as:

$$RMS = \frac{P1 + P2}{2.828} \quad (26.2)$$

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  $I_{symm}$

- $I_{27peak_m}$  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 (sometimes also known as the superposition method) is, in terms of system modelling, an accurate calculation method. The fault currents of the short-circuit are determined by overlaying a healthy load-flow condition before short-circuit inception with a condition where all voltage supplies are set to zero and the negative operating voltage is connected at the fault location. The procedure is shown in Figure 26.2.6. The initial point is the operating condition of the system before short-circuit inception (see Figure 26.2.6a). This condition represents the excitation conditions of the generators, the tap positions of regulated transformers and the breaker/switiching status reflecting the operational variation.

From these pre-fault conditions the pre-fault voltage of the faulted busbar can be calculated. For the pure fault condition the system condition is calculated for the situation where, the negative pre-fault busbar voltage for the faulted bus is connected at the fault location and all other sources/generators are set to zero (see Figure 26.2.6b). Since network impedances are assumed to be linear, the system condition after fault inception can be determined by overlaying (complex adding) both the pre-fault and pure fault conditions (see Figure 26.2.6c).

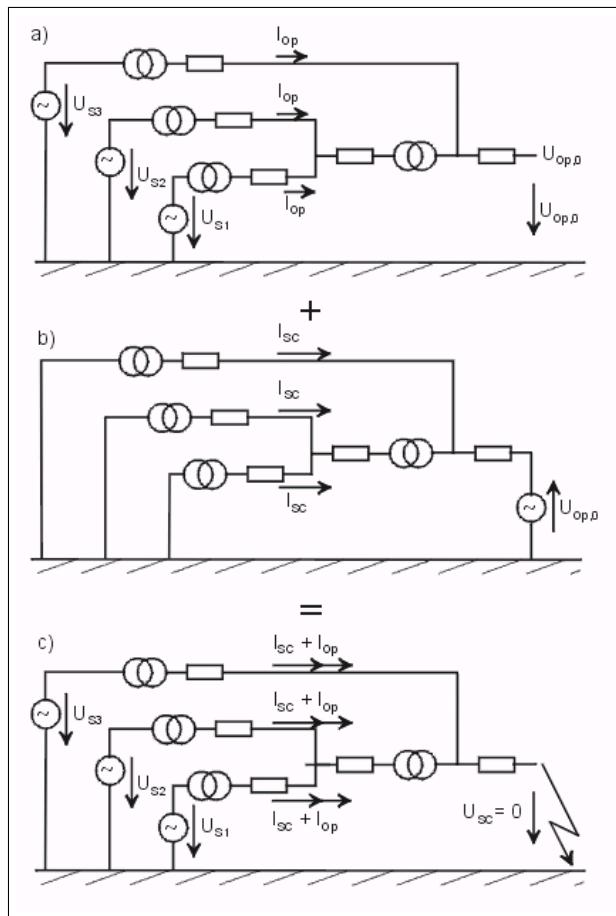


Figure 26.2.6: Illustration of the Complete Method

The *Complete Method* for calculating short-circuits has been improved in *PowerFactory* Version 14 as described below. Additionally, the quantities described below are shown 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[18]);
- 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).

#### 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) Method

The IEC 61660 standard [8] describes 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](#).

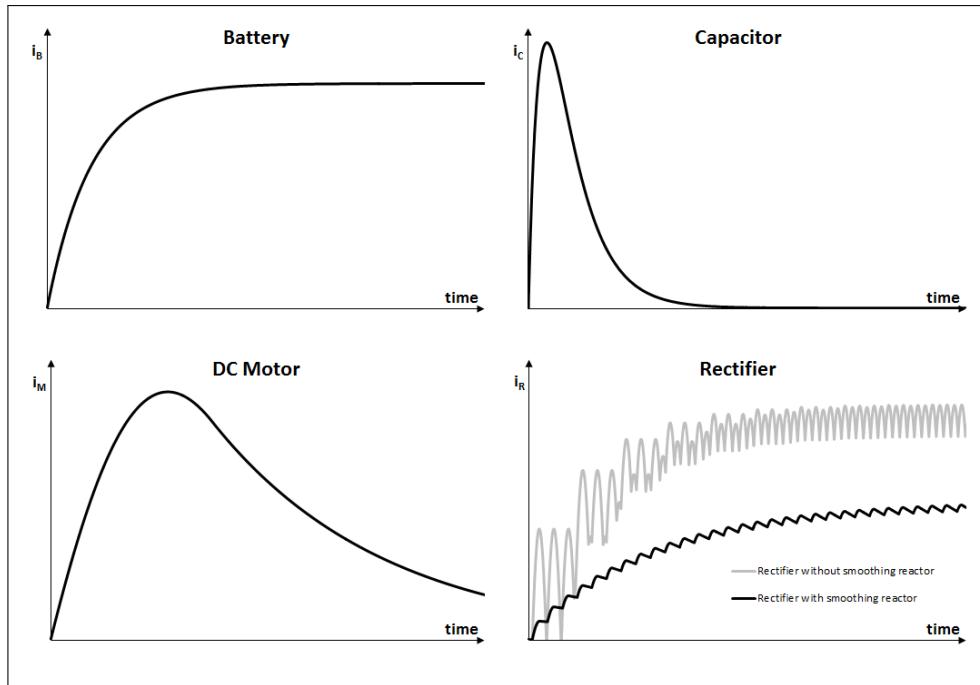


Figure 26.2.7: DC Short-Circuit Current Time Function of different sources

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.

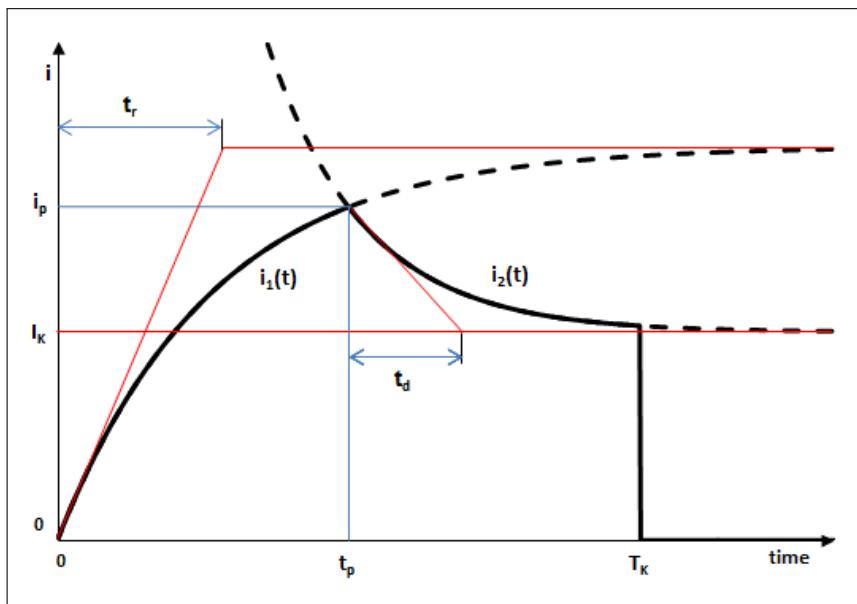


Figure 26.2.8: IEC 61660 standard DC short-circuit approximation function

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 [5] describes a recommended practice for the design of DC auxiliary power systems for nuclear and non-nuclear power stations. The standard provides guidance on the selection of equipment including ratings, interconnections, instrumentation, control and protection.

The IEEE 946 standard [5], is closely linked to General Electric's *Industrial Power Systems Data Book* [23]. 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 Document](#).

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

## 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...* 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-Sensitive Menu Execution

The short-circuit command may be executed from the context-sensitive menu in *PowerFactory* by selecting an element(s) in the single-line diagram, right-clicking and selecting one of the following options:

- *Calculate → 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.

- *Calculate → 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 *Calculate → Short-Circuit* is chosen from the context-sensitive menu, the *Fault Location* reference (bottom of dialog) is set to the selected element.
- When two or more busbars/terminals are selected and *Calculate → Short-Circuit* is chosen from the context-sensitive 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 ...* 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* .... 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.

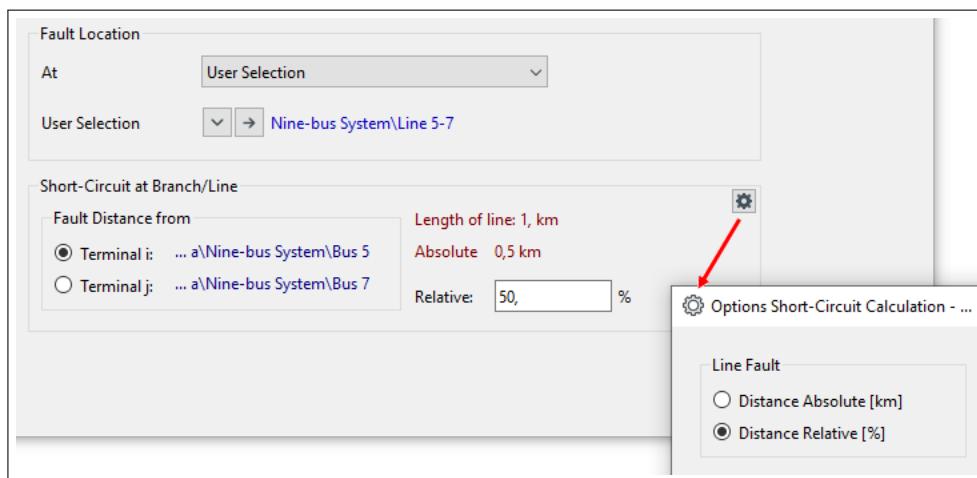


Figure 26.3.1: Configuration of Line/Branch Faults in *ComShc* dialog

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 *Calculate* → *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.

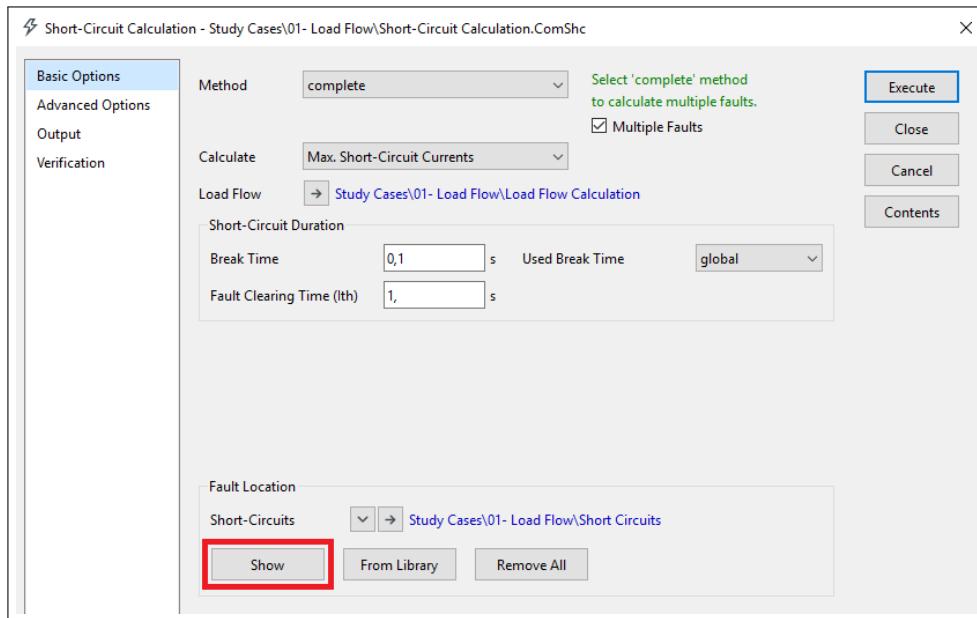
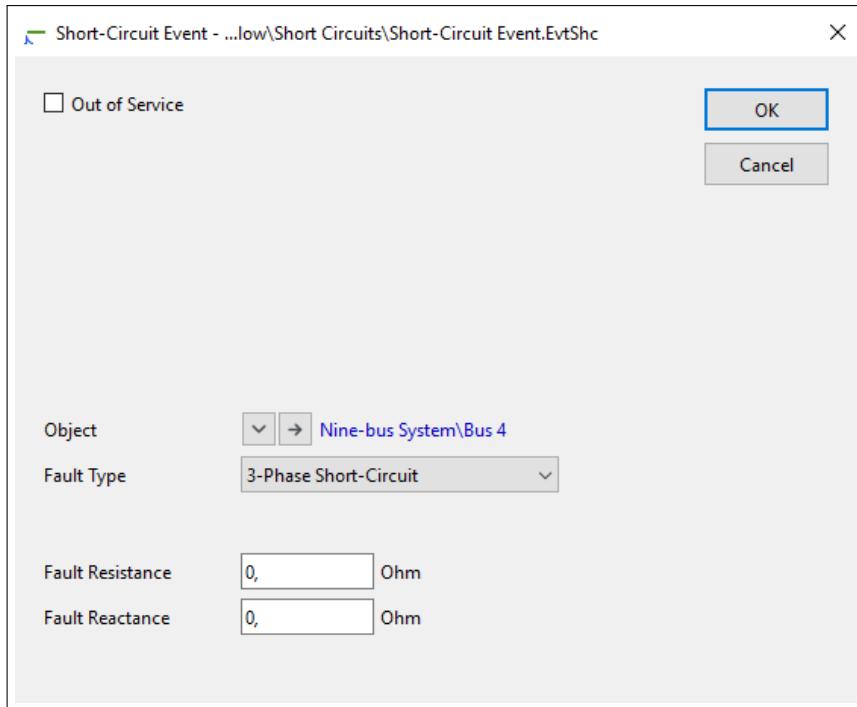


Figure 26.3.2: Accessing the Short-Circuit Events List

- 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.
- The user could add more fault locations to the “Short-Circuit Event List” (*IntEvtshc*) object by right clicking on addition elements in the single line diagram *Add to.. → Multiple Faults*.

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 *Calculate* → *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 *PowerFactory*'s short-circuit calculation command. Some of these options are dependent upon the selected calculation method, therefore separate sections dedicated to each method are presented.

### 26.4.1 Basic Options (All Methods)

The options presented in this section are common to all implemented calculation methods and are used to define the general settings of the short-circuit calculation. The specific options for each method are presented below in separate sections.

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* [18] (the German VDE standard);
- *IEC 60909* [18] (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.

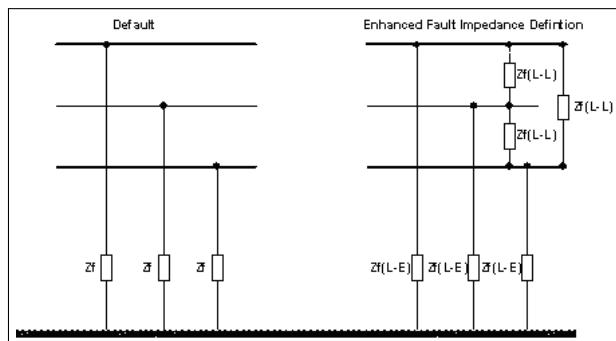


Figure 26.4.1: Fault Impedance Definition: 3-Phase Short-Circuit

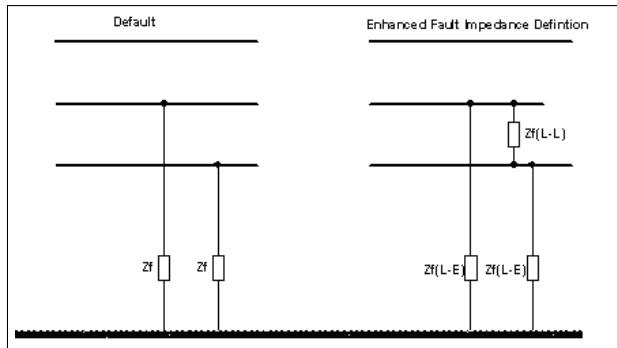


Figure 26.4.2: Fault Impedance Definition: 2-Phase to Ground Fault

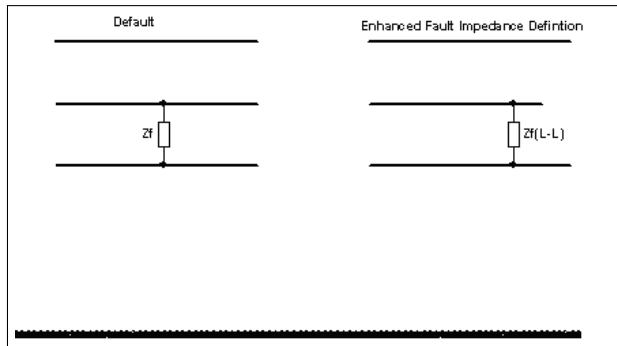


Figure 26.4.3: Fault Impedance Definition: 2-Phase Fault

## Show Output

Tabular result reports can be generated if the *Show Output* option of the dialog is enabled. The command which generates this report is displayed in blue text next to the *Command* button. The user can click on this button to select which type of report will be printed out. Just below the *Command* button, blue text informs the user of the currently-selected report type. As an alternative to a tabular report the user can also select a textual (ASCII) report which is written to *PowerFactory*'s output window when the short circuit command is executed.

## 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).

---

### 26.4.2 Verification (Except for IEC 61363, IEC 61660 and ANSI/IEEE 946)

When enabled (*Verification* page), the user can enter thresholds for peak, interrupting and thermal maximum loading. The *Verification* option will then write a loading report to the output window with all devices that have higher loadings than the defined max. values. This report shows the various maximum 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.

### 26.4.3 Basic Options (IEC 60909/VDE 0102 Method)

In general, please note that the calculation according to IEC 60909 [18] 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 *ElmPvsys*) and asynchronous generators (*ElmAsm*) will all be supported in the short-circuit calculation.

#### Calculate

The drop-down list offers the choice between the minimal or maximal 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 26.4.7. 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.

---

#### 26.4.4 Advanced Options (IEC 60909/VDE 0102 Method)

Generally, the *Advanced Options* page is used for settings to tune the various short-circuit calculation methods. Familiarisation with the IEC/VDE standard before modifying these options is strongly recommended.

##### 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.

##### 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 IEC60909/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.

##### 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.

##### 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.

##### Decaying Aperiodic Component

Allows for the calculation of the DC current component, for which the decay time must be given. According to the IEC/IEC 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:

$$i_{DC} = \sqrt{2} \cdot I_k'' \cdot e^{-mega \cdot T_b \cdot \frac{R}{x}} \quad (26.4)$$

**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.

$$I_{DC} = \sqrt{2} \cdot I_k'' \cdot e^{-\omega \cdot T_b \cdot \frac{R_f}{x_f}} \quad (26.5)$$

$$\frac{R_f}{X_f} = \frac{R_c}{X_c} \cdot \frac{f_c}{f_{nom}} \quad (26.6)$$

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.

### 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.

### Calculate Ik

The steady-state short-circuit currents can be calculated using different means to consider asynchronous machines:

**Without Motors** Asynchronous motors will not be considered in the calculation of the current  $I_k$ . The motors will be treated as disconnected.

$f_n * T_b$	< 1	< 2.5	< 5	< 12.5
$f_c/f_n$	0.27	0.15	0.092	0.055

Table 26.4.1: Breaking Times

**DIGSILENT Method** Considers all asynchronous motors according to their breaking current. The breaker opens after the maximum possible time.

**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 fault itself.

### Consider Protection Devices

This option will calculate measured currents for all protection devices and will evaluate tripping times. To increase the speed of the calculation, this option can be disabled 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.

---

### Automatic 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 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.

## 26.4.5 Basic Options (ANSI C37 Method)

### 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 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](#).

---

### 26.4.6 Advanced Options (ANSI C37 Method)

#### 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
- Low-Voltage Current

#### 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 C31.010 requires that the X/R ratio be determined by a separate R network reduction.

### Consider Protection Devices

This option will calculate measured currents for the selected protection devices and will evaluate tripping times. To increase the speed of the calculation, this option can be disabled 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 in the corresponding paragraph from the IEC60909/VDE section (section [26.4.4](#)).

## 26.4.7 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 [26.4.8](#)) 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.

---

## 26.4.8 Advanced Options (Complete Method)

### 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 *wordBasic 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.

### 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)..

### Current Iteration

If the *Current Iteration* function is enabled, the reactive current contribution of static generator, PWM, Doubly-Fed Induction Machine or Asynchronous Machine (set as Double Fed Induction Machine) is calculated using the values of *K factor* and *Max. Current*. These two parameters are set inside the *Complete Short-Circuit* page of such elements if *Dynamic Voltage Support* has been selected as *Short-Circuit Model*. More information on this is available in the Technical Reference for the elements of interest.

This iterative method is based on a fast current iteration, which typically requires 5 to 10 iterations.

### 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 Protection Devices

This option will calculate measured currents for all protection devices and will evaluate tripping times.

This option can be disabled to increase the calculation speed when protection devices do not need to be analysed.

#### 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.

#### 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 in the corresponding paragraph from the IEC60909/VDE section (section [26.4.4](#)).

#### 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.

---

**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.

---

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.

### 26.4.9 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.

#### 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.

### 26.4.10 Advanced Options (IEC 61363)

The settings available on the advanced options page of the IEC 61363 dialog will depend on the selected calculation method.

#### 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.

### 26.4.11 Basic Options (IEC 61660 Method)

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. For the maximum short-circuit case the resistance of joints (in busbars and terminations) is ignored, the conductor resistance is 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. For the minimum short-circuit case the conductor resistance is referred to the maximum operating temperature, resistance of joints is considered, the contribution of rectifiers is set at the rated short-circuit current, batteries operation is set at the minimum voltage and 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](#).

---

### 26.4.12 Advanced Options (IEC 61660 Method)

Generally, the *Advanced Options* page contains settings applicable to the IEC 61660 calculation method. Familiarisation with the IEC 61660 standard before modifying these options is strongly recommended.

#### Initialisation

This option is provided by *PowerFactory* to give the user more flexibility and represent an enhancement of the calculation standard. The pre-fault voltage can be calculated via a load flow by checking the *Load Flow Initialisation* check-box. In this case *PowerFactory* will run a load flow calculation to estimate the pre-fault voltage. Alternatively to this user defined pre-fault voltage can be set using the *Pre-fault voltage factor*.

#### 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

This option is only considered for minimum short circuit calculations. That is why it will only appear if the minimum fault current calculation is selected on the Basic Option page. If the *Local value* option is selected the joint resistance is calculated geometrically for each line according to input parameters entered in the line element and line type and according to the equation given in the standard. If the *Global value* option is chosen, the resistance of each joint is given the value specified in the command. 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 local value calculation is conducted.

### 26.4.13 Basic Options (ANSI/IEEE 946 Method)

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 [5] 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. For the maximum short-circuit case the following assumptions are applied:

- the resistance of joints (in busbars and terminations) is ignored.
- the conductor resistance is referred to  $20^{\circ}C$ .
- rectifier current limiting controls are disabled.
- decoupling diodes are considered as either blocked (infinite impedance) or conducting (zero impedance) depending on a setting in the individual model.
- batteries are assumed to be fully charged (maximum voltage).

For the minimum short-circuit case:

- the conductor resistance is referred to the maximum operating temperature.
- the resistance of joints is considered.
- the contribution of rectifiers is set by the rated short-circuit current.
- battery operation is set to the minimum voltage.
- 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](#).

---

## 26.4.14 Advanced Options (ANSI/IEEE 946 Method)

Generally, the *Advanced Options* page contains settings applicable to the ANSI/IEEE 946 calculation method. Familiarisation with the IEEE 946 standard before modifying these options is strongly recommended.

### Initialisation

The pre-fault voltage can be set via a load flow via checking the *Load Flow Initialisation* check-box. Alternatively the pre-fault voltage is determined by the nominal bus voltage. The *Joint Resistance of Busbars* input can be used to specify a user-defined joint resistance for bus terminations.

### Using...

The calculation methodology can be defined with either the *Approximation Calculation* selection, which performs the DC short circuit calculation according to the IEEE 946 standard or with the *Superposition Calculation* selection, which performs a superposition calculation. In the case of the *Superposition Calculation*, each DC source is treated individually and the contribution to the fault current is calculated with other sources open circuit, the network is then represented by an equivalent resistance in series with the source resistance. The *Superposition Calculation* method then determines the total fault current using the superposition principle similar to the Complete Method ([26.2.3](#)).

## 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. Also available are several diagram colouring options 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 [9](#): Network Graphics, section [9.5](#). 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 (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.

### 26.5.3 Predefined Report Formats (ASCII Reports)

In *PowerFactory* there are predefined report formats also called ASCII reports, available to the user. These ASCII reports can be created by pressing the *Output Calculation Analysis* button (document icon) located on the main menu (a load flow must be calculated first). Output reports are described in Chapter 19: Reporting and Visualising Results, section 19.4.

A *Show Output and Verification Report* can be also printed out automatically each time a short-circuit calculation is executed (see Section 26.4.1 and 26.4.2).

### 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. 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 a 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 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  $I_r = \frac{U_r}{\sqrt{3}|R_e+jX_e|}$ .

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.

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 *Calculate* → *Contingency Analysis...*

Contingency cases can be created using references to user defined *Fault Cases* and *Fault Groups* (introduced in Chapter [14](#): Project Library, Section [14.3.4](#)) 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 *Calculate* → *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 timesweeps (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 vault (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 11: Building Networks, Section 11.2.7 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.

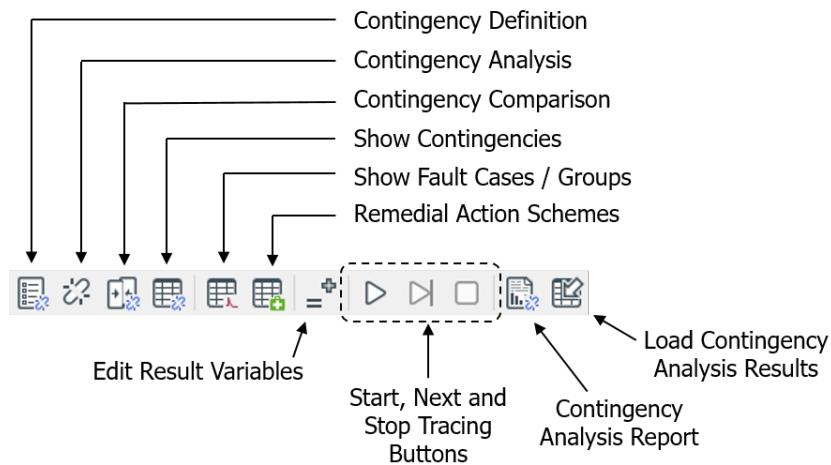


Figure 27.3.1: Contingency Analysis Toolbar Functions

### 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 calculations. 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.

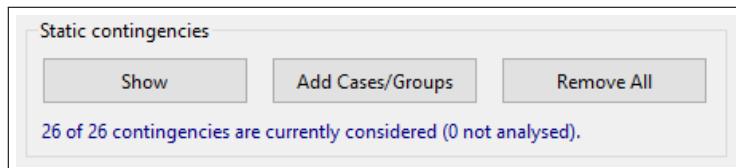


Figure 27.4.1: *Static contingencies* Section of Contingency Analysis Dialog

- **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 sec-

tion [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. 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. For this reason, the third button is provided:
- 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 voltages below (p.u.)** Voltages lower than this value will be recorded in the results file for the corresponding terminal.
  - **Record voltages above (p.u.)** Voltages higher than this value will be recorded in the results file for the corresponding terminal.
  - **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.

#### 27.4.2.3 Restricted Recording of Contingency Results

- **Do not record if base case is above** If in the pre-fault load flow elements have loadings above this value, then they are not recorded in the results.

### 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.

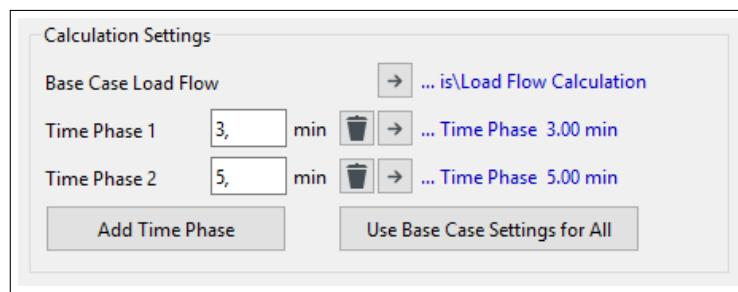


Figure 27.4.2: Different Settings for Base Case and Contingency Load Flows

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 14.3.11) 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  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.

##### 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 power flow on a given circuit. 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)

---

To add study times to the list, first enable the *Calculate Time Sweep* option, then right-click anywhere in the table and select *Insert Rows* (alternatively select *Append Rows* or *Append n Rows*). To modify the

date and time, double-click on the corresponding *Study Time* cell. Additionally, the user has the option to ignore previously defined *Study Times* by enabling the *Ignore* flag. This ensures that the contingency analysis will not take into account the ignored *Study Times* in the calculation.

## 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 11, Section 11.2.7.4.

### 27.4.6.2 Advanced tab

Three 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.

### 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 22.4. 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 sensitive 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 sensitive 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 (☒, see Section 27.7.1). 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 (☒), 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.

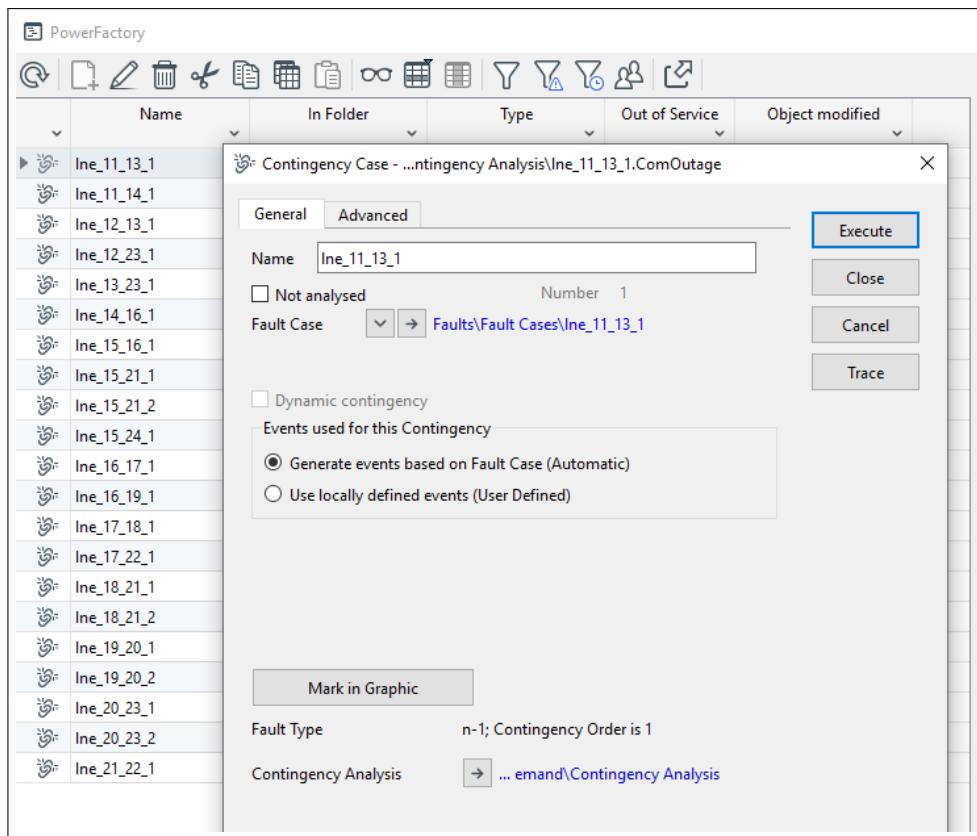


Figure 27.4.3: Contingency Cases (*ComOutage* objects)

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:

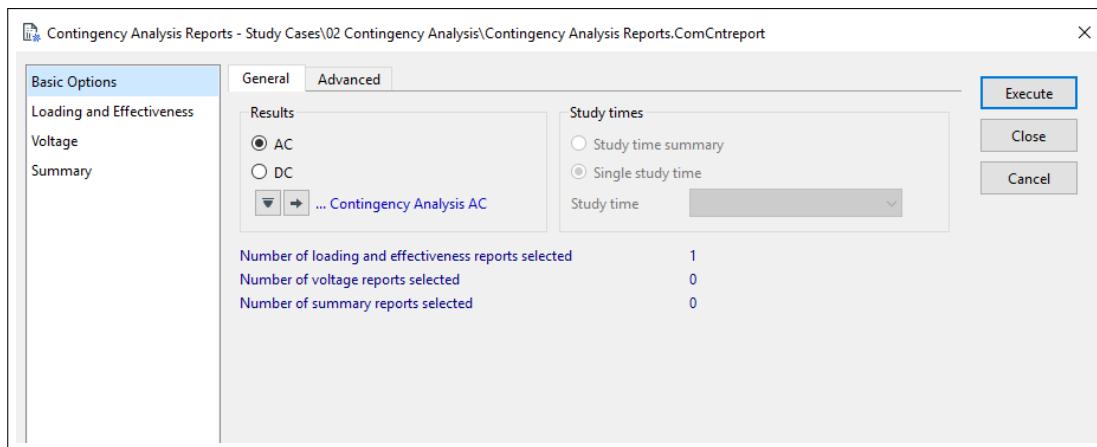


Figure 27.5.1: Contingency Analysis Reports dialog

### 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.

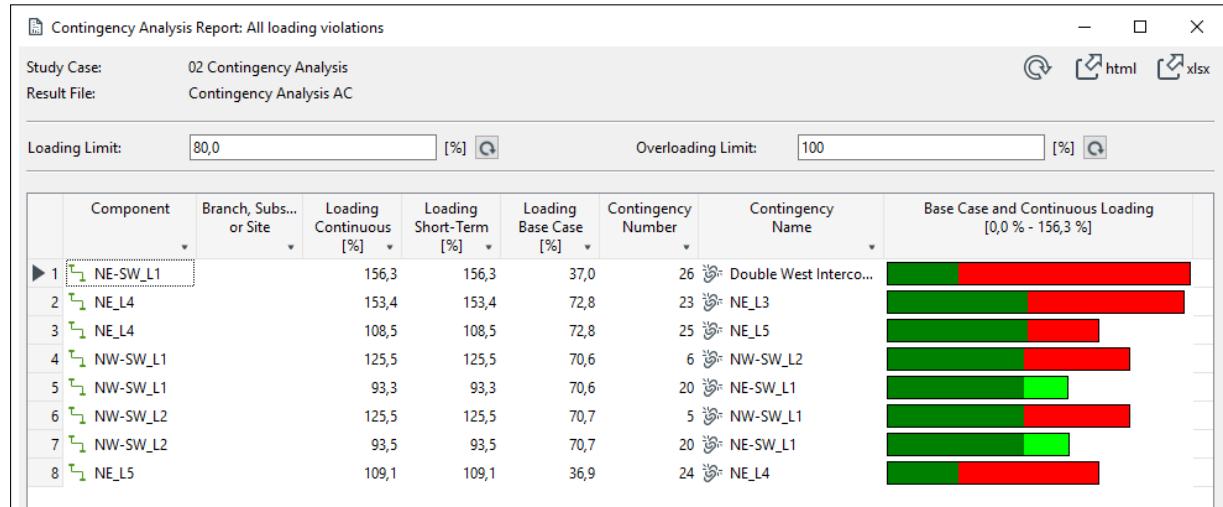


Figure 27.5.2: Tabular Report of Loading Violations

## Effectiveness Reports

- **Generator effectiveness:** generators having an effectiveness greater than or equal to the specified value (%/MW) are displayed in a single list, with an indication of whether generation increase or decrease reduces the overload.
- **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 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).

#### 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.
- **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.

##### 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.

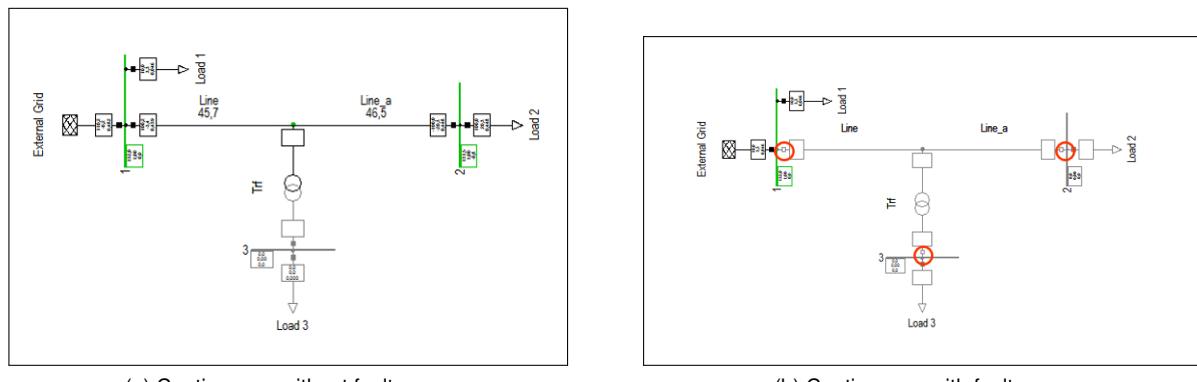


Figure 27.7.1: Use of fault cases in contingency analysis

### 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 Contingency Cases for Analysis.** Generates contingencies which are stored in the contingency analysis command.

### 27.7.1.2 Outage Level

- **n-1.** Creates single contingency cases for each of the selected components.
- **n-2.** Creates contingency cases for every unique combination of two selected components.
- **n-k cases of mutually coupled lines/cables.** Creates contingency 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.** Contingency cases according to the selected outage level will be generated for all lines and cables (*ElmLne* objects) in the system.
  - **Transformers.** Contingency cases according to the selected outage level will be generated for all transformers (*ElmTr2*, *ElmTr3*, *ElmTr4* objects) in the system.
  - **Generators.** Contingency cases according to the selected outage level will be generated for all synchronous generators (*ElmSym* objects) in the system.
  - **Series Capacitors.** Contingency cases according to the selected outage level will be generated for all series capacitors (*ElmScap* objects) in the system.
  - **Series Reactors.** Contingency 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, 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.

## 27.7.2 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, *Calculate* → *Contingency Analysis* **or**
3. Select one or more objects and right-click, *Calculate* → *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, *Calculate* → *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, *Calculate* → *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. 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). 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 timesweeps (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: Project Library, Section [14.3.4](#): 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).

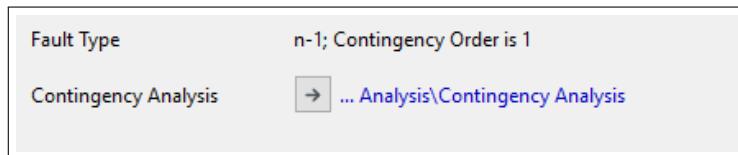


Figure 27.8.1: *Fault Type* Field in the Contingency Case (*ComOutage*) Dialog

**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. In the drop-down list, 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: *Define... → Fault Case → Single Fault Case* or *Define... → Fault Case → Multiple Fault Cases, n-1 (or Multiple Fault Cases, n-2)* or *Define... → Fault Case → 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 *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) *Define... → Fault Case....* 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.

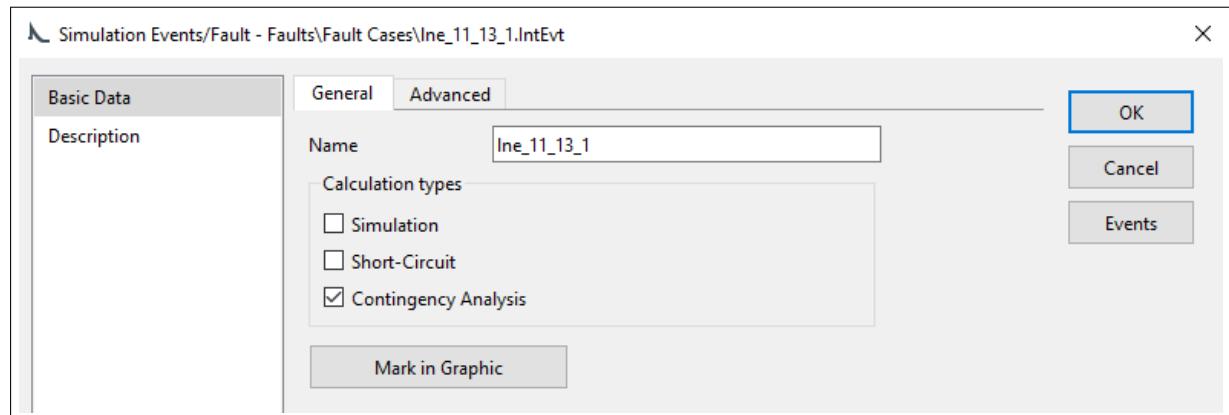


Figure 27.8.2: Creation of Fault Case (*IntEvt*)

For further background on fault cases, refer to Chapter 14: Project Library, Section 14.3.4 (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 create contingencies was described. In a similar way, it can be used to create fault cases for the library. The same principles apply, but the user should select:

- **Generate Fault Cases for Library.** With this option selected, when Execute is pressed, the requested fault cases will be created in the Fault Cases folder (Operational Library, Faults).

### 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.

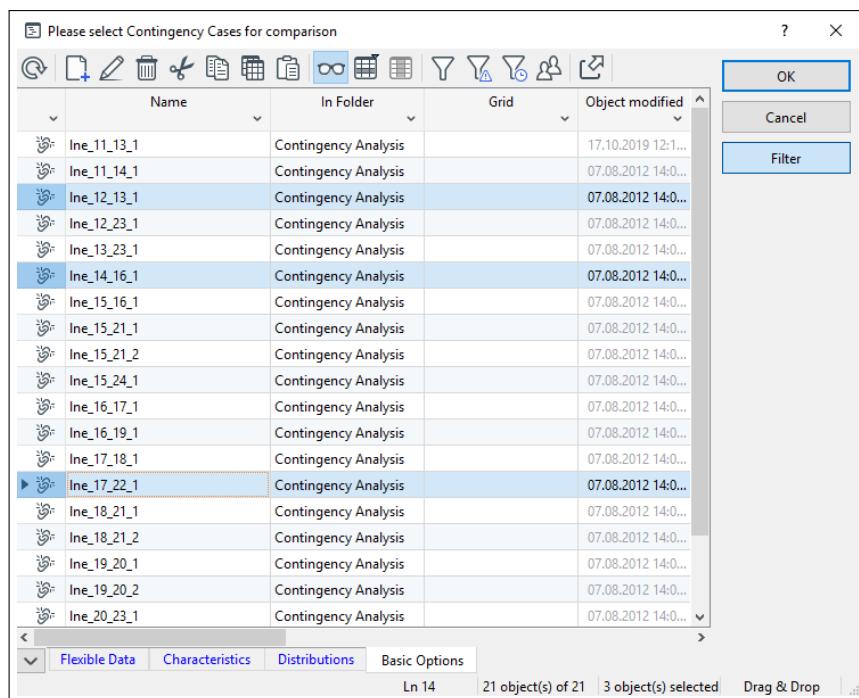


Figure 27.9.1: Selection of Contingency Cases for Comparison

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.

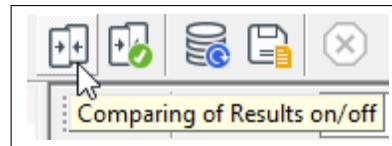


Figure 27.9.2: Comparing of Results Button

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).

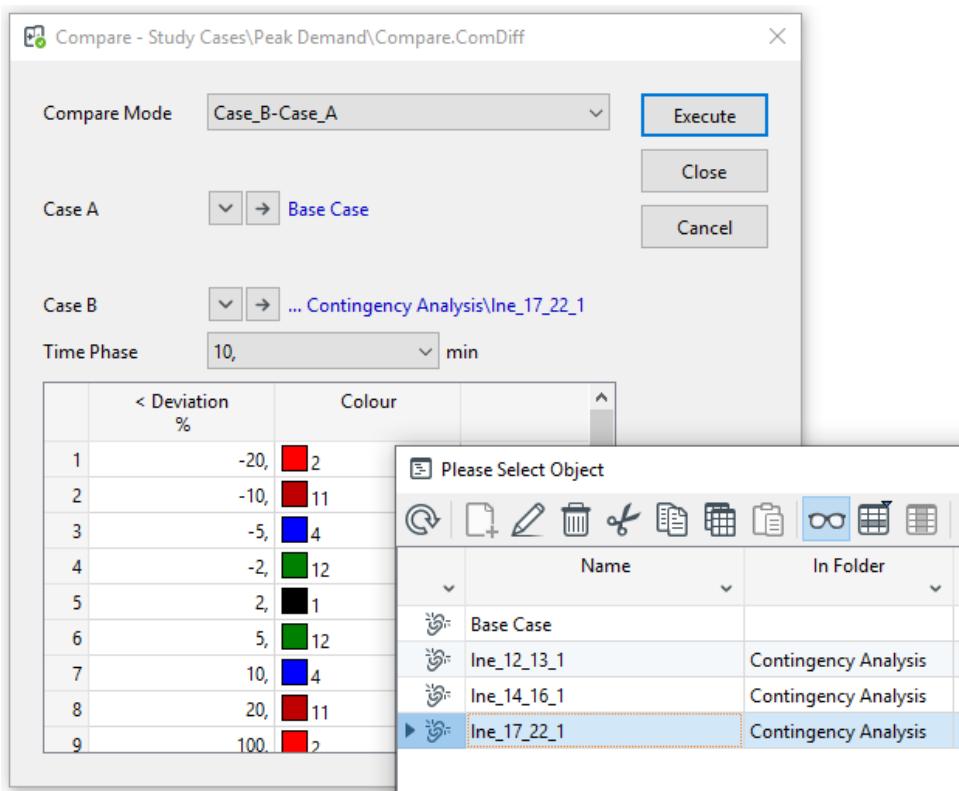


Figure 27.9.3: Selection of other Cases for Comparison

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 *IntMon* object, but it is possible to add additional variables to these *IntMon* objects if required.

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 drop-down, 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 (Int Ras). 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
- 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.

## 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.

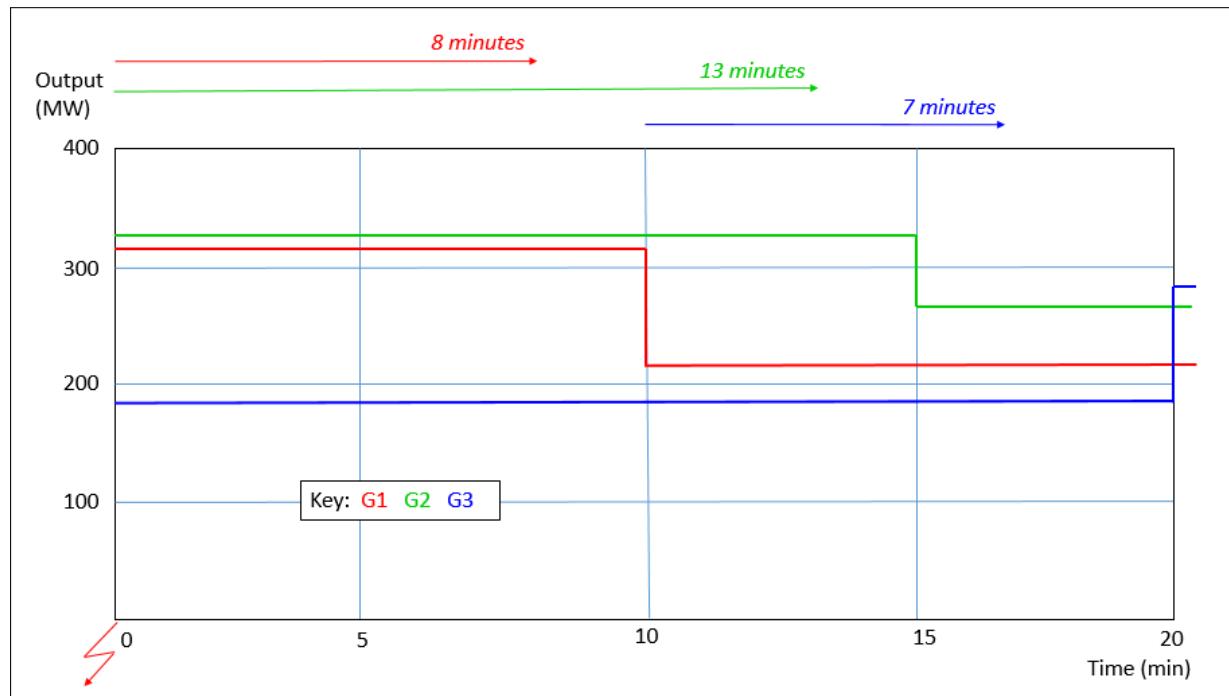


Figure 27.11.1: RAS timings in Multiple Time Phase Contingency Analysis

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.

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.

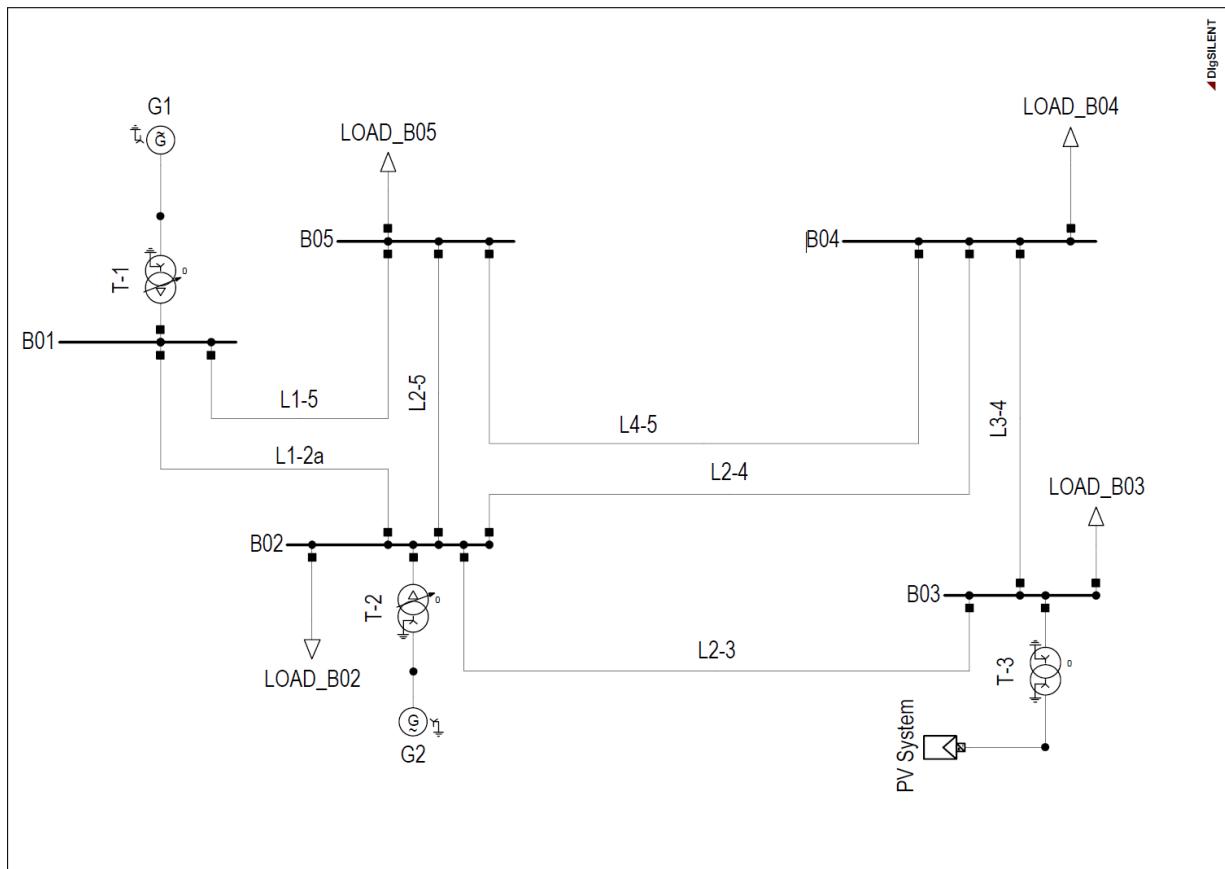


Figure 28.2.1: Single line diagram of example network

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.

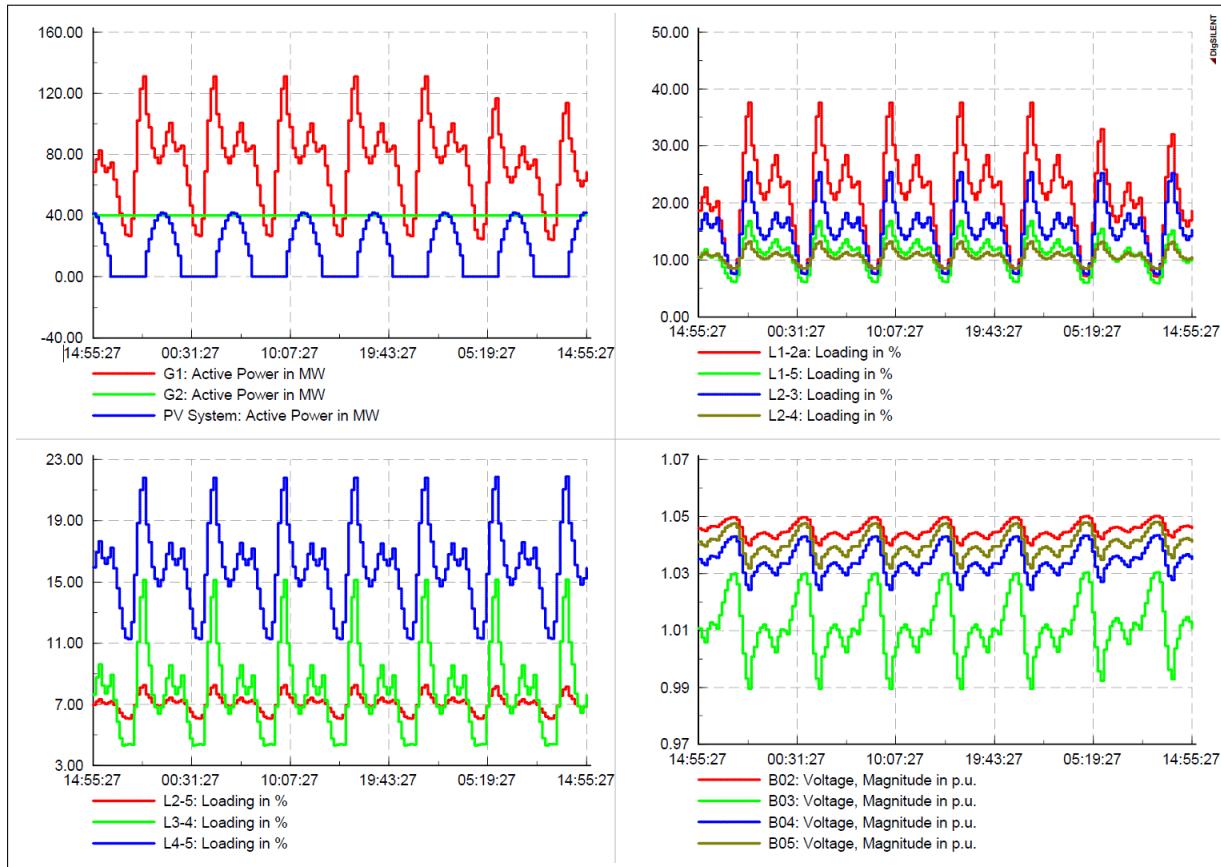


Figure 28.2.2: An example of weekly power flows in the example system calculated using the Quasi-Dynamic simulation tool

## 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.
5. Optional: Modify the default variables:
  - (a) Double-click the icon for the object. The variable browser dialog will appear.
  - (b) Navigate to the appropriate page and select variables from the desired variable set.
  - (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. Navigate to the desired page and select variables from the desired variable set.
    - 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.
    - 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. Navigate to the desired page, and select variables from the desired variable set.
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 [14.3.7](#).

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 those outages that occur during the defined simulation period on the basic data page are shown in this list.
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* Simulation 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 pops up. 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 (28.5.4). 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.
- **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 settings used by the *Quasi-Dynamic Simulation*.
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. *Basic Options* → *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 (*ElmQds*).
  - *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 [28.5.3](#).

---

**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.
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 [22.4](#).

---

**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 [24.12](#).

## 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 19.7 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 =  $\sigma^2$

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 simulation 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 Simulation model definition** (*TypQdsI*, library type object). Represents the type definition of a Quasi-Dynamic simulation model. It is referred throughout this Section as *QDSL type*;
- **Quasi-Dynamic Simulation 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 Simulation model definition *TypQdsI*

A new *QDSL type* object *TypQdsI* may be created in the project library (default location is *Library* → *User Defined 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* → *User Defined Models*) folder in the active project. Use the *New Object*  toolbar icon and select the object *Quasi-Dynamic Simulation Type*;
- Alternatively, using the Data Manager, right-click inside the *Library* → *User Defined Models*) folder in the active project and select *New... → Quasi-Dynamic Simulation Type* from the context-sensitive 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* → *User Defined Models* folder.

The *QDSL type* (*TypQdsI*) dialog is divided in several sections with the following properties:

**Basic Data** page options:

- *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* (*TypQdsI*). 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 *ElmQdsI* 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.

#### Initialisation page options:

- *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 28.5.3).

#### Load Flow page options:

- 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 28.5.3).
- *Inputs/Outputs for load flow equations* - As described in Section 28.5.3, 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:
  - Insert 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 type the class associated to the object (e.g. for a busbar, the class is *ElmTerm*). If in doubt, one can open the dialog of any network element of the same class and, in the dialog header, the path, element name and element class are provided (e.g. for a static generator - “Grid\Static Generator.**ElmGenstat**”).
  - 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 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.
- *Equations* tab/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 28.5.3).
- *Control* tab/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 28.5.3).

**Quasi-Dynamic Simulation** page options:

- *Equations* tab - The *Quasi-Dynamic Equations* 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 28.5.3.
- *Control* tab - 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 28.5.3).

---

**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
- Release Notes

### 28.5.1.2 Creating the Quasi-Dynamic Simulation 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 Simulation Model*;
- Alternatively, from the single line diagram, right-click on any network element that shall be controlled and select *Define* → *Quasi-Dynamic Simulation Model*. 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:

- *Model Definition* - 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:

- *Network elements of inputs/outputs* - Within this table, assignment of the referenced elements is effected for the declared input/output variables within the *QDSL type* object, *Load Flow* page, *Inputs/Outputs for load flow equations* table. Only network elements of the declared class (as defined in the *QDSL type*) can be selected for a specific row.

**Description** page - A standard *Description* page is made available for version control purposes.

## 28.5.2 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 Simulation* model *QDSL model*.

The scripting language for all the model definition (*TypQdsI*) scripts is *DigSILENT 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 28.5.4.

*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.

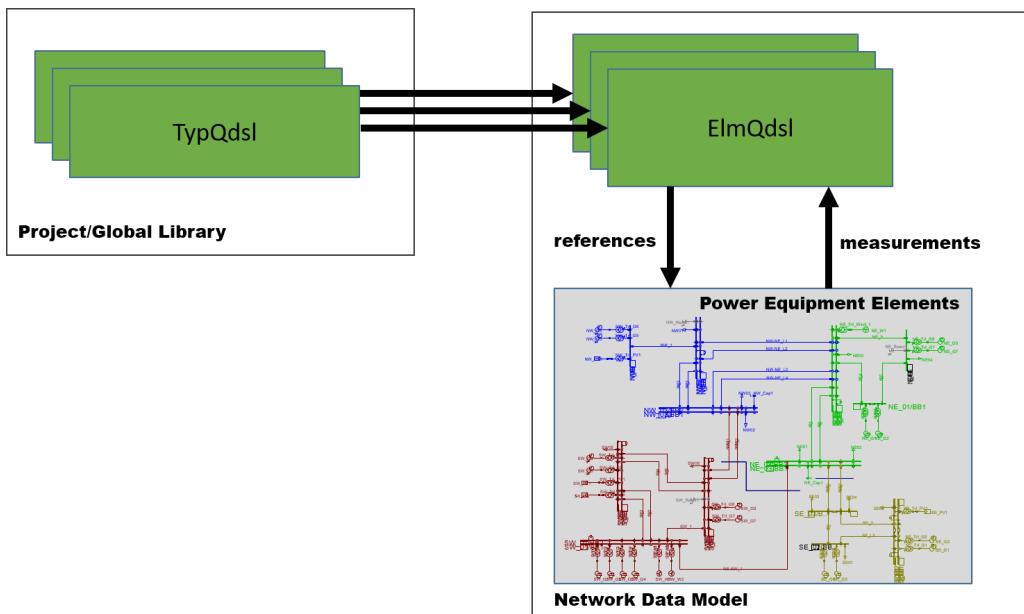


Figure 28.5.1: Integration of *QDSL models* within the *PowerFactory* project

### 28.5.3 Algorithm flow of user defined Quasi-Dynamic Simulation 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:

$$t_{k+1} = t_k + \text{Step}$$

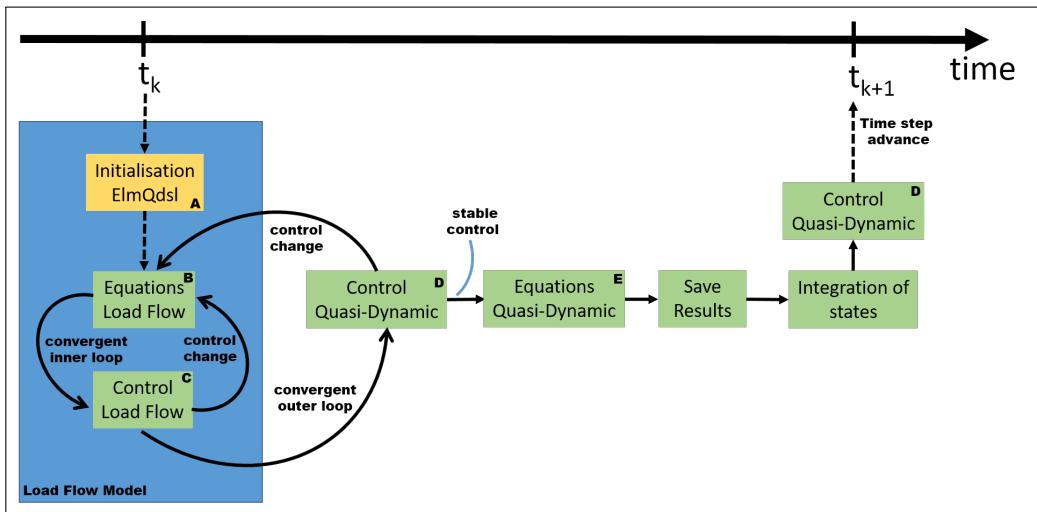


Figure 28.5.2: *QDSL models* - Simulation Procedure

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 Equations block (**B**)

3. Load Flow Control block (**C**)
4. Quasi-Dynamic Simulation Control block (**D**)
5. Quasi-Dynamic Simulation 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 TypQdsl*. 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 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;
- 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:

$$f(P_{set}) = P_{set} - 2 = 0$$

$$\frac{df}{dP_{set}} = 1 = \text{constant} \implies \text{model is linear, in the only output } P_{set}$$

Assuming the load flow equation function of a certain model is defined by equation below (where  $P_{ini}$  is a parameter):

$$f(P_{set}, u_i, u_r) = P_{set} - (u_r^2 + u_i^2) \cdot P_{ini} = 0$$

Calculating the first order derivative of the function with respect to its inputs:

$$\begin{aligned}\frac{df}{dP_{set}} &= 1 = \text{constant} \implies \text{model is linear in } P_{set} \\ \frac{df}{du_i} &= -2 \cdot P_{ini} \cdot u_i = \text{not constant} \implies \text{model is non-linear in } u_i \\ \frac{df}{du_r} &= -2 \cdot P_{ini} \cdot u_r = \text{not constant} \implies \text{model is non-linear in } u_r\end{aligned}$$

#### C. - Load Flow **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 **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;
- 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 **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;

#### 28.5.4 Scripting Functions for Quasi-Dynamic Simulation

function	description
<a href="#">CreateEvent</a>	Creates a simulation event with a certain execution time
<a href="#">CreateMultiLoadEvent</a>	Creates a simulation load event defined for multiple loads
<a href="#">GetSimulationTime</a>	Retrieves the current simulation time
<a href="#">GetStepSizeSeconds</a>	Returns simulation step size in seconds
<a href="#">SetEventParameter</a>	Function used to set a certain event parameter
<a href="#">SetEquation</a>	Defines the load flow equation of one load flow output variable
<a href="#">GetEquationMismatch</a>	Retrieves the current value of the SetEquation of a certain index
<a href="#">SetControlLoopFinished</a>	Enables the user to govern the outer loop behaviour of the <i>Control</i> scripts
<a href="#">IsOutaged()</a>	Checks if element is outaged

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:

```

1  double time;
2  object event, gen1;
3  set setGens;
4
5  time = GetSimulationTime();
6  event = CreateEvent('EvtGen', time+1);
7
8  setGens = GetCalcRelevantObjects('ElmSym');
9  gen1 = setGens.First();
10
11 if ({event<>NULL}.and.{gen1<>NULL}) {
12     SetEventParameter(event, 'loc_name', 'GenEvtQDSL');
13     SetEventParameter(event, 'p_target', gen1);
14     SetEventParameter(event, 'dP', 1.0);
15     SetEventParameter(event, 'dQ', 2.0);
16 }
17

```

### **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:

```

1  object load,
2  event;
3  set setLoads,
4  allLoads;
5
6  allLoads = GetCalcRelevantObjects('ElmLod');
7  load = allLoads.First();
8  setLoads.Add(load);
9  load = allLoads.Next();
10 setLoads.Add(load);
11 event = CreateMultiLoadEvent(setLoads);
12

```

```

13 if (event<>NULL) {
14     SetEventParameter(event, 'loc_name', 'MultiLoadEvent1');
15     SetEventParameter(event, 'iopt_type', 0);
16     SetEventParameter(event, 'dp', 1.0);
17     SetEventParameter(event, 'dq', 2.0);
18 }
19

```

**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.

**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*.

```
1 SetEquation(0, in-out);
```

The variable *in* represents in the example above an *Output* signal as declared within the *Load Flow* page 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<sub>rated</sub>* is a parameter:

```
1 double out;
2 out = Prated * u ;
3 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:

```

1 double mismatch;
2
3 mismatch = GetEquationMismatch(0);
4 printf('The error in equation 0 is %f', mismatch);
5

```

**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):

```

1 if (gen:m:P:bus1 > 1.0) {
2     gen:s:pset = 0.5;
3     SetControlLoopFinished(0);
4 }
5 else {
6     SetControlLoopFinished(1);
7 }
8

```

**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.

### 28.5.5 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.

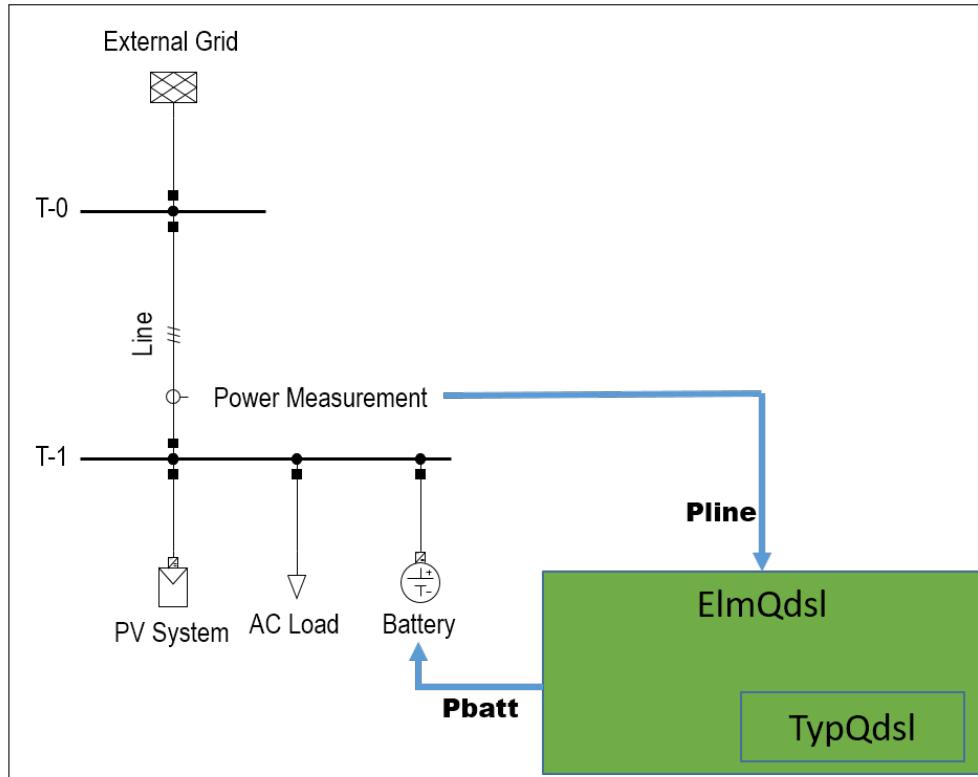


Figure 28.5.3: Example of Battery System considering branch flow measurement

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:

$$chargeP = \begin{cases} 1, & \text{battery charging} \\ 2, & \text{battery inactive} \\ 3, & \text{battery discharging} \end{cases} \quad (28.2)$$

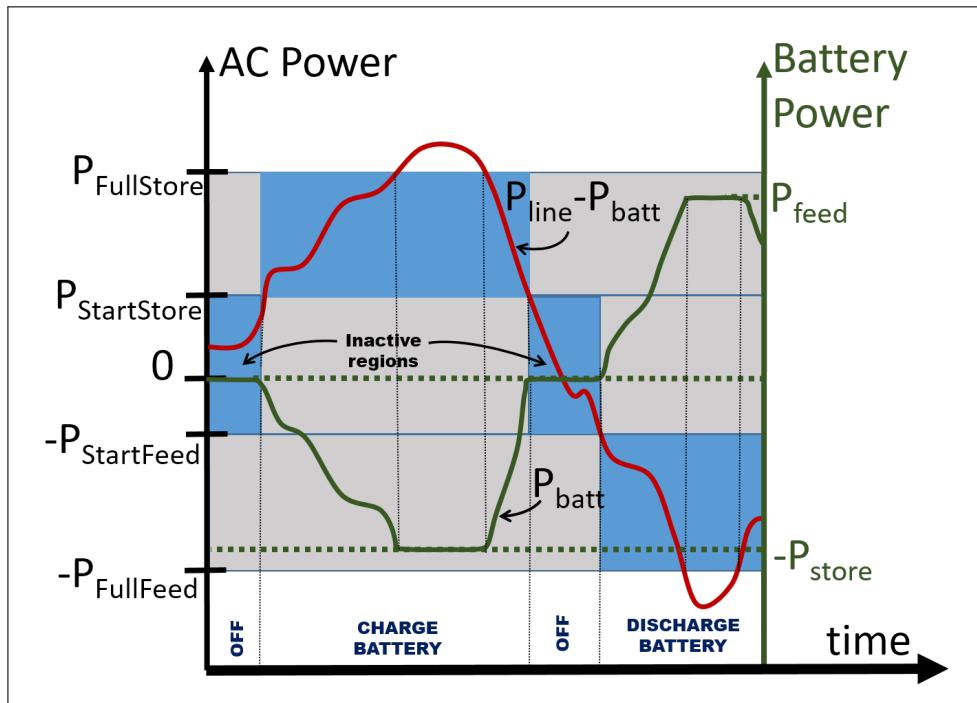


Figure 28.5.4: Example of a generic battery control strategy

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.

state variable	SOC	%	State of charge
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):

Pmeas	MW	Measured power (PV and load)
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:

```

1 double pmeas;
2 SOC = SOCini;
3 pmeas = 0.; ! arbitrary value being provided; load flow has not been yet executed
4 ! measured power operation area
5 chargeP = 0.;
6 if ({PFullStore <= PStartStore}.or.{-PStartFeed <= -PFullFeed}) {
7   chargeP = 0; ! Error
8   Warn('PFullStore must be > than PStartStore and PFullFeed > than PStartFeed');
9 else if (pmeas < PStartFeed) chargeP = 3;
10 else if (pmeas > PStartStore) chargeP = 1;
11 else chargeP = 2;
12 ! energy operation area
13 iniSOCob = 0; ! Inside bounds
14 if (SOCmin >= SOCmax) {
15   chargeE = 0; ! Error
16   Warn('SOCmin must be < than SOCmax.');
17 }
18 else if (SOC > SOCmax) {
19   chargeE = 3;
20   iniSOCob = 1;
21 else if (SOC = SOCmax) chargeE = 3;
22 else if (SOC = SOCmin) chargeE = 1;
23 else if (SOC < SOCmin) {

```

```

24 chargeE = 1;
25 iniSOCob = 1;
26 else chargeE = 2;

```

The load flow outer loop equations ([Load Flow 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:

```

1 Pmeas = Pline*orientation - Pbatt; ! negative=load
2 ! measured power operation area
3 if (chargeP > 0) { ! Not initial error
4   if (Pmeas < -PStartFeed)      chargeP = 3;
5   else if (Pmeas > PStartStore) chargeP = 1;
6   else                           chargeP = 2;
7 }
8 ! energy operation area
9 if (chargeE > 0) { ! Not initial error
10  if (SOC >= SOCmax) {
11    chargeE = 3;
12    if ({iniSOCob = 0}.and.{SOC > SOCmax}) {
13      SOC = SOCmax;
14    }
15  }
16  else if (SOC <= SOCmin) {
17    chargeE = 1;
18    if ({iniSOCob = 0}.and.{SOC < SOCmin}) {
19      SOC = SOCmin;
20    }
21  }
22  else {
23    chargeE = 2;
24    iniSOCob = 0; ! Inside limits now
25  }
26 }

```

The load flow inner loop equations ([Load Flow 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>* ):

$$P_{batt} = \begin{cases} P_{feed} \cdot \left(1 - \frac{P_{meas} + P_{FullFeed}}{-P_{StartFeed} + P_{FullFeed}}\right), & \text{discharging} \\ -P_{store} \cdot \left(1 - \frac{P_{FullStore} - P_{meas}}{P_{FullStore} - P_{StartStore}}\right), & \text{charging} \\ 0, & \text{battery inactive} \end{cases}$$

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 [Load Flow 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  $P_{batt}$  (static generator element in *PowerFactory*) to the calculated setpoint  $P_{gen}$ . Note also the generator sign convention chosen when computing the setpoint  $P_{gen}$  such to comply with the requirements of the *Static Generator* element (positive  $P_{gen}$  means discharging, negative  $P_{gen}$  means charging, as in Figure 28.5.4).

```

1 double Pgen,
2     redFac;
3 Pmeas = Pline*orientation - Pbatt; ! negative=load
4 redFac = 1.0;
5 if ({chargeP = 3}.and.{chargeE >= 2}.and.{chargeE > 0}) {
6     if (Pmeas > -PFullFeed)
7         {redFac = 1 - ((Pmeas + PFullFeed)/(-PStartFeed + PFullFeed));}
8     Pgen = Pfeed * redFac; ! discharge = GEN, feeding
9 }
10 else if ({chargeP = 1}.and.{chargeE <= 2}.and.{chargeE > 0}) {
11     if (Pmeas < PFullStore)
12         {redFac = 1 - ((PFullStore - Pmeas)/(PFullStore - PStartStore));}
13     Pgen = -Pstore * redFac; ! charge = LOAD, storing
14 }
15 else {
16     Pgen = 0. ;
17 }
18 SetEquation(0, Pbatt - Pgen);

```

The [Quasi-Dynamic Simulation 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:

```
1 SOC. = -Pbatt * 100. / (Eini * 3600.); ! slope of charge/discharge in %
```

The [Quasi-Dynamic Simulation 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:

```

1 ! energy operation area
2 if (chargeE > 0) { ! Not initial error
3     if (SOC >= SOCmax) {
4         chargeE = 3;
5         if ({iniSOCob = 0}.and.{SOC > SOCmax}) {
6             SOC = SOCmax;
7         }
8     }
9     else if (SOC <= SOCmin) {
10        chargeE = 1;
11        if ({iniSOCob = 0}.and.{SOC < SOCmin}) {
12            SOC = SOCmin;
13        }
14    }
15    else {
16        chargeE = 2;
17    }
18 }

```

```
17     iniSOCoob = 0; ! Inside limits now
18 }
19 }
```

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.

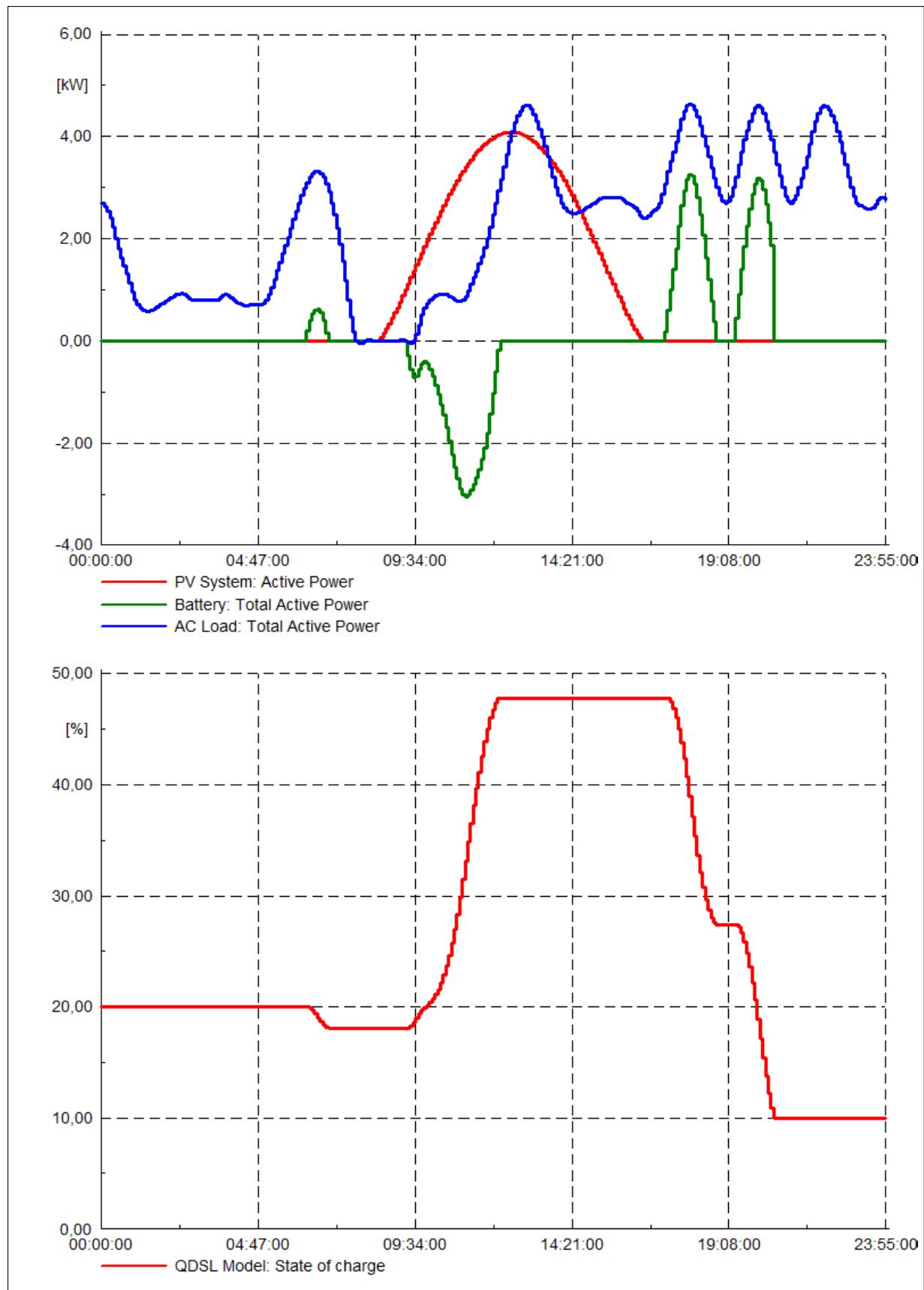


Figure 28.5.5: Power Flow and state of charge SOC behaviour during a one day simulation

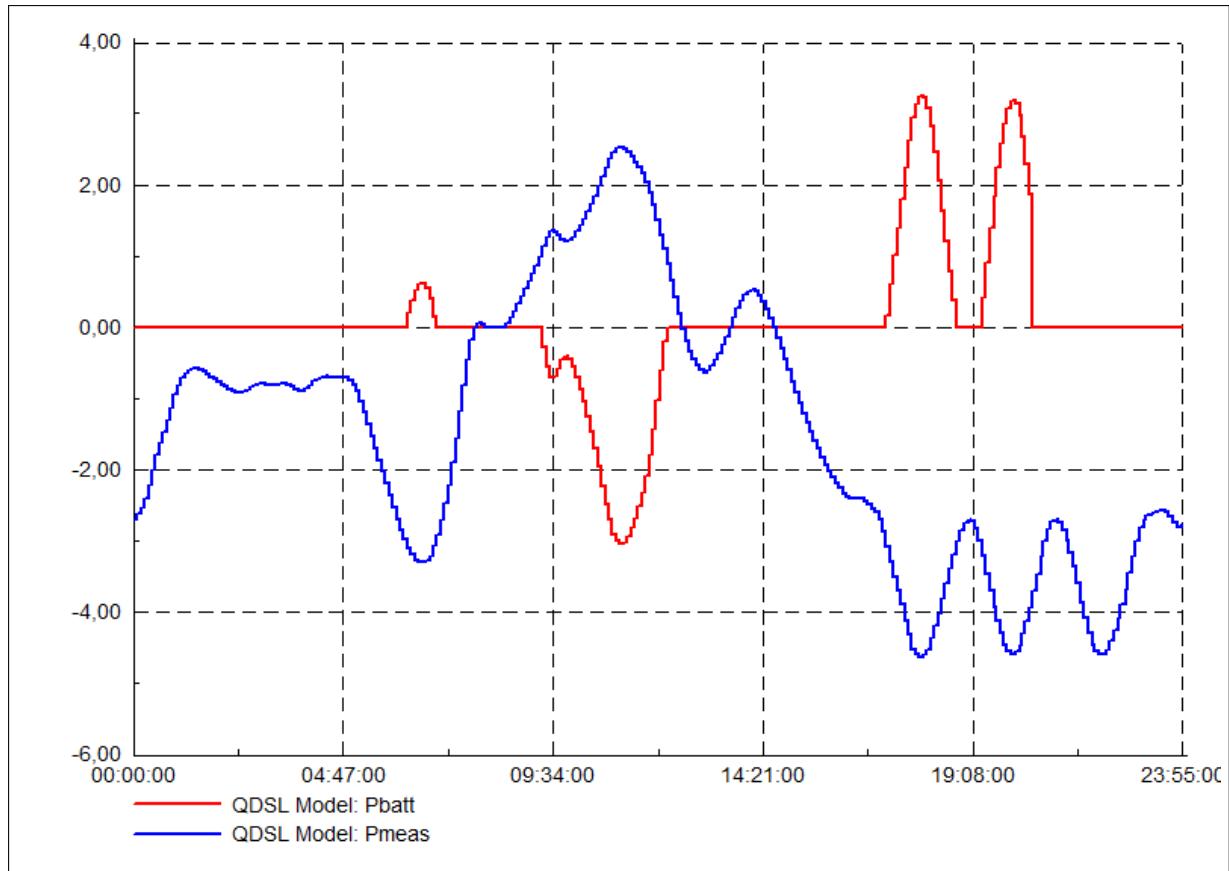


Figure 28.5.6:  $P_{meas}$  - sum of PV and Load power [kW];  $P_{batt}$  - battery power output [kW]

### 28.5.6 QDSL Model Encryption

PowerFactory offers the possibility to encrypt the script code of a QDSL type (Quasi-Dynamic Simulation 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.

# Chapter 29

## RMS/EMT Simulations

### 29.1 Introduction

The RMS/EMT simulation functions available in DIgSILENT *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:

1. Calculation of initial values, including a load flow calculation (refer to Section 29.3)
2. Definition of result variables (refer to Section 29.4)
3. Definition of simulation events (refer to Section 29.5)
4. Execution of simulation (refer to Section 29.6)
5. Creation of simulation plots (refer to Section 29.7)
6. Exporting results (optional, refer to Section 19.6.1).

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 29.8.

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 29.10)
- Co-simulation with external solver (described in Section 29.11)

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 29.12 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 29.13:

- 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 DSL 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 29.4), the event list (described in section 29.5) 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

#### 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 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 *f1rel* (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 *f1rel* reaches the detection angle.
- Out of step is detected when the rotor angle *f1rel* 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

#### Integration step size

When using a fixed step size for the simulation, the integration step size for EMT or RMS has to be set.

- Electromechanical transients (RMS) (typical value: 0.01 s)
- Electromagnetic transients (EMT) (typical value: 0.0001 s)

When using a variable step size, the following values should be entered:

- *Electromagnetic transients / electromechanical transients*: minimum step size for EMT and RMS simulations, respectively.
- *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 nominal power of the machines and the voltage levels. As an adequate starting value, should be set to:  $\text{errsm} = 10 * \text{errlf}$ , 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

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.

### 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.

#### 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 detail in section 29.8. 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 (*ElmNoise*) 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 - 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 [29.9](#).

#### 29.3.6.1 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 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](#).

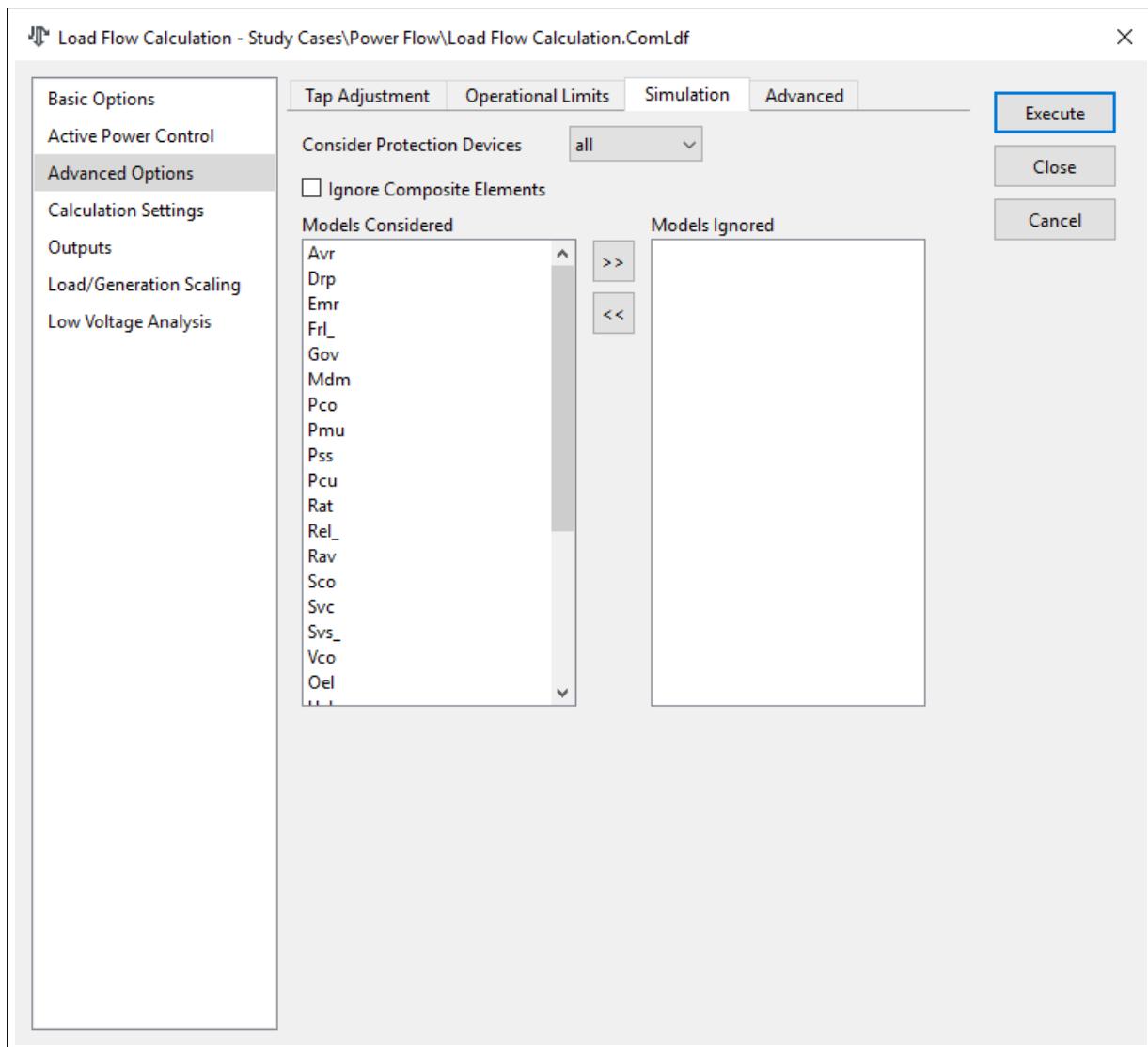


Figure 29.3.1: *Advanced Simulation Options* in the Load Flow command dialog

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 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 29.4.1 for the *Initial Conditions* dialog.

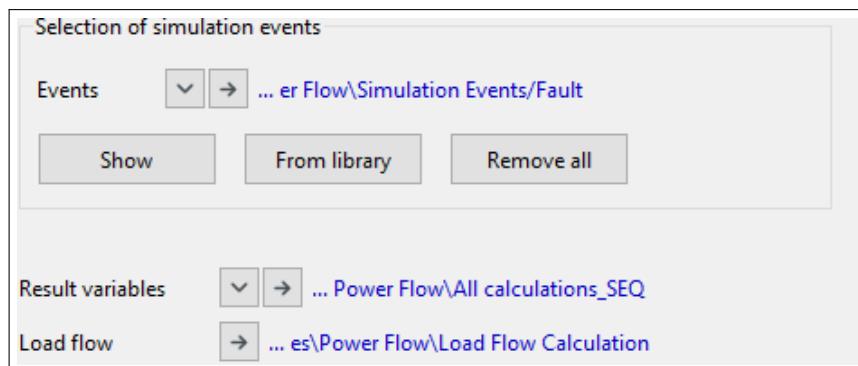


Figure 29.4.1: Results Object Reference

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).

Variables of different elements can be added to the results object for RMS and EMT simulations by right-clicking on the element and selecting *Define* → *Results for Simulation RMS/EMT...*

This element will then be monitored during the simulation. A browser window is automatically opened, and by double-clicking on the variable set icon (grid) of the relevant row, the variables of interest to be recorded can then be selected. See also Section 19.6 (Results Objects).

---

**Note:** Most of the variables for RMS and EMT simulations are identical. Nevertheless there may exist variables that are valid for EMT but not for RMS calculations. It is advisable to only use variables for the calculation which is currently being performed.

---

### 29.4.1 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 19.7 (Plots).

## 29.5 Simulation Events

This section provides a general description of Events, as they apply to time-domain simulations. See Chapter 13: Study Cases, Section 13.9 (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  icon in the toolbar. The event type can be chosen from the list in the element selection dialog which pops up, as shown in Figure 29.5.1. The events can also be modified during a simulation by stopping the calculation, editing the events and continuing the simulation.

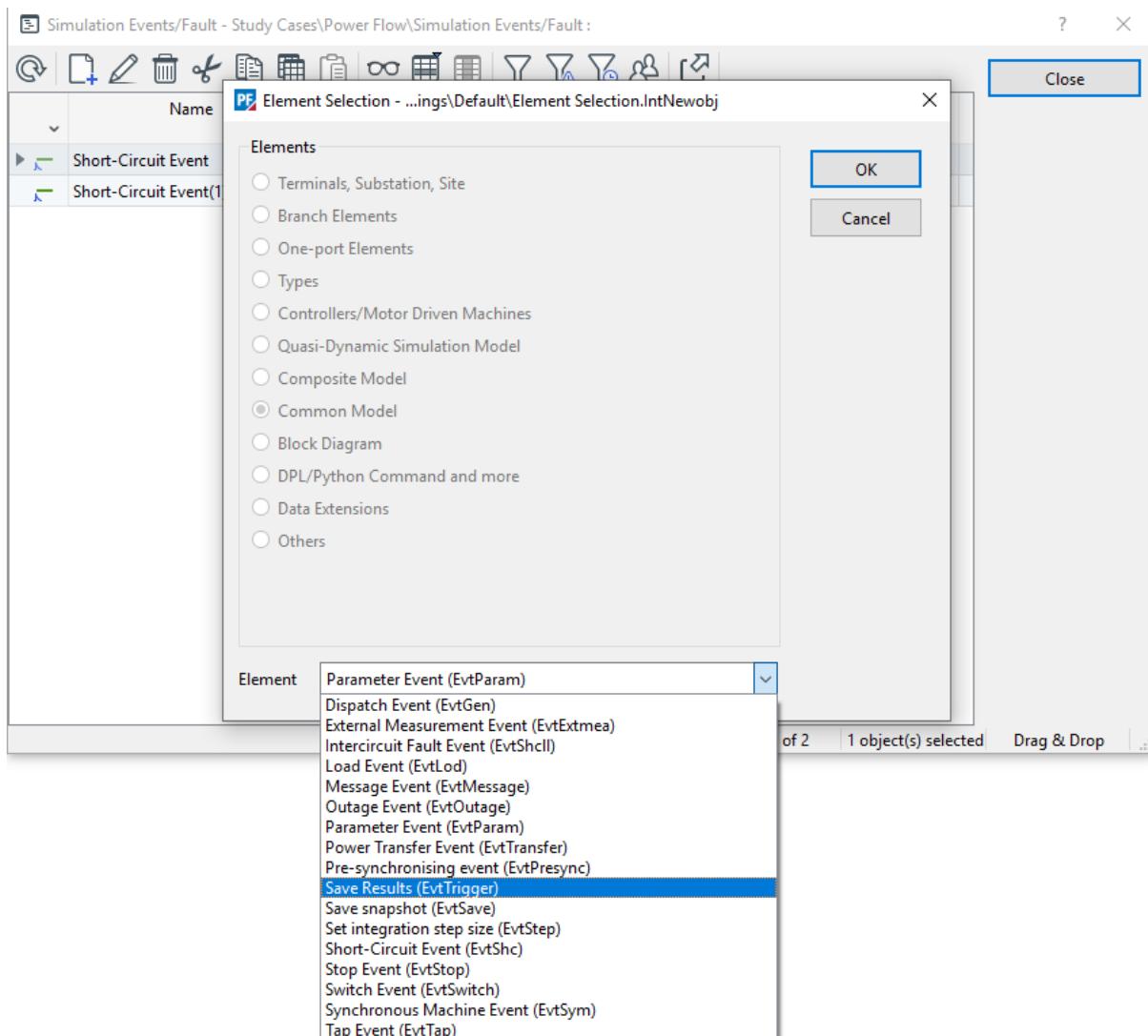


Figure 29.5.1: Defining a New Simulation Event

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 *Define... → Switch Event*, *Define... → Load Event* or *Define... → Short-Circuit 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.

**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 29.5.2.

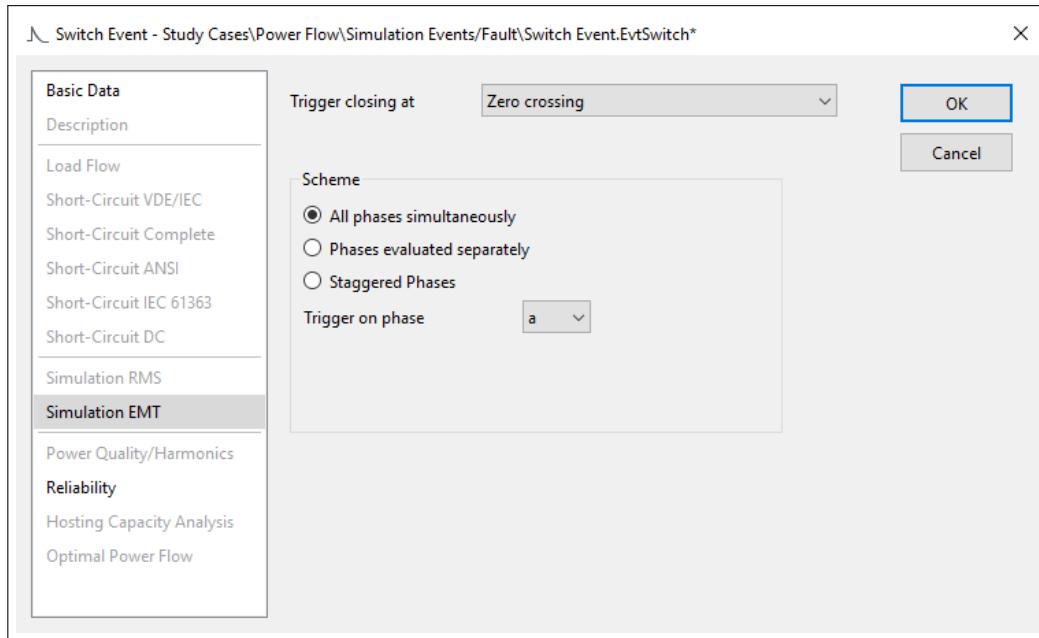


Figure 29.5.2: Additional options for breaker close events

### Load Event used on a selection of load elements

It is possible to apply a single load event (*EvtLod*) to multiple load elements by referencing a selection object (*SetSelect*) as shown in Figure 29.5.3.

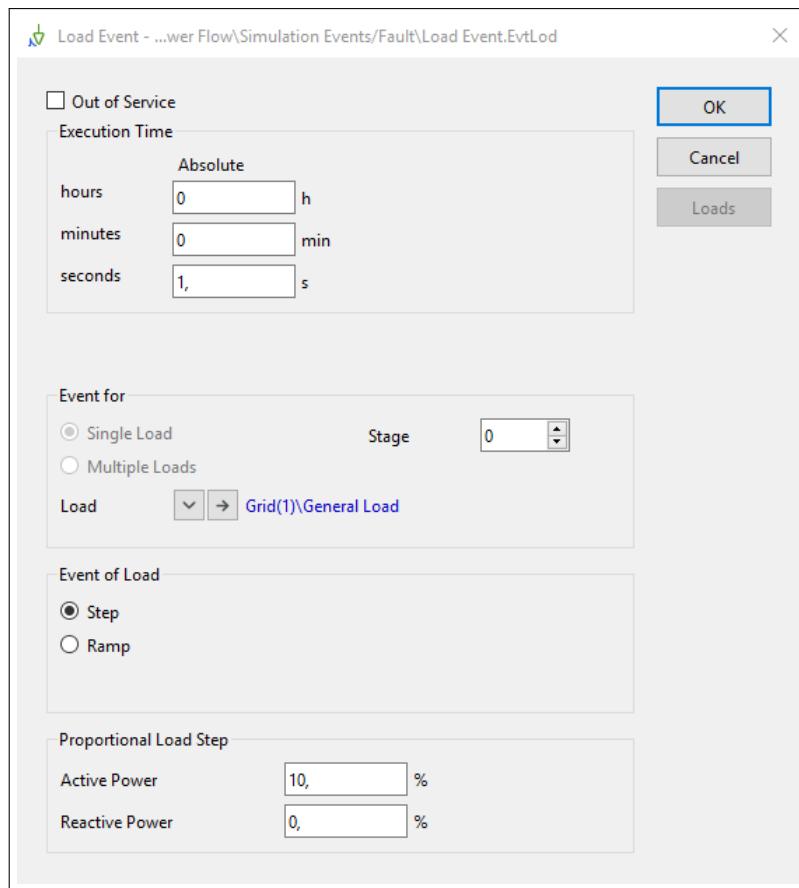


Figure 29.5.3: Referencing multiple loads to a single load event

#### 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*).

## 29.6 Executing the Simulation

Upon successful calculation of the initial conditions (i.e. execution of *ComInc* ), the icon on the Simulation RMS/EMT toolbar will be activated and can be pressed to start the simulation.

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 the icon again, and entering a new stop time. In this case, the stop time may also be entered relative to the current simulation time.

A running simulation may be interrupted by pressing either the icon or the icon on the main toolbar. Additional events can be created and results may be viewed while the simulation is paused. The simulation is then continued by pressing the icon again. Pausing and continuing the simulation may be done as often as required.

The *Run Simulation* has a link to the initial conditions command used and additional options for the displayed messages can be set in the *display in output window* and *internal DSL warnings* fields.

## 29.7 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 19.7 (Plots).

## 29.8 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.

---

### 29.8.1 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:

#### 29.8.1.1 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.
- *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.

#### 29.8.1.2 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.

#### 29.8.1.3 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.

#### 29.8.1.4 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.

## 29.8.2 Frequency Scan Module

The frequency scan module (*ScnFreq*) monitors bus frequency.

#### 29.8.2.1 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.
- *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.

### 29.8.2.2 Frequency Scan Module - Limits

The limits for the frequency violation detection are set in this page. The *Maximum* and *Minimum* limits should be defined.

### 29.8.2.3 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.

### 29.8.3 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.
- *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 and 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.

#### 29.8.4 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.
- *Trip generator*: 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 generator 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.

## 29.8.5 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.

### 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.
- *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.

## 29.8.6 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.

### 29.8.6.1 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.
- *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.

#### 29.8.6.2 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.

#### 29.8.6.3 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.

## 29.9 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 30.6.1) are also supported.

---

### 29.9.1 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 13.9.9;
- 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 29.3.6;
- 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.

### 29.9.2 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.

## 29.10 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 29.10.1. An overview of the co-simulation concepts is given in Section 29.10.2. For information on how to configure the co-simulation tool, Section 29.10.3 is to be followed. Information regarding the execution and plotting of co-simulation results is provided in Section 29.10.4.

### 29.10.1 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.
- **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 29.10.1. 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*.

---

**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 15.3 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*.

## 29.10.2 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 29.11, 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.

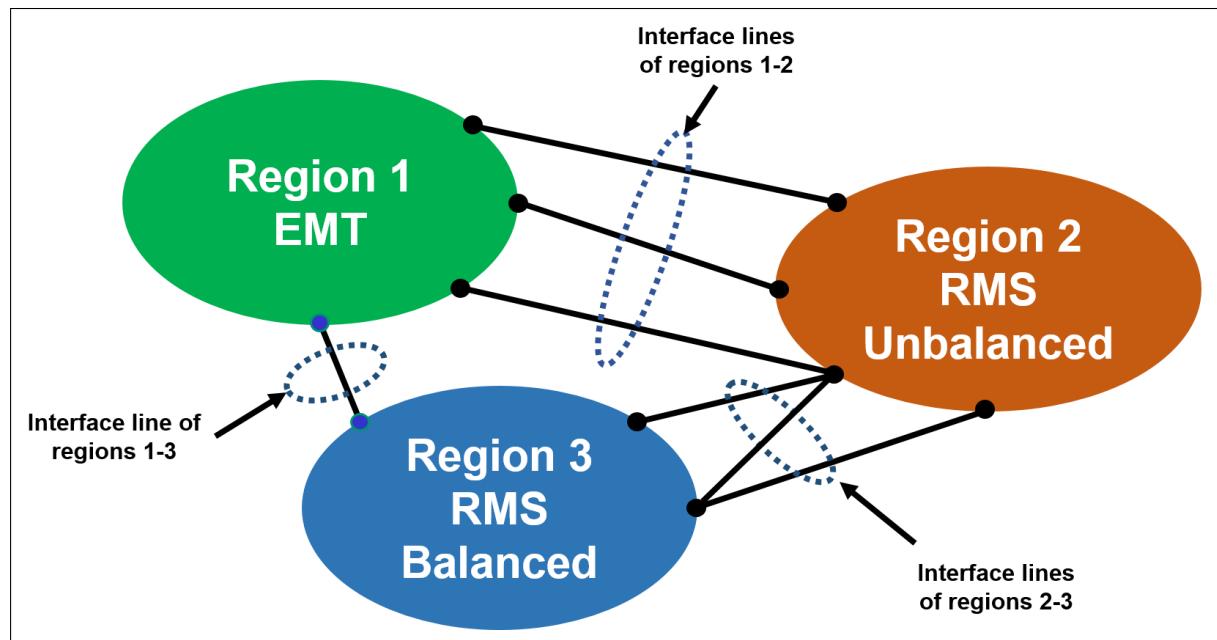


Figure 29.10.1: Example of co-simulation *domains* and corresponding *interface elements*

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 [29.10.1](#)) **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).
- *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.

### 29.10.3 Configuring the Co-simulation

A new calculation function has been developed for setting up a co-simulation, namely, 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 [29.10.5](#) 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)* .

An example of a *Multiple domain* co-simulation configuration is shown in Figure [29.10.2](#).

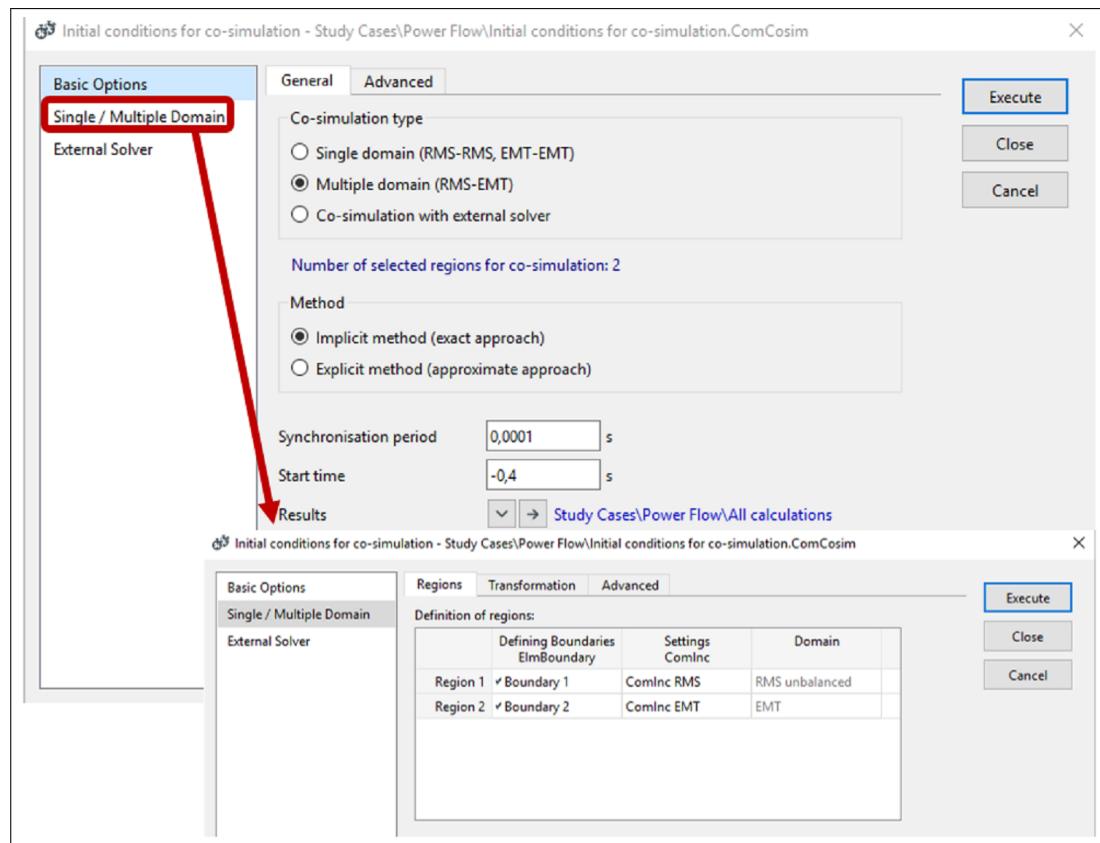


Figure 29.10.2: Co-simulation configuration options (example)

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 on the line element(s) which represent the co-simulation border. As an example, Figure 29.10.3 shows a simple two area power system interconnected via two AC lines, a candidate for a co-simulation based on two *regions*. For this setup, the two AC lines need to be contained within both boundary objects (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*.
- **Configuration of simulation options** is done via the *Initial conditions for co-simulation* by editing the default *ComInc* command (in the case of single domain co-simulation) or by assigning individual *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 29.10.2
- **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*.

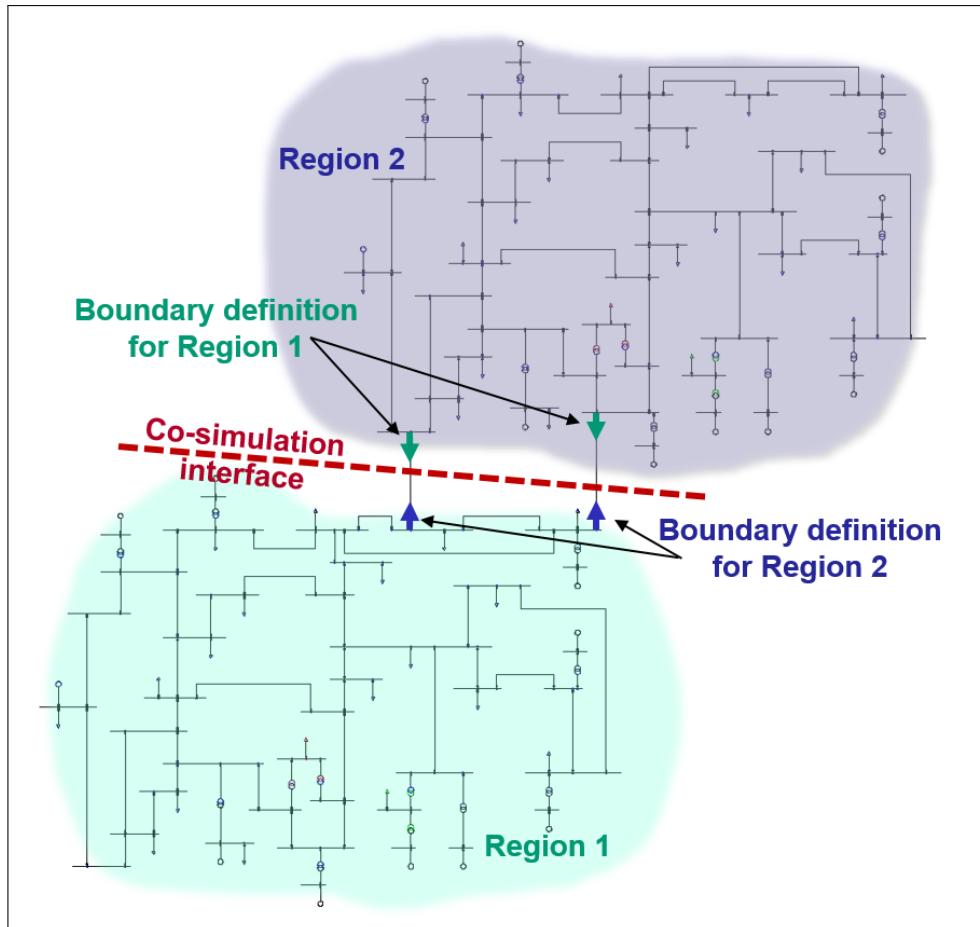


Figure 29.10.3: Co-simulation example for a two areas interconnected power system

#### 29.10.4 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 29.10.3).
- 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 29.10.5)
  - 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 the “y-Axis” page, double-click on an empty field of the column *Results file* of the *Curves* tabular list.
    - Navigate **inside** the main co-simulation *Results* object (e.g. “All Calculations.ElmRes”) and select the *Results* object which contains the network element of interest. Click *OK*.
    - Under column *Element* select the relevant element.
    - Under column *Variable* select the variable of interest. Click *OK*.

### 29.10.5 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**

*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 solver*: 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 29.11).

*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.

*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;

*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 conditions*: 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.

### **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).

### **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.
- *Different transformation per boundary branch:* Each *boundary branch* may have its own transformation settings. All possible combinations are to be defined in the corresponding tabular 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 - Advanced tab**

*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.

*Prediction for explicit method:* A prediction algorithm is available for the explicit method;

*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 22.4.

*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.

**External Solver page:** this page is relevant for the co-simulation with external solver. Refer to Section 29.11 for further information.

## **29.11 Co-simulation with External Solver**

This section provides guidance on the basic use and configuration of the *External solver* based co-simulation tool. An overview of the co-simulation concepts is given in subsection 29.11.1. For information on how to configure the co-simulation tool, subsection 29.11.2 is to be followed. Information

regarding the execution and plotting of co-simulation results is provided in subsection [29.11.3](#). Two commands are used extensively for co-simulation with an external solver:

- *Prepare co-simulation with ext. solver* (*ComCosimsetup*  ) - detailed in Section [29.11.4](#)
- *Initial conditions for co-simulation* command (*ComCosim*  ) - detailed in Section [29.11.5](#)

The terms and definitions used in this Section are identical with the ones used in the *Single/Multiple domain* co-simulation. As such, for an understanding of various concepts like *regions*, *boundaries*, *simulation units* and *domains*, refer to Section [29.10.1](#).

*Co-simulation* is based on a thin and open implementation of a peer-to-peer concept as described in the following [Specification](#).

The framework for *Co-simulation* supports any kind of combination of simulation methods:

- RMS balanced
- RMS unbalanced
- EMT

### 29.11.1 Overview of the External Solver Co-simulation

*PowerFactory* provides an open interface for distributed *co-simulation* that allows a single or multiple third party *simulation units* to carry out a *co-simulation* with *PowerFactory*. The tool is designed for use in two main applications:

- *Co-simulation* between *PowerFactory* and a third party simulation tool: 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 *co-simulations* via a TCP/IP based network.

The *Co-simulation with external solver* is organised in three functional layers, as shown in Figure [29.11.1](#) and detailed below:

- Layer 1 - Communication protocol: The network communication layer is based on the TCP/IP protocol.
- Layer 2 - Data exchange protocol: The data exchange layer is based on the widely accepted IEEE C37.118 standard “IEEE Standard for Synchrophasor Measurements for Power Systems”.
- Layer 3 - Co-simulation protocol: The co-simulation protocol is a thin and open implementation of a peer-to-peer (decentralised) co-simulation concept. The co-simulation specification documentation is available [here](#), thus providing a consistent framework of required steps and procedures for any external tool developers wishing to support co-simulation with *PowerFactory*.

The *Co-simulation* protocol is versioned. The participating processes must use the same protocol with the same version when running a *Co-simulation*.

The *Co-simulation* program flow is shown in Figure [29.11.2](#). It is organised as sequential activities, such as “Setup”, “Configuration”, “Initialisation”, etc. The individual processes have to iterate through the different activities in a synchronised way. That is, a process may only pass to the next activity if all neighbouring processes finished the current activity successfully.

As in the *Single/Multiple Domain Co-Simulation* case (refer to Section [29.10](#)), interface data is exchanged via the C37 channels at fixed time points in a synchronised way. A process will wait for all neighbours to send the data for a given synchronisation time before continuing simulation beyond that point in time.

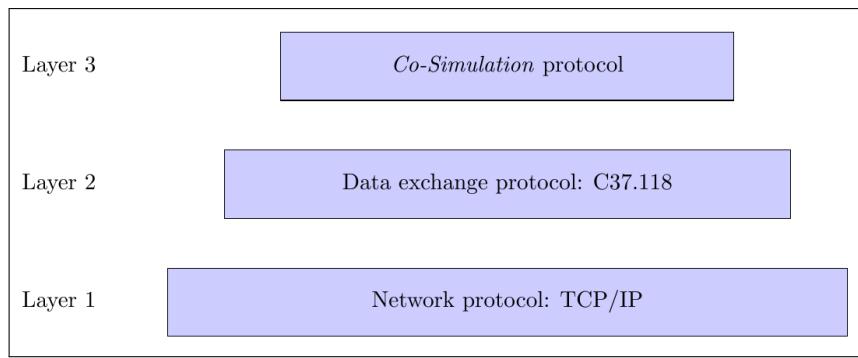


Figure 29.11.1: Co-simulation with external solver - underlying layers

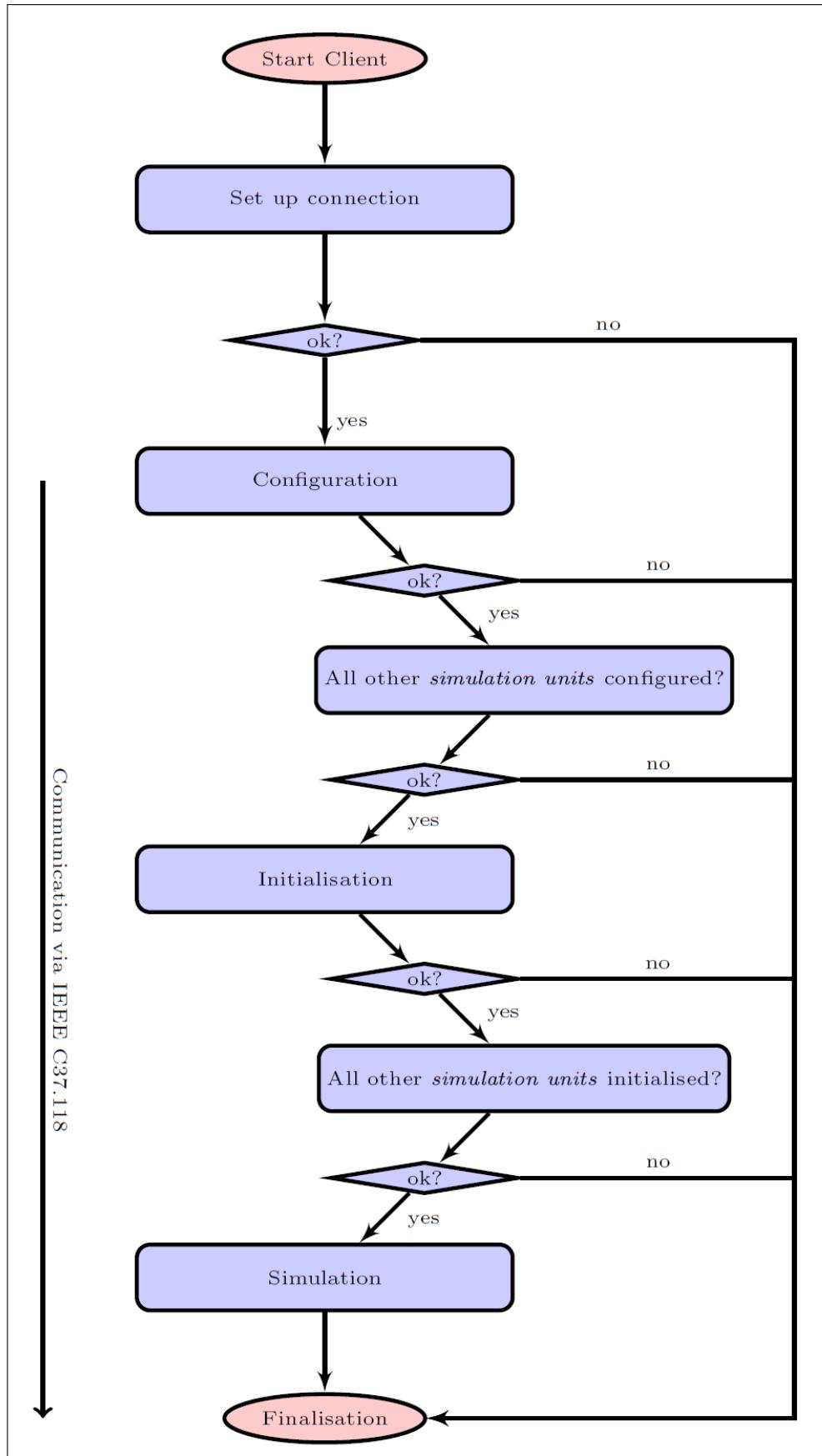


Figure 29.11.2: Co-simulation with external solver - program flow

### 29.11.2 Prepare Case for Co-simulation with External Solver

On *PowerFactory* side, a project can be prepared for co-simulation with a third party solver by using the command *Prepare co-simulation with ext. solver* (*ComCosimsetup*). An example of a configuration setup based on a two area power system (refer to figure 29.10.3) is shown in figure 29.11.3.

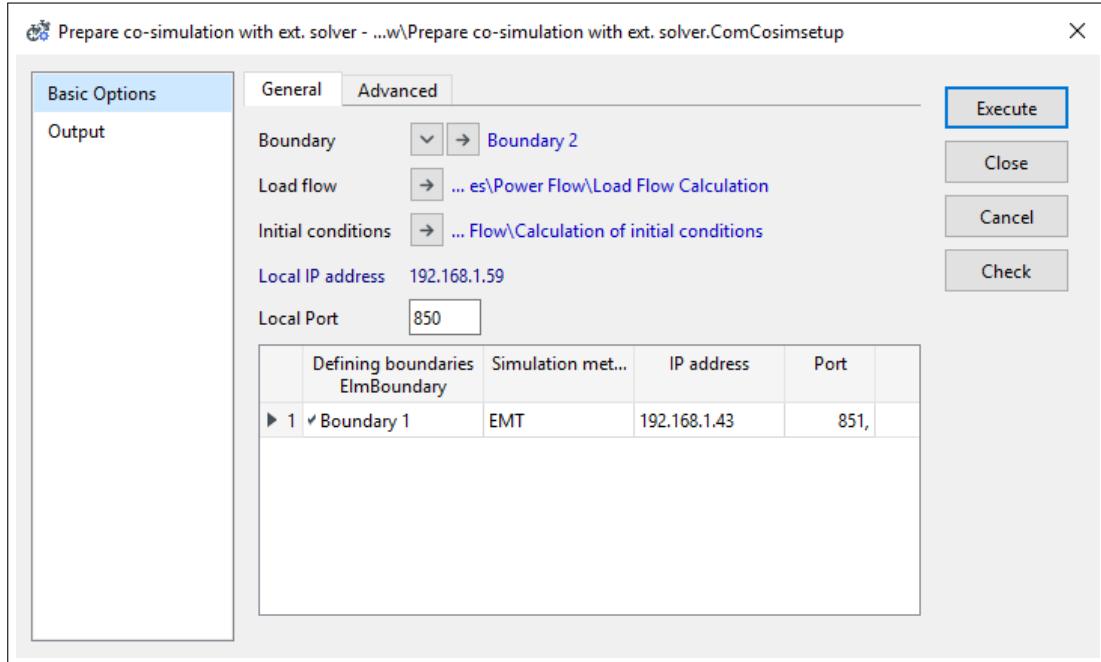


Figure 29.11.3: Co-simulation with external solver - configuration options (example)

*PowerFactory* automatically performs a network reduction of the power system *regions* which are to be simulated externally and also configures necessary voltage or current injection sources at the co-simulation interface, as shown in figure 29.11.4. It is important to underline that the *interface elements* must be a set of AC lines. The network reduction can be applied directly on the base network (existing network model is permanently changed) or using a network variation for flexible management of the power systems model (possibility to revert to the original network).

There are no specific requirements on the mandatory inclusion of the *regions* belonging to the external *simulation units* i.e. it is not necessary to represent in detail all components existing in the external solver co-simulation *region* (which are shown in figure below for illustrative purposes only). Nevertheless, a minimum representation in the *PowerFactory* model of the external *region(s)* is required: the *external region* must consist of the *interface elements* (e.g. AC line, relevant terminals) and the simplified network equivalents. The load flow across the *interface elements* should also be configured such that it correctly represents the initial starting point of the co-simulation.

The command *Prepare co-simulation with ext. solver* will create *Co-Simulation In/Out* elements (*ElmCosimext*). These elements contain the following configuration data for linking to the external processes:

- TCP/IP configuration
- C37 signal definitions

**Note:** *Co-Simulation In/Out* elements (*ElmCosimext*) contain the definition of all signals for any simulation method that might be applied on the local process (e.g. RMS, EMT). The channels opened to the external process(es) during *Co-Simulation* execution depend on the local simulation method, as well as on the simulation method of the remote process.

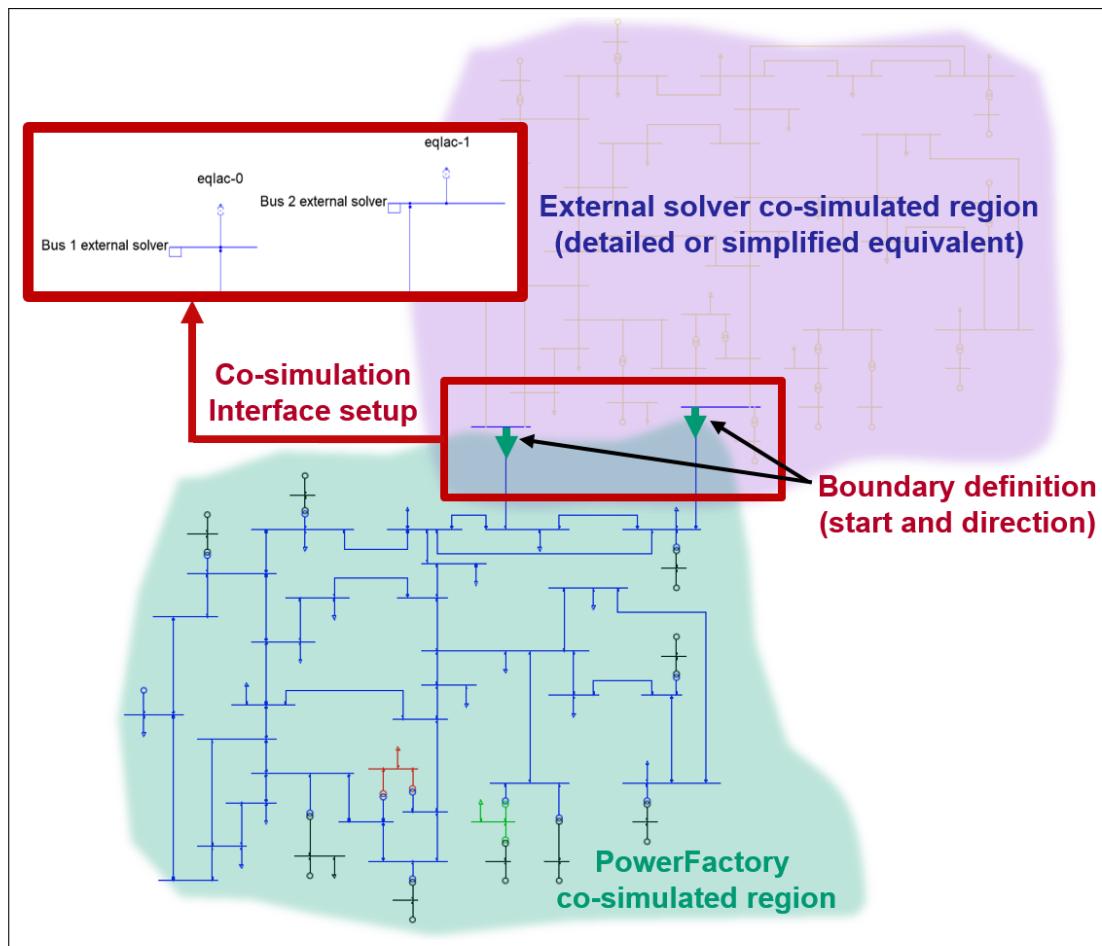


Figure 29.11.4: Co-simulation with external solver - two area power system example

The steps for setting up a co-simulation with external solver are as follows:

1. *Identification of the interface elements within PowerFactory*: The user needs to identify the AC line elements based on which the co-simulation interface will be set-up. This is a manual procedure.
2. Define the boundary associated with the *local region*. This is a manual procedure.
3. Define the boundaries associated with the *external regions*. This is a manual procedure.
4. Open the *Prepare co-simulation with ext. solver* command dialog . Detailed descriptions of each option in this command are available in Section 29.11.4.
5. Select the *Boundary* associated with the *local region*.
6. Set the *Local Port* to an unused port on the local machine.
7. Add rows in the tabular list for as many *external regions* are defined. Assign in the *Defining boundaries* column the boundary associated with each *external region*.
8. Select the corresponding co-simulation *domain* in the column *Simulation method*.
9. In the *IP address* and *Port* columns, assign the associated external *simulation unit* IP address and port values.
10. Switch to the *Output page* and choose whether the command should save the network modifications in a network variation or directly in the existing network model.
11. Click on **Execute**.

---

**Note:** It is assumed that a similar procedure is executed on the external *simulation units* as well. For distributed co-simulation done exclusively with *PowerFactory* machines, i.e. if the external *simulation unit* is *PowerFactory* then one will need to repeat the above steps on each additional *PowerFactory* machine.

---

### 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 case has been successfully prepared (refer to Section 29.11.2).
- Open the *Initial conditions for co-simulation (ComCosim)* command dialog and configure it appropriately (for a detailed description of options refer to Section 29.11.5):
  - Go to the *Basic Options* page and select the *Co-simulation with external solver* type.
  - Configure the *Synchronisation period* appropriately. The synchronisation period will strongly influence 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*.
  - Go to the *External Solver* page and configure the *domain* transformation settings. Set it to *Same transformation for all* if a single configuration is to be applied.
  - In the tabular list, edit(double click on) each interface object (“Co-Simulation In/Out” *ElmCosimext*):
    - \* *Basic Data* page - *Simulation in external process*: select here the co-simulation *domain* of the associated external process (i.e. third party *simulation unit*). The choices are: RMS balanced, RMS unbalanced and EMT.
    - \* *Basic Data* page - *Input* pane: The external *simulation unit* connection details are required here. Make sure that the IPv4 address and IP port fields are correctly configured, as defined in the external *simulation unit*.
    - \* *Basic Data* page - *Output* pane: The local *simulation unit* connection details are required here. Make sure that the IP port field is correctly configured, as defined at step 6 of the steps for preparing a case for co-simulation with external solver (refer to Section 29.11.2).
    - \* Click **OK**.
- Then **Execute** the *Initial conditions for co-simulation*. *PowerFactory* automatically starts the required parallel processes and initialises the simulation in each *region*.
- Execute the aforementioned *Initialisation* procedure on all external *simulation units* in a similar fashion.

---

**Note:** In the case of a distributed co-simulation exclusively done with *PowerFactory* machines, i.e. if the external *simulation unit* is *PowerFactory*, then **Execute** the *Initial conditions for co-simulation* on the external *simulation unit*.

---

- Start the simulation on the local machine using the *Run Simulation (ComSim)* command.
- Start the simulation on the external *simulation unit*.

---

**Note:** For distributed co-simulation exclusively with *PowerFactory* machines: start the simulation on the external machine 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)* contains the results of elements within the *local region*. Results of elements belonging to external *simulation units* are not available in the local *simulation unit*.

- 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 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*.

#### 29.11.4 The “*Prepare co-simulation with ext. solver*” (*ComCosimsetup*) Command

The co-simulation with external solver makes use of the *Prepare co-simulation with ext. solver* command . The command executes a network reduction on all *neighbouring regions* of the local *region* and introduces corresponding network equivalents (voltage or current injection elements). The changes to the power system model existing in *PowerFactory* can occur via a network variation (default option, changes can be easily undone) or by directly applying modifications to the existing network model.

##### **Basic Options page**

*Boundary*: Select here the *boundary* associated with the power system *region* that is to be simulated on the local machine. This *region* is referred to as the *local region*.

*Load flow*: A reference link to the currently used *Load Flow Calculation* command is provided.

*Local IP address*: The local IP address is shown here.

*Local port*: The IP port for inbound connections can be configured here.

*List of external regions*: A tabular list is available in which *regions* simulated externally are to be defined. Fill in the list with *boundaries* associated with all *neighbouring regions* of the *local region*.

---

**Note:** Although the external power system models do not need to be modelled in detail within *PowerFactory*, the *boundary* elements and a small power system equivalent must be manually represented within *PowerFactory* for each external region (e.g. a Thevenin equivalent with a given power flow setpoint). The power flow solution across the *boundary* should represent the intended operating scenario in which the co-simulation is to be run.

---

**Output page**: defines the network reduction options:

- *Create a new variation for reduced network*:- the command will save all applied changes in a separate network variation.
- *Change existing network*: - the command will save all changes directly in the existing network model.

#### 29.11.5 The “*Initial conditions for co-simulation*” (*ComCosim*) Command

The co-simulation with external solver makes use of the *Initial conditions for co-simulation* command . As the command is described in detail in Section 29.10.5, only the relevant information regarding the external solver based co-simulation is given here.

##### **Basic Options page**

*Co-simulation type:* select the option “Co-simulation with external solver”.

*Information message:* the number of already configured neighbouring *regions* for co-simulation is provided. The neighbouring *regions* are typically configured following the execution of the *Prepare co-simulation with ext. solver (ComCosimsetup)* command.

*Synchronisation period:* via this field, a user-defined fixed synchronisation period between the co-simulated *regions* can be provided;

*Initial conditions:* field available for *Single domain (RMS-RMS,EMT-EMT)* option only; a selection 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.

#### **External Solver page**

*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.

---

## **29.12 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.

### **29.12.1 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 *Calculate → 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.

### 29.12.2 Basic Data Page

The *Basic Data* page is shown in Figure 29.12.1.

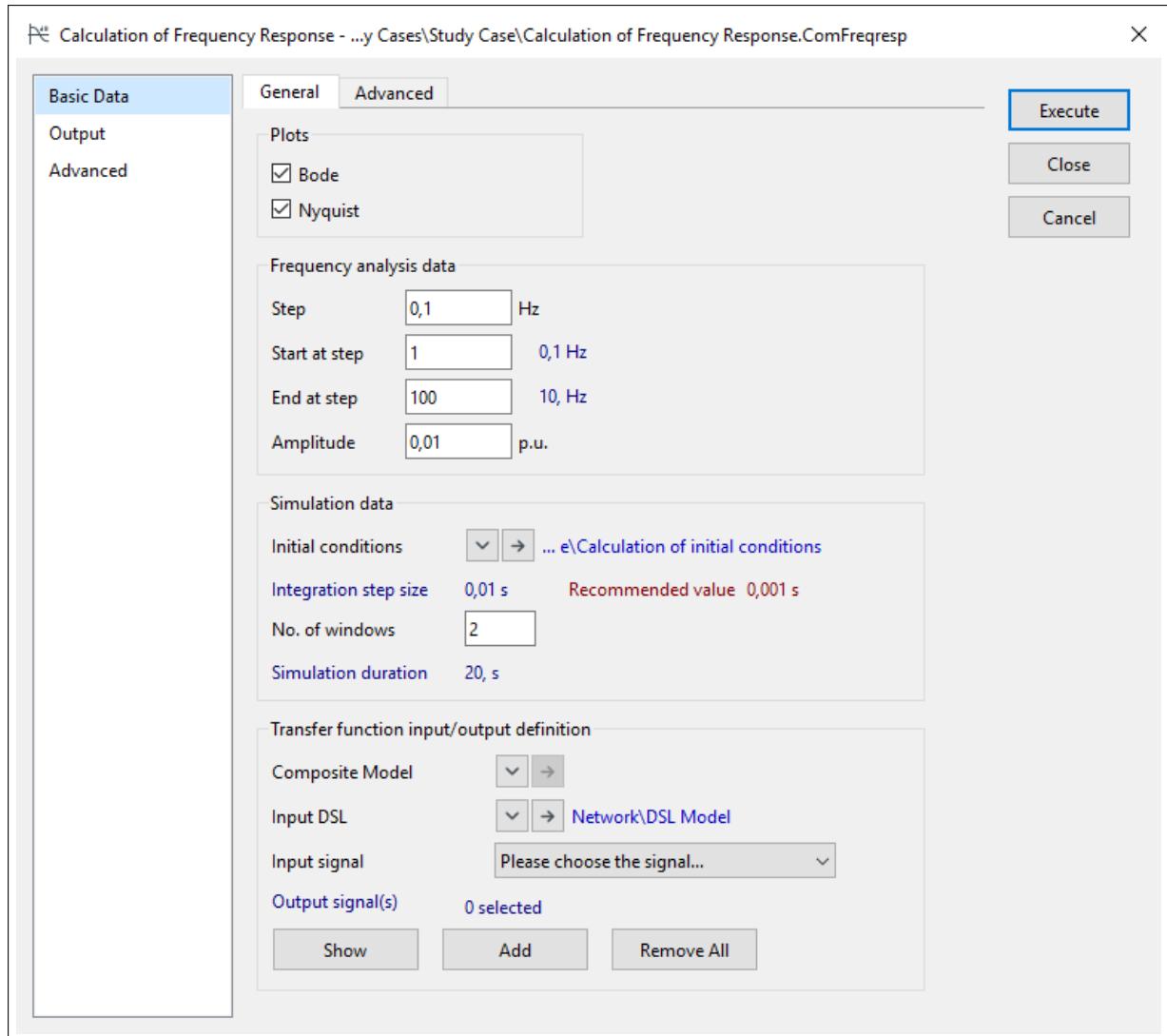


Figure 29.12.1: Basic Data Page of Frequency Response Analysis Command (*ComFreqresp*)

The following settings are available:

**Plots**: 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**: 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.
- **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. either within the *Simulation RMS* or in the *Simulation EMT* page) 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.

### 29.12.3 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).

#### 29.12.4 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*.

#### 29.12.5 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)

## 29.12.6 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 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*).

## 29.13.1 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 29.13 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 29.13 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 29.13.1. Calculation of Prony Analysis over a given *Time Range* is useful to detect changes in harmonic, interharmonic or subsynchronous oscillations, see Figure 29.13.2. Both figures below show an example of a generator current in a 60 Hz power

system in which a subsynchronous resonance is excited at  $t = 0.0$  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.

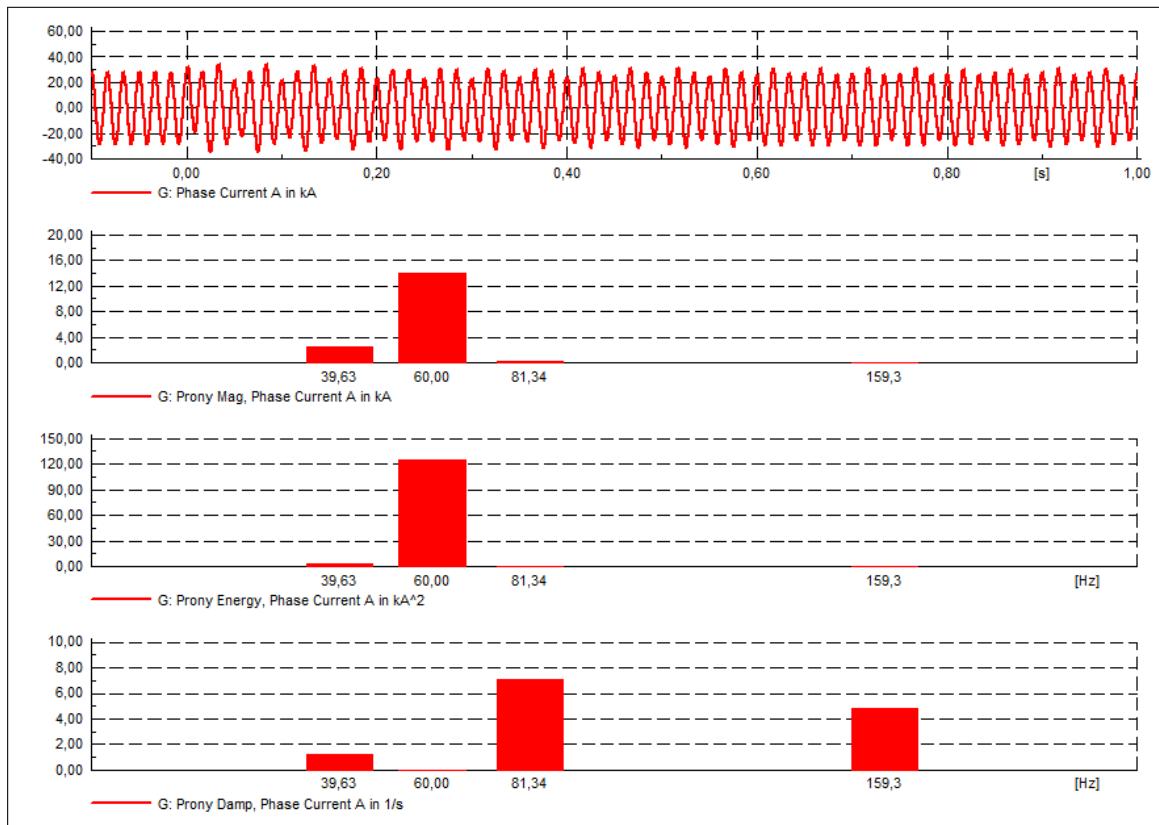


Figure 29.13.1: Prony Analysis results, three plots showing the magnitude, energy and damping of the modes of a given signal for a single *Time Point*

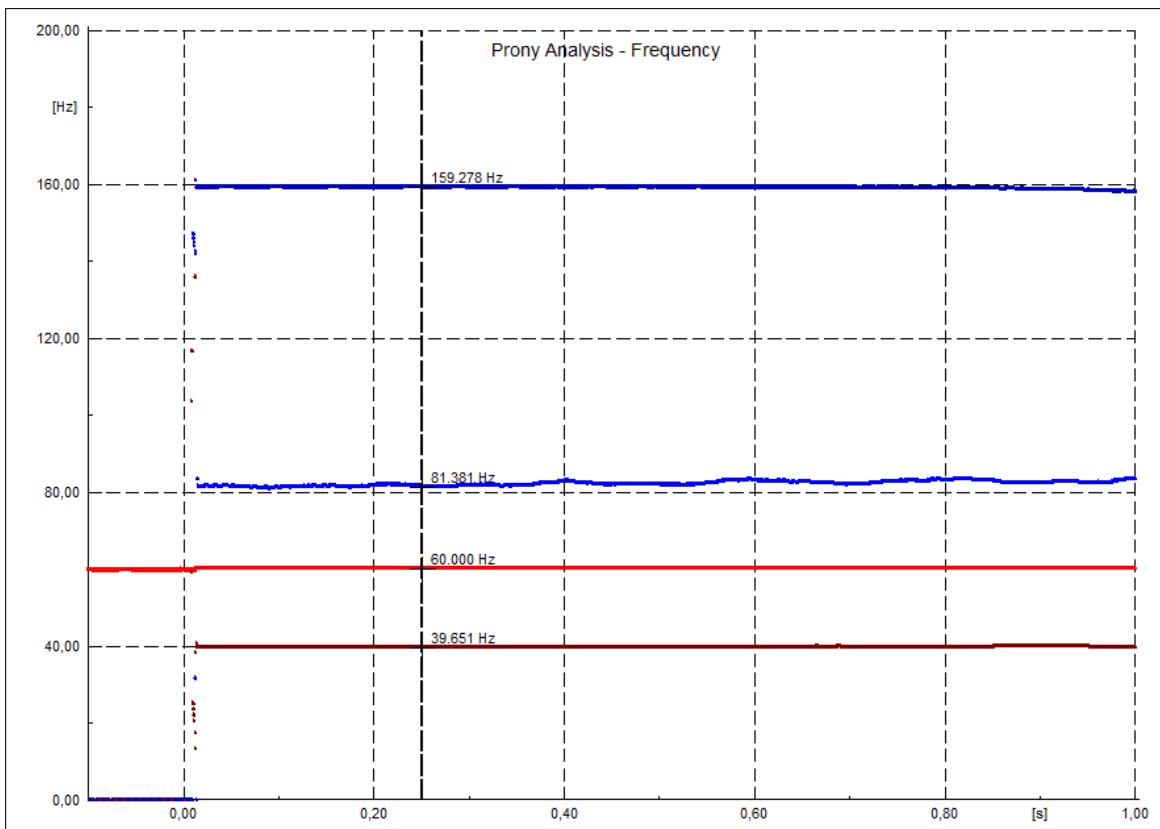


Figure 29.13.2: Prony Analysis calculated over a *Time Range*

## 29.13.2 Basic Usage

To perform a Frequency Analysis calculation several approaches are possible:

- Via the context sensitive 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 sensitive menu choose *Frequency Analysis*. . . . The mouse pointer will change appearance to  $\downarrow\uparrow$ .
  - 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  $\downarrow\uparrow$ .
  - 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.

### 29.13.3 Basic Options Page

The *Basic Options* page is shown in Figure 29.13.3.

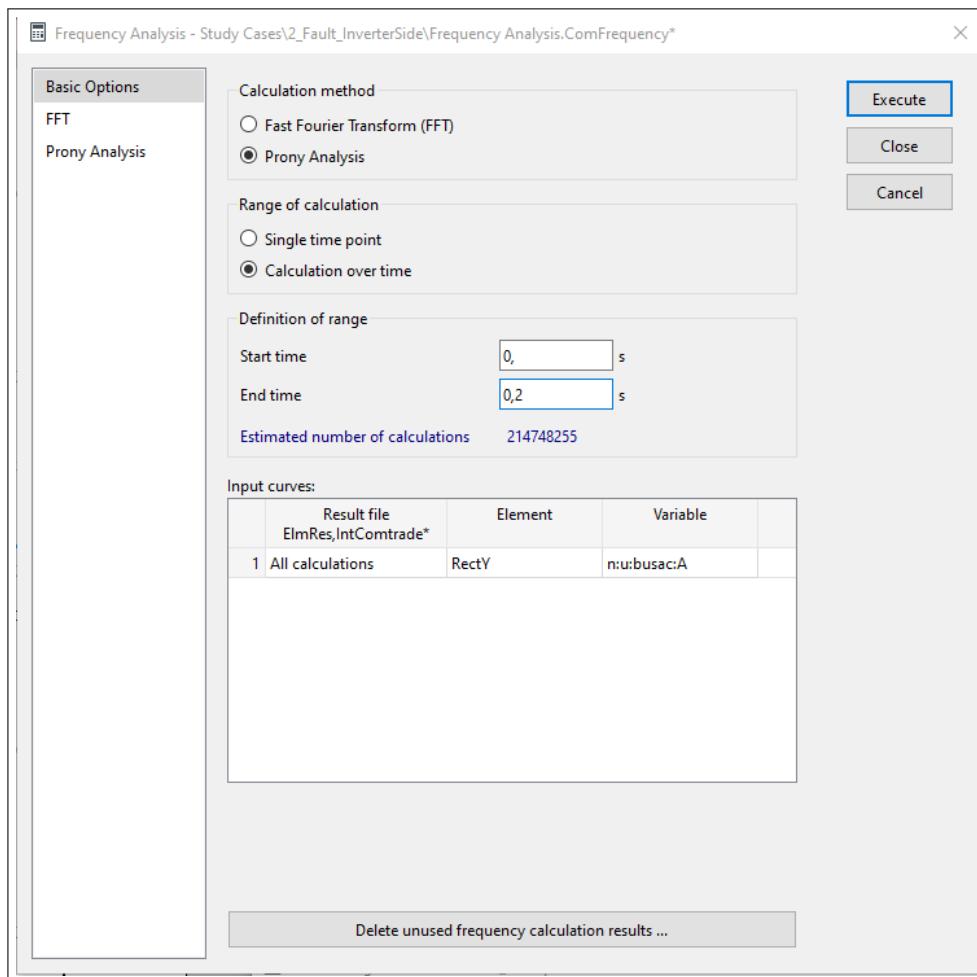


Figure 29.13.3: Basic Options Page of Frequency Analysis Command (*ComFrequency*)

The following settings are subject to user configuration:

**Calculation method**: The two calculation functions are made available via this pane:

- **Fast Fourier Transform (FFT)**
- **Prony Analysis**

**Range of calculation**: 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 [29.13.5](#).

---

**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 (.Elm-Res) 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**.

## 29.13.4 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 fulfills 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, compare Section 36.2.4.

- **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, compare Section 36.2.4.

### 29.13.5 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 29.13.3) 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 29.13.3) 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.

### 29.13.6 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.

---

### 29.13.7 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

### 29.13.8 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.

### 29.13.9 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 29.13.4.

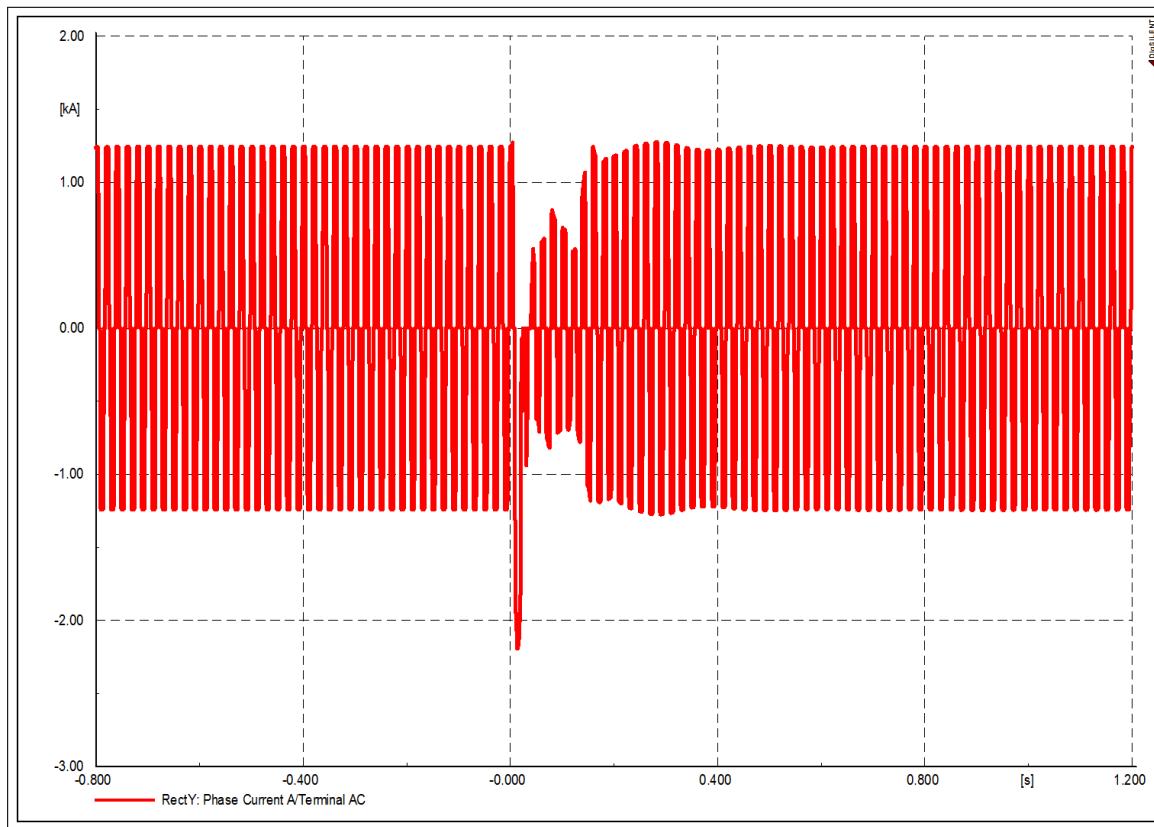


Figure 29.13.4: Original signal (AC phase current) to be processed by Prony Analysis

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 29.13.5. 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 29.13.5. 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.

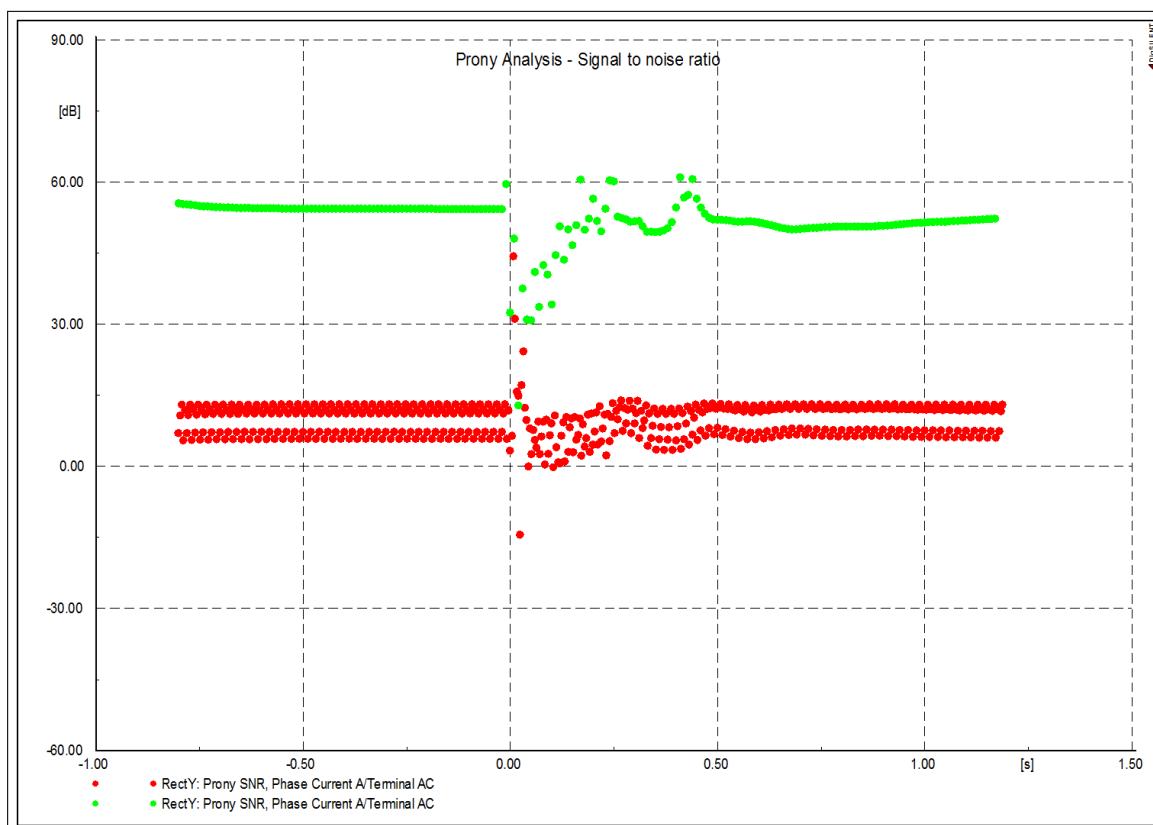


Figure 29.13.5: Evaluation of signal-to-noise ratio for different Prony Analysis settings

The final optimised results are shown in Figure 29.13.6. 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).

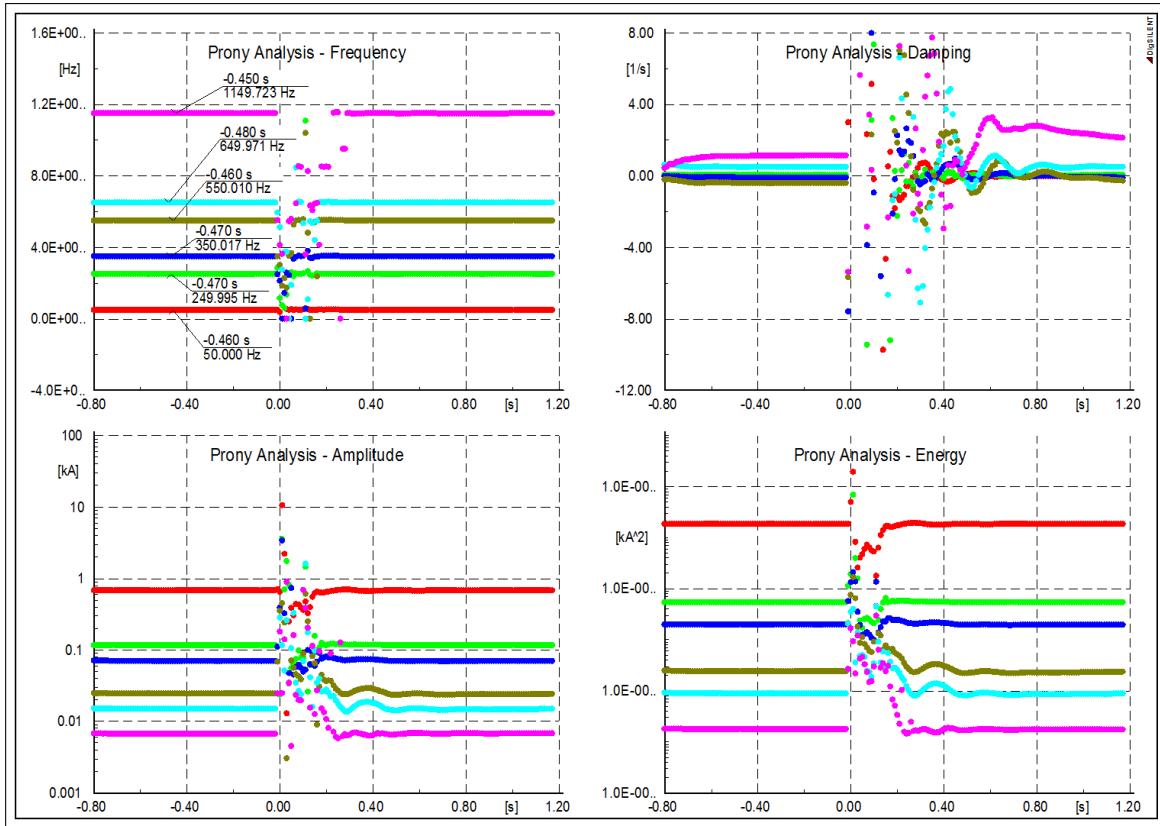


Figure 29.13.6: Prony Analysis on AC phase current of a thyristor rectifier before/during/after a three phase short circuit

### 29.13.10 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

Time domain analysis are typically based on predefined system models. In the majority of cases the standard IEEE definitions for controllers, prime movers and other associated devices and functions are used. Refer to Appendix B Standard Models in *PowerFactory* for a description of the standard models available in the global library.

For planning purposes, this approach might be acceptable. The predefined sets of parameters will allow a favourable and reasonable behaviour of the analysed system. This approach is often also applied to operation analysis, and the system should show a response similar to a real system.

For systems and configurations for which no IEEE models exist, such as wind generators, HVDC-systems, etc., powerful tools for user defined modelling are required. For this purpose, highly specialised, exact models can be created in *PowerFactory*.

In cases when manufacturers are able to supply exact controller models including real parameters, the system model can be improved by not using the IEEE standard models, but instead building a new block diagram of the individual controller/mechanical system to represent the device. This facilitates highly accurate system modelling.

Utilities and consultants often conduct system operation performance and optimisation studies, and therefore have a clear need for accurate methods and tools for creating accurate transient models for stability analysis.

This includes complex operation analysis and special component planning problems. This demand led to the development of highly flexible and accurate DIgSILENT *PowerFactory* time-domain modelling features.

Figure 30.0.1 provides an overview of the *PowerFactory* modelling approach, as discussed in this chapter.

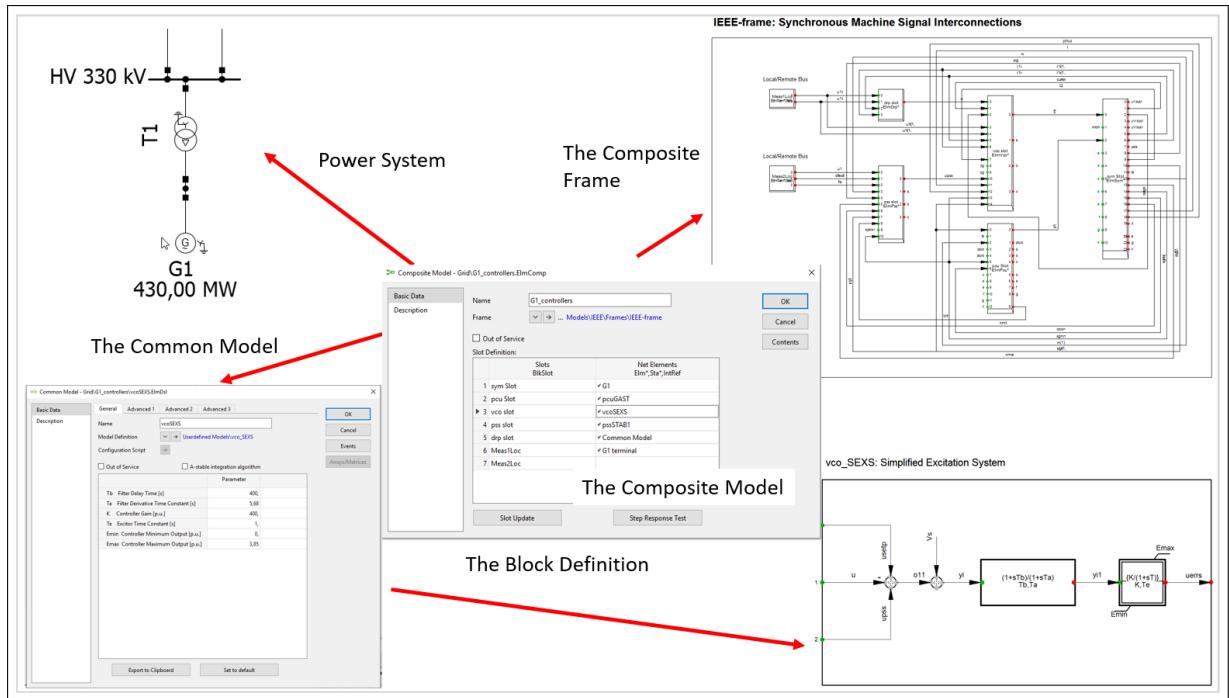


Figure 30.0.1: Overview of Modelling Approach

## 30.1 System Modelling Approach

System modelling for stability analysis purposes is one of the most critical issues in the field of power system analysis. Depending on the accuracy of the implemented model, large-signal validity, available system parameters and applied faults or tests, nearly any result could be produced and arguments could be found for its justification.

This is one aspect of the complexity of a transient stability study. The other aspect results from the often large set of time-domain models that are required, each of which may be a combination of other models. All these time-domain models are ultimately wired together into one large, single transient model from which the basic set of system differential equations can be obtained.

Given this complexity of a transient analysis problem, the *PowerFactory* modelling philosophy is targeted towards a strictly hierarchical system modelling approach, which combines both graphical and script-based modelling methods.

The basis for the modelling approach is formed by the basic hierarchical levels of time-domain modelling:

- The *DSL Block Definitions*, based on the “*DlgSILENT Simulation Language*” (DSL), form the basic building blocks to represent transfer functions and differential equations for the more complex transient models.
- The *built-in models* and *common models*. The built-in models or elements are the transient *PowerFactory* models for standard power system equipment, i.e. for generators, motors, static VAr compensators, etc. The common models are based on the DSL Block Definitions and are the front-end of the user-defined transient models.
- The *composite models* are based on composite frames and are used to combine and interconnect several elements (built-in models) and/or common models. The *composite frames* enable the reuse of the basic structure of the composite model.

The relation between these models and the way that they are used is best described by the following

example.

Suppose the frequency deviation due to the sudden loss of a fully-loaded 600 MW unit in a particular network is to be analysed. Depending on the network and the required detail in the calculated results, such analysis may ask for a detailed modelling of the voltage controllers, prime movers and primary controllers, or any other important equipment for all large generators in the system.

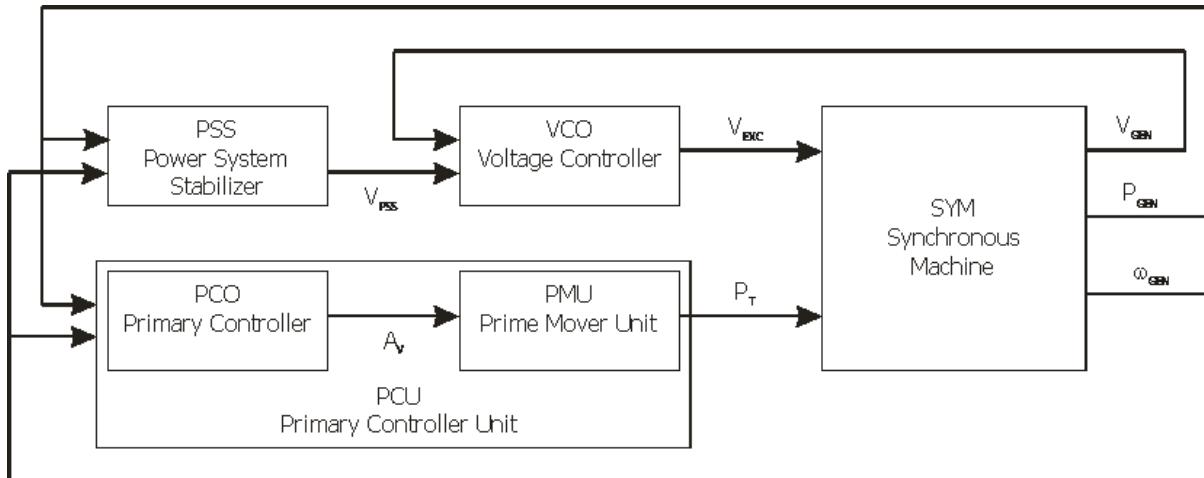


Figure 30.1.1: Example of a Composite or Power Plant Model

Figure 30.1.1 shows a typical configuration of a synchronous generator with power system stabiliser, voltage controller, primary controller, and prime mover model. The primary controller and prime mover can be summarised as the primary controller unit model. To create this kind of model, the following actions are required:

1. Transient models for each required controller type or unit type have to be defined (**Model/Block Definition**).
2. For each generator, the transient models of the individual controller must be customised by setting the parameters to the correct values (**Common Model**).
3. A diagram has to be made defining the connections between the inputs and outputs of the various models (**Composite Frame**).
4. For each generator, the diagram and the customised transient models are to be grouped together to define an unique 'composite' generator model (**Composite Model**).

It may seem unnecessary to include steps 2 and 3: it would be possible to create customised transient models for each generator directly, with 'burned-in' parameter settings, and to link these models to a generator without having to define a diagram first. This, however, would mean that one would have to create a new voltage controller, for example, for each generator in the system.

Often the design of many of these voltage controllers will be similar. The above approach removes the need to create copies of these controllers for each generator and avoids redundant copies of controllers or whole generator models.

Here the same relationship as that between individual controller (Common Model) and controller definition (Model Definition) is used; this time between the generic power plant diagram (Composite Frame) and the individual power plant (Composite Model). DlgSILENT PowerFactory uses two key objects in creating composite models, which can be compared to the **element** definition of the different elements:

- The **Common Model**(ElmDsl) combines general time-domain models or model equations (a Block Definition) with a set of parameter values and creates an integrated time-domain model.
- The **Composite Model**(ElmComp) connects a set of time-domain models inside a diagram (a composite frame) and creates a 'composite model'.

The following diagrams explain the relation between the Composite Model (which is using a Frame as type) and the Common Model (based on a block diagram as type) in detail.

- The **Composite Model** (*ElmComp*), Figure 30.1.2, references the definition of a composite frame. This composite frame is basically a schematic diagram containing various empty slots, in which controller or elements can be assigned. These slots are then interconnected according to the diagram, see Section Composite Block Definitions (part of Section 30.3.3: Defining DSL Models). The slots in the composite frame are pre-configured for specific transient models.
- The schematic diagram in Figure 30.1.3 shows a **Composite Frame** (*BlkDef*) which has one slot for a synchronous machine, one for a primary controller unit (pcu slot), and one for a voltage controller (vco slot). The composite model, which uses this composite frame, shows a list of the available slots and the name of the slot. Now the specific synchronous generator, voltage controller or primary controller unit model can be inserted into these slots.
- The synchronous machine that is used in the Composite Model is called a **Built-In Model**, see Figure 30.1.4. This means that such elements are pre-configured elements which do not need a specific model definition. Any kind of element which is able to provide input or output variables, e.g. converters, busbars, etc, can be inserted into the slots.
- The voltage controller, and primary controller unit, however, are user-defined **Common Models**, see Figure 30.1.5. The 'front-end' of all user-defined transient models is always a common model (*ElmDsl*), which combines a model definition with specific parameter settings. There are predefined definitions as well, so that the user can create model definitions as required.
- The common model has a reference to the **Model Definition** (*BlkDef*), which looks similar to the composite frame (shown in Figure 30.1.6). Here different blocks are defined and connected together according to the diagram. The input and output variables have to fit with the slot definition of the slot that the model is defined for.

Not all slots of the composite model must necessarily be used. There can also be empty slots. In such cases, the input of this slot is unused and the output is assumed to be constant over the entire simulation. The usage of composite models with a composite frame, and the common model with its Block Definitions are described in the next sections.

The design and creation of user defined common models using the “*DlgSILENT Simulation Language*” (*DSL*) can be found in Section 30.3 (User Defined (DSL) Models).

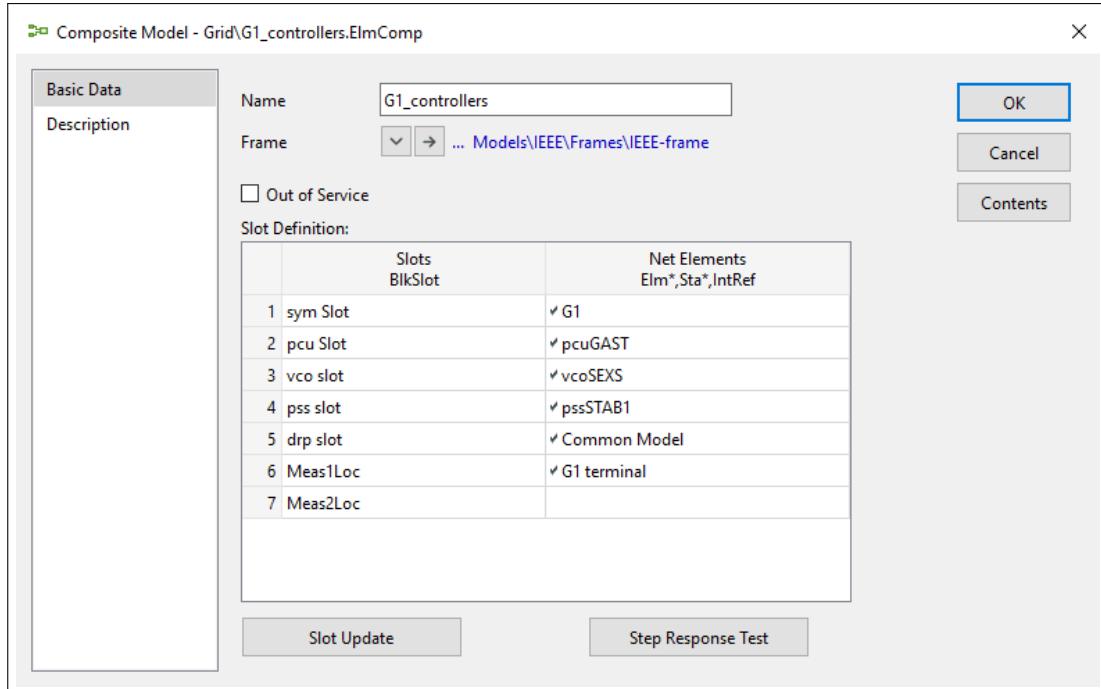


Figure 30.1.2: Example of a Composite Model Using the Frame “Frame\_Generator”

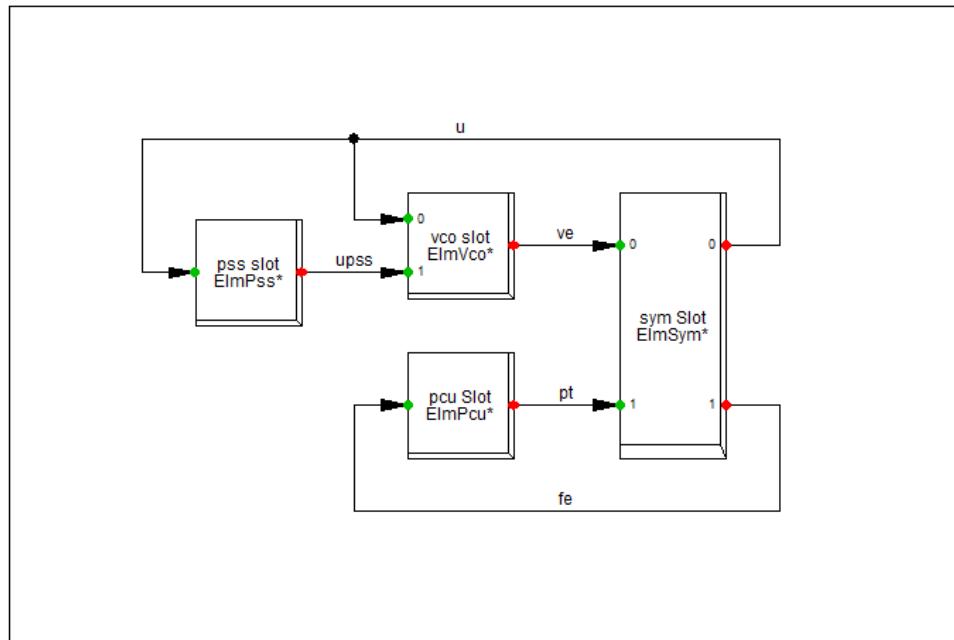


Figure 30.1.3: Composite Frame “Frame\_Generator”

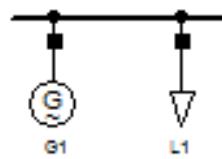


Figure 30.1.4: Generator "G1" (Built-In Model)

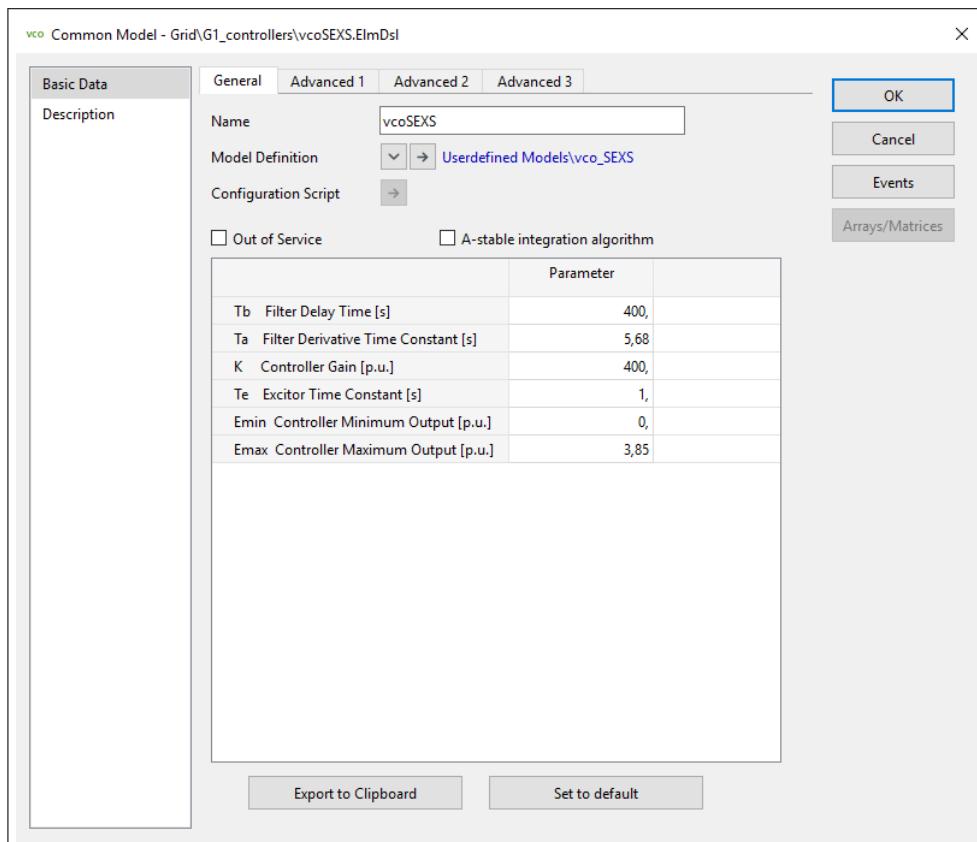


Figure 30.1.5: Example of a Common Model Using the Definition "vco\_simple"

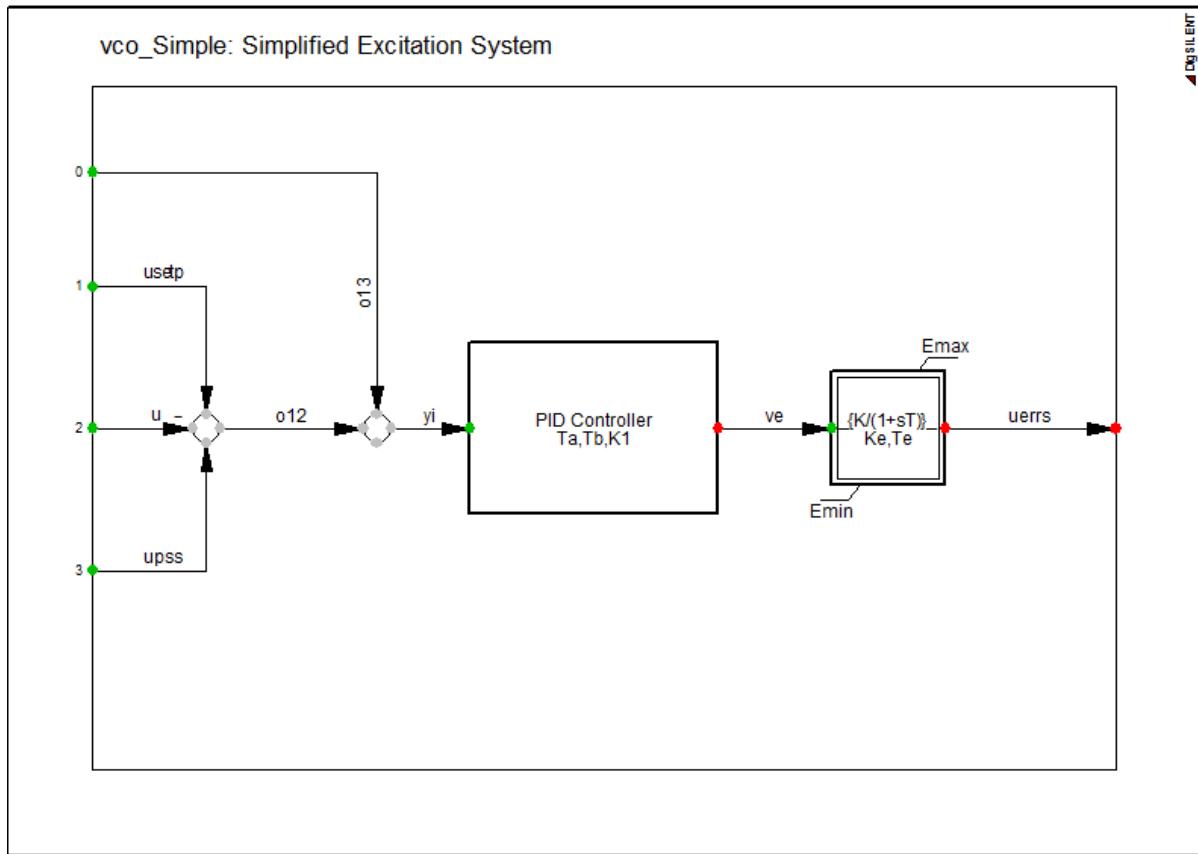


Figure 30.1.6: Example of a Model Definition “vco\_simple”

### 30.1.1 The Composite Model

A composite model element (*ElmComp*) can be created using the *New Object* ( icon), located in the toolbar of the Data Manager and selecting *Composite Model* from the available options. The next step is to select the composite frame. The composite frame can be stored either in the global library or in the local library, and is conceptually similar to a type definition for an electrical element. The composite model then shows the list of slots in the composite frame.

Existing controllers or models can be assigned to a slot manually by right-clicking the slot and selecting *Select Element/Type*. A data manager window will pop up and the user can then browse the grid for the element to insert into the selected slot.

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 select *New Element/Type* from the slot's context-sensitive menu. *PowerFactory* will automatically jump to the project Library and show a list of available user defined models (*ElmDsl*).

Selecting a model definition from the project library or the global library will open the element dialog of the newly-created common model, so that its parameters can be defined, similar to (for example) a transformer element. If no suitable model is found, a Block Definition has to be selected prior to setting the model parameters (see Section 30.1.2 (The Composite Frame) and Figure 30.1.6).

If an element is 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 common model, if it is stored inside the composite model in the Data Manager. These settings are explained in detail in Section 30.1.2 (The Composite Frame).

A faster method for defining standard composite models is to right-click on an object in the single line diagram and select *Define...* from the context menu of the element.

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 yet exists for the selected object.

Standard composite models are available for:

- The synchronous motor and generator;
- The asynchronous motor and generator;
- The static VAr system.

### Slot Update

The **Slot Update** button in the composite model (*ElmComp*) dialog will re-read the slot definitions from the composite frame and will cancel all invalid slot assignments.

A slot assignment is invalid when a model has been assigned to a slot which is not suited to receive this kind of model, i.e. a voltage controller cannot be assigned to a slot defined for a primary controller model.

All built-in models and common models which have been created for a specific composite model are stored in that composite model itself. The contents of a composite model are 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 in the composite folder, but in the grid itself.

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 the paragraph Classification in Section 30.1.2 (The Composite Frame).

### Step Response

The **Step Response** button in the composite model (*ElmComp*) dialog will activate the *Step Response* command (*ComStepres*).

Next to the references to the composite model, the template and the target directory, the two step response tests, which will be created, can be specified. The study case to be activated can also be selected. When **Execute** is pressed, *PowerFactory* will create a new folder in the current project named *Step Response Test*. Figure 30.1.7 shows this folder in the Data Manager.

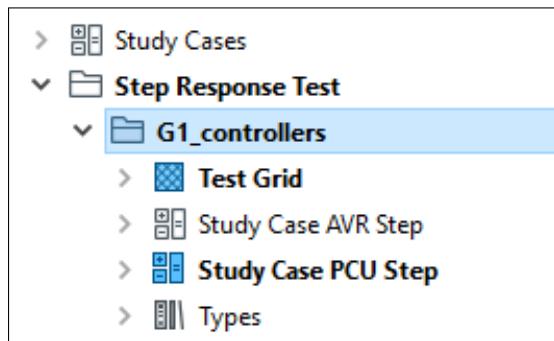


Figure 30.1.7: Step Response Folder in the Data Manager

Inside the *Step Response Test* folder, a second folder is created, named according to the composite model which is to be tested. Here the simple test grid can be found including only the generator, the complete composite model and a load. Additionally there will be two new study cases in which a step response for the AVR and the PCU, respectively, of the composite model can be tested.

The user can switch between these two study cases and previously-used study cases by activating and deactivating them.

---

**Note:** There is now no longer any connection between the original elements and the new elements of the composite model. Therefore, you can change any controller settings without changing your network.

---

After testing the controller, the folder *Step Response Test* can be deleted completely without loss of information in the original network.

### 30.1.2 The Composite Frame

A composite frame is a block diagram which defines two or more slots, their input and output signals, and the connections between them. A composite frame is defined graphically by drawing it.

Drawing a composite model frame is similar to drawing a normal block diagram. The main difference is that instead of common blocks, only slots may be used.

To create a new composite frame select the *Insert New Graphic*  icon on the main toolbar (in the graphics window) and then select *Block/Frame Diagram* and press **Execute** as shown in Figure 30.1.8. This new Block Definition will then be automatically created in the local library.

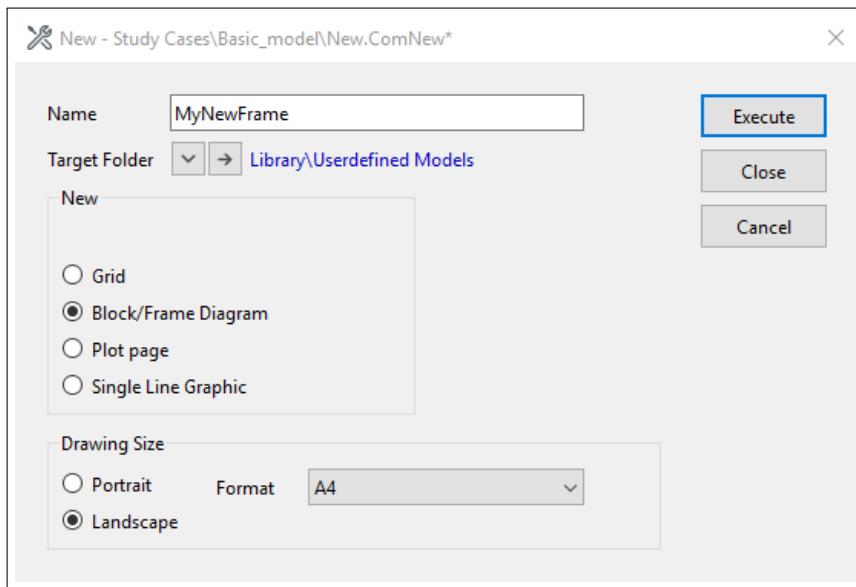


Figure 30.1.8: Creating a New Composite Frame

An empty diagram of the frame will appear in the graphics window. A slot is then created by selecting the  icon in the graphics toolbox and positioning the slot on the drawing surface by clicking once at the desired location. This is similar to placing elements in the single-line diagram.

An empty slot will be drawn on the page. To define the slot's input and output signals and different parameters, edit the slot by double-clicking it.

### 30.1.2.1 Name and Sequence

The name of the slot will appear later in the composite model dialog, and it is therefore recommended to name this element according to which slot it will be assigned (e.g. 'vco slot'). The Sequence parameter defines the order of the slots appearing in the composite model dialog.

### 30.1.2.2 Assigning a Block Definition to a Slot

A Block Definition (*BlkDef*) can be assigned directly to a slot. This option will simplify the handling of the slot and prevent errors due to miss-matched signal names of slot and assigned block.

To assign the external form of a Block Definition to the selected slot, edit the slot by double-clicking it and choose the *select* button for the "Block Definition" in the dialog. Now the Block Definition 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 machine voltage regulator (AVR) in the frame diagram, a predefined Block Definition 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 Block Definition 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 Block Definition.

---

**Note:** When a Block Definition is assigned directly to a slot, only the input/output signals are set automatically. The internal equations/definitions of the Block Definition are not implemented in the slot and the slot itself remains empty. There is always the need to create a common model, which is the model inserted into the slot of the composite model. When the slot refers to an outside Block Definition, 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.

---

### 30.1.2.3 Filter for

#### Class/Name Filter

There is also the possibility to specify a filter for the class name and/or for the model name to be inserted. This makes sense when (for example) only synchronous machines should be assigned to the slot. In this case, the class name *ElmSym\** would be entered. *PowerFactory* then will only allow the element class "synchronous machine" to be inserted into the slot. A filter for a specific (part of an) element name can also be defined.

### 30.1.2.4 Classification

The classification options only affect the external behaviour of the slot.

**Linear** The slot representation in the frame diagram will be as a linear or non-linear model.

**Automatic, model will be created** When this option is activated, the function 'Slot Update' (see Section 30.1.1: The Composite Model) will automatically create a DSL model and ask for a Block Definition from the library.

**Local, model must be stored inside** This option is activated by default. This means that when a **Slot Update** is executed in the composite model, *PowerFactory* will only search for elements which are stored inside the *ElmComp*. A reference to models which are stored outside, i.e. the synchronous generator in a plant model, will be removed from the slot.

Not all input or output signals of built-in elements or common models have to be used and defined in the slot. A slot may only have an input or an output signal.

For example, the voltage or frequency of an AC voltage source *EImVac* may be controlled by an external function. Therefore, the slot for the source will only have two input signals *u0* and *f0*. More information about drawing composite frame diagrams can be found in Section [30.2.1](#) (Drawing Composite Block Diagrams and Composite Frames).

### 30.1.2.5 Upper and Lower Limitation

#### Limiting Signals

There is also the possibility to enter 'limiting signals'. These signals are handled by *PowerFactory* exactly like normal input signals. The difference is only in the graphical representation in the block diagram. These signals will be shown as inputs on the top or bottom of the slot.

### 30.1.2.6 Variables

#### Input and Output Signals

The input and/or output signal(s) have to be defined for each slot. The available signal names for the Built-In transient models (Elements) can be found in the [Technical References Document](#)

The given input and output signal names in this slot dialog have to match the input/output signals of the given transient model exactly, or the signals will not be connected properly and an error message will result.

Only after one or more input and output signals have been defined for a slot, is it possible to connect the slot with signal lines to other slots. It is therefore recommended to first position and edit all slots and to draw the signal connections thereafter.

### 30.1.3 The Common Model

The common model element (*EImDsl*, *dsl*) is the front-end object for all user-defined Block Definitions. This means that user-defined transient models, but also the block diagrams that are ready-shipped with the *PowerFactory* program, cannot be used other than through a common model. The common model combines a model or Block Definition with a specific set of parameter values. The common model shown in Figure [30.1.9](#) uses the Block Definition "vco\_Simple".

Typically the model definition is implemented as a Block Definition, such as that shown in Figure [30.1.10](#).

A model definition contains block references which may in turn either point to a primitive Block Definition (see Figure [30.1.11](#)) or to another composite Block Definition (see Figure [30.1.12](#)). The structure of the Block Definition is thus recursive and the user should check that this recursive structure does not contain circular references to composite Block Definitions.

A primitive Block Definition contains one or more DSL expressions and forms a basic block for more complex transient models. A description of how to use and create DSL models can be found in Section [30.3](#) (User Defined (DSL) Models).

It is also possible to implement the model definition not as a Block Definition, but directly as a primitive Block Definition (Figure [30.1.11](#)), coded using DSL.

Each Block Definition generally has one or more parameters which can be changed to define the model's behaviour. Two kinds of parameters are supported:

- Scalar parameters, i.e. amplification factors, offsets, setpoints, etc.
- Two and three dimensional array parameters, which are used in the DSL lapprox()/lapprox2() and sapprox()/sapprox2() functions.

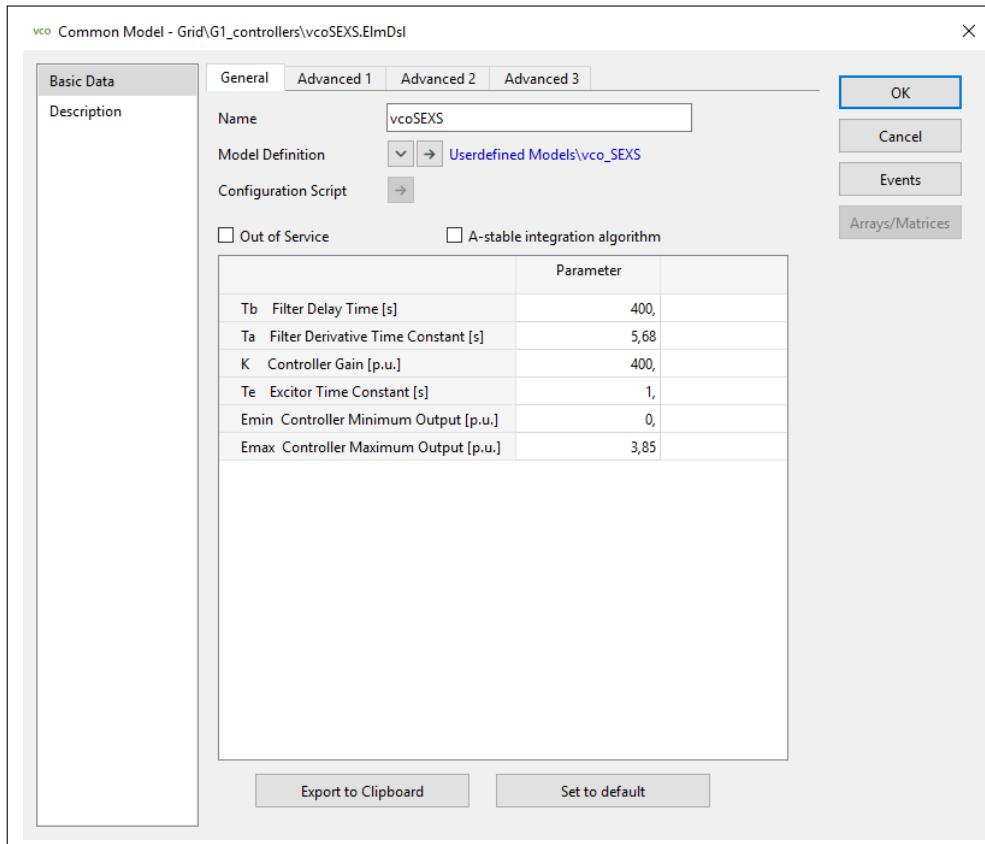


Figure 30.1.9: Common Model for the VCO

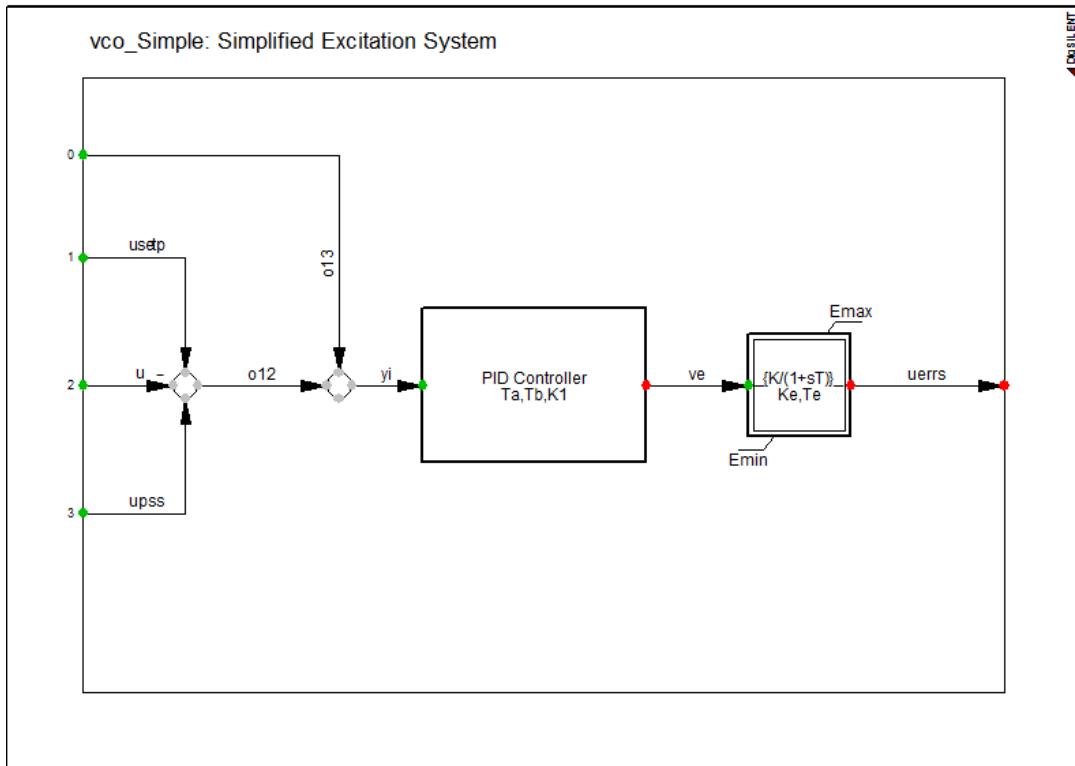


Figure 30.1.10: Block Definition of the VCO, Using a Sub-Definition



Figure 30.1.11: Implementation of a basic first order limiter block, using a DSL routine

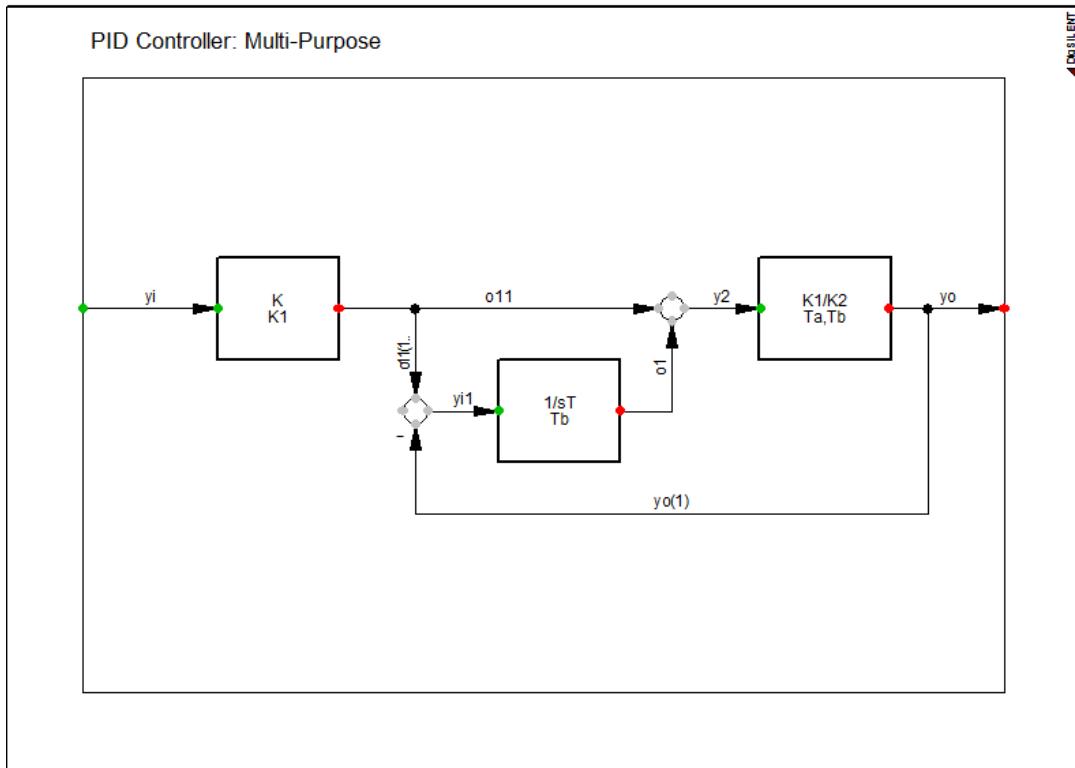


Figure 30.1.12: Implementation of the controller, defining a sub-block

To create a common model, use the *New Object* ( icon in the toolbar of the Data Manager and select *Common Model*. The block/model definition has to be selected first. Similar to the composite frame, this definition is either stored in the global library or in the local library.

The common model then displays the list of available parameters and arrays from the block diagram, as shown in Figure 30.1.13. All parameters are listed on the first page of the common model, and their values can be specified there.

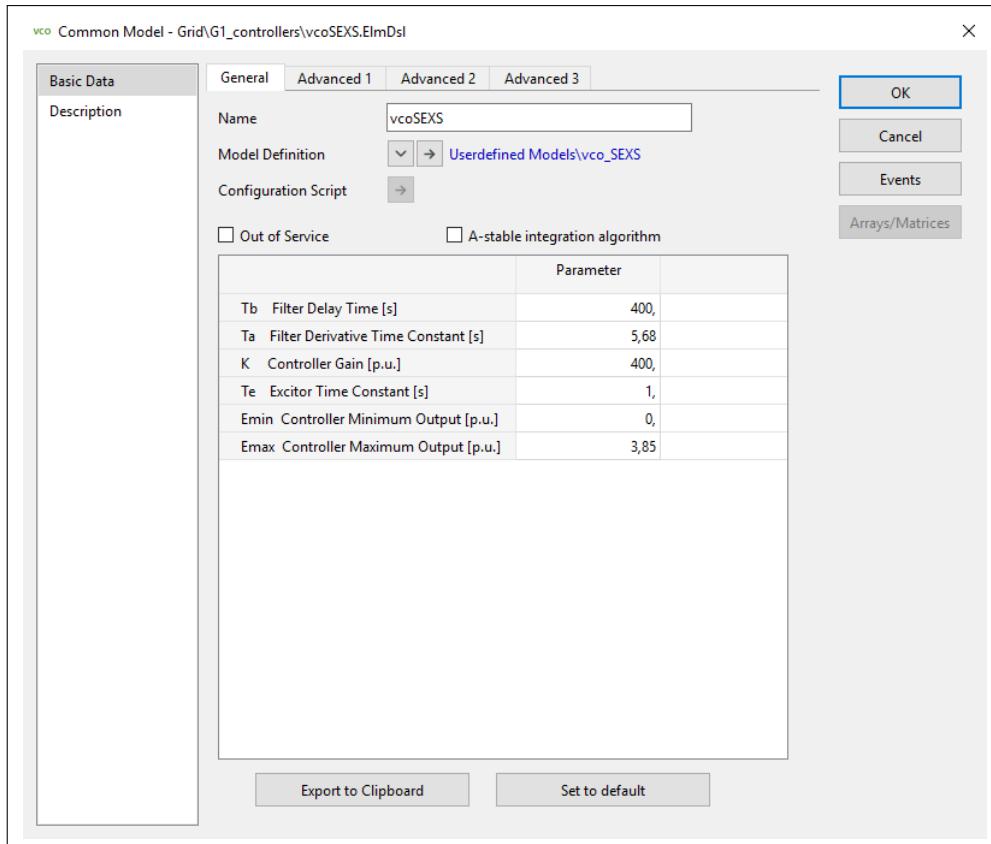


Figure 30.1.13: Common Model with Parameter List

If the selected Block Definition uses one or more arrays in its definition, then these arrays are displayed on the second page (for simple characteristics) and third page (for two-dimensional characteristics) of the *ElmDsl* object. In Figure 30.1.14 an example is shown for an array definition.

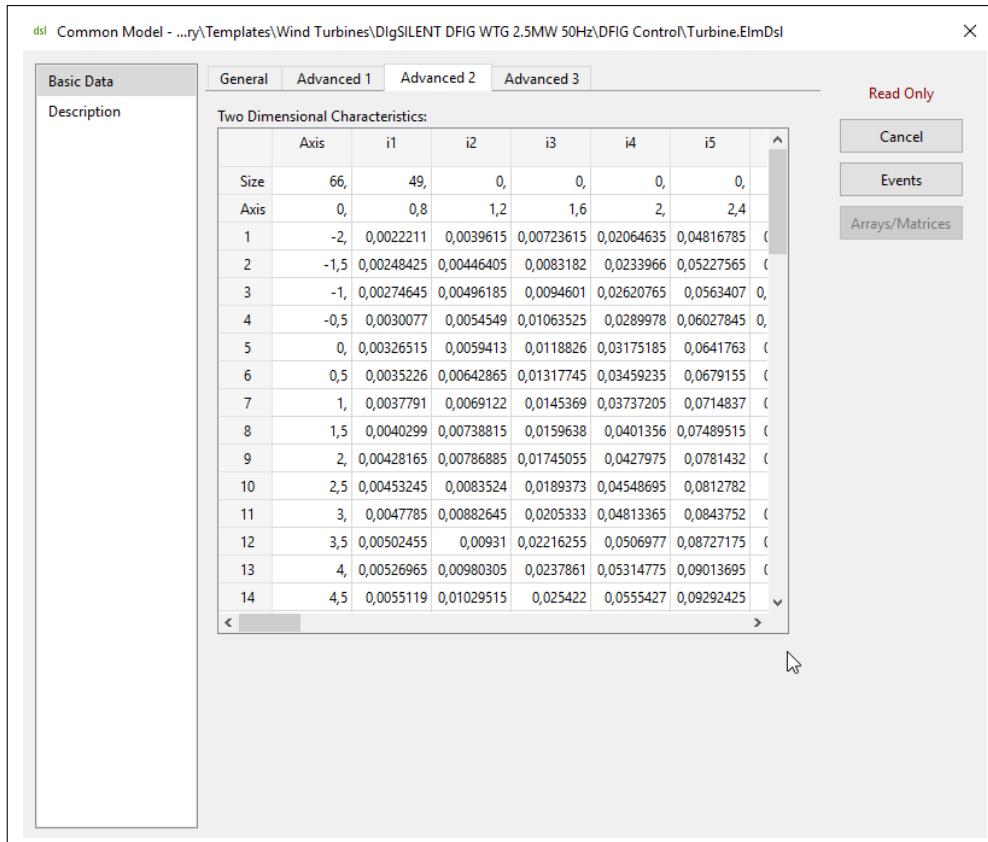


Figure 30.1.14: Common Model with Array List

The characteristics are defined as follows:

**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.

## 30.2 The Composite Block Definition

A composite block diagram of the model definition is a graphical representation of a mathematical transfer function, which produces one or more output signals as a function of one or more input signals. A block diagram may also have limits (minimal and maximal values) as input signals.

A block diagram may thus be described as:

$$(y_0, y_1, \dots) = \text{function}(u_0, u_1, \dots)$$

where  $y_0, y_1, \dots$  represent output signals 0, 1, ... and  $u_0, u_1, \dots$  represent for input signals 0, 1, .... These signals are all functions of time.

Block diagrams consist basically of the following elements:

- **Summation Points** which produce the single output  $y=(u_0+u_1+\dots)$
- **Multiplicators** 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  $y=u_1$
- **Signal Lines** which produce one or more outputs from one input:  $y_0 = y_1 = \dots = u$
- **Block References** which are used to include other Block Definitions.

Block references can be looked upon as macros that insert a low-level Block Definition inside a composite block diagram definition. A block reference may either point to another composite Block Definition or to a primitive Block Definition.

The *PowerFactory* program is shipped with a large set of primitive block diagrams for most common controller elements like PID-controllers, Dead Bands, Valve Characteristics, etc., and can be found in the *PowerFactory* tree under *Database /Library/Models/Global\_Macros*. These predefined DSL primitives may be copied and altered for specific needs.

A block reference is created by using the  icon in the graphics toolbox. This creates an empty square which can then refer to any existing Block Definition in the library.

**Note:** The composite frame and the model definition are very similar and their usage is almost identical. When creating one or the other *PowerFactory* recognises the class when you place the first slot or block. If you place a block () first, the icon for the slot will become inactive, so the user cannot inadvertently mix up slots and blocks in one diagram. See also Section [30.2.1](#) (Drawing Composite Block Diagrams and Composite Frames).

If the block type is selected, *PowerFactory* inserts all available parameters of the referred block. 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 Block Definition.

Signal lines are directed branches, connecting input and output signals. A single output line may be branched off and connected to more than one input terminal.

After the block reference has been edited, it will show the input, output and limiting signal connection points of the referenced Block Definition as one or more coloured dots on the left and right side, respectively, on the upper and lower side of the box. Signal lines may then be connected to these points. It is allowed to refer to the Block Definition 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 definition.

An example of a simple block diagram, comprising a multiplier, a summation point and a standard PI block, is shown in Figure [30.2.1](#).

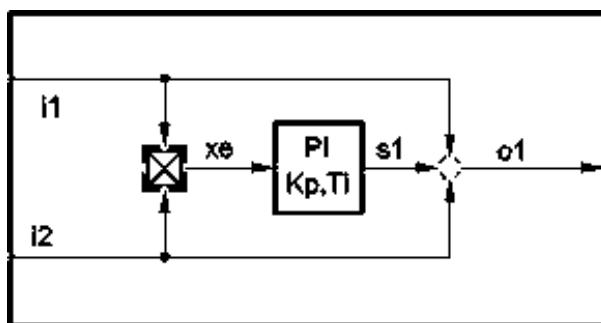


Figure 30.2.1: Example of a Simple Block Diagram

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 [30.2.1](#), this results in the following output:

```
model o1 = 'MyBlock' (i1,i2;x1;Kp,Ti;yi)
s1 = '\System\Library\Models\DSL\PI.BlkDef' (xe;x1;Kp,Ti;yi)
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 would also be possible to define exactly the same block diagram by entering the above DSL script manually and thereby create a primitive Block Definition.

### 30.2.1 Drawing Composite Block Diagrams and Composite Frames

Although the composite block diagram and the composite frame diagram should be distinguished from one other, they are drawn in the same way.

The basic distinction between a block diagram and a frame diagram is that the latter contains only slots and signals, whilst the block diagram must not contain any slots.

A new block or frame diagram can be created in various ways:

- Selecting the main menu entry *File* → *New* or **Ctrl-N** and then selecting the option *Block/Frame Diagram* from the *New command* dialog (*ComNew*);
- By clicking on the *Insert New Graphic*  icon on the toolbar of an open graphic, and selecting the option *Block/Frame Diagram*;
- By right-clicking on, or inside, a (library) folder in the active project in the Data Manager and selecting *New... → Block/Frame - Diagram* from the context-sensitive menu;
- By using the *New Object*  icon in the database manager and selecting *Block Definition (BlkDef)*.

---

**Note:** The latter two options only create a Block Definition object (BlkDef), but no graphic. This method is therefore not suitable for creating a composite block or frame diagram, but only for creating primitive Block Definitions by entering the DSL code.

---

In the first two methods, a graphic will be created and will appear in the open graphics board. A new graphics board will be created when no graphics board is open. The new block/frame diagram graphic will show a single rectangular block, which depicts the block or frame. The name of the new diagram will appear on top of the frame.

Inside this rectangle the following objects can be placed from the graphic toolbox for the block diagram:

#### Node objects:

- block references
- summation points
- multipliers
- divisors
- switches
- *different kinds of graphical objects*

#### Branch objects:

- signals lines

Inside a frame diagram only the following elements are allowed:

**Node objects:**

- slots
- *different kinds of graphical objects*

**Branch objects:**

- signals lines

These objects can be selected from the Drawing Toolbox. The toolbox also has buttons for pure graphical add-on objects (lines, polygons, rectangles, texts, etc.) as shown in Figure 30.2.2. It should be noted that the availability of this toolbox is according to whether or not the graphic is 'frozen' (🔒). When the graphic is not frozen, the toolbox is available, and likewise, when the graphic is frozen for editing, the toolbox is hidden.

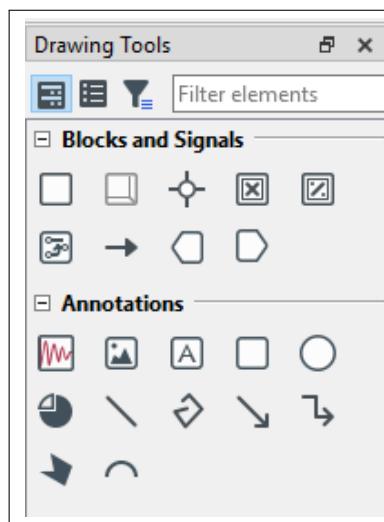


Figure 30.2.2: Block/Frame Diagram Objects

**Note:** When creating a frame or a Block Definition, *PowerFactory* recognises the type of definition when you place the first slot or block. Because a composite frame diagram may only contain slots and signal lines, creating a slot will disable all other node objects in the drawing toolbox. If you place a block (□) first, the icon for the slot (□) will become inactive, so you can't mix up slot and block elements in one diagram.

### 30.2.1.1 Adding a Block Reference

Drawing the block objects and connecting them with signals is done in a similar way as is done with elements in the single line graphic. A block reference is first displayed as an empty square which has to be edited in order to assign a (low level) block diagram to it.

Because of lack of information about the number of inputs and outputs of the new block reference before a (lower level) Block Definition is assigned to it, it is not possible to connect signals to the empty block. It is therefore recommended to first draw all block references and to assign Block Definitions to them. The block references then show all available input and output signal connections.

A block reference is edited by right-clicking on it and selecting *Edit* from the context-sensitive menu, or simply by double-clicking on it. The dialog as displayed in Figure 30.2.3 will pop up.

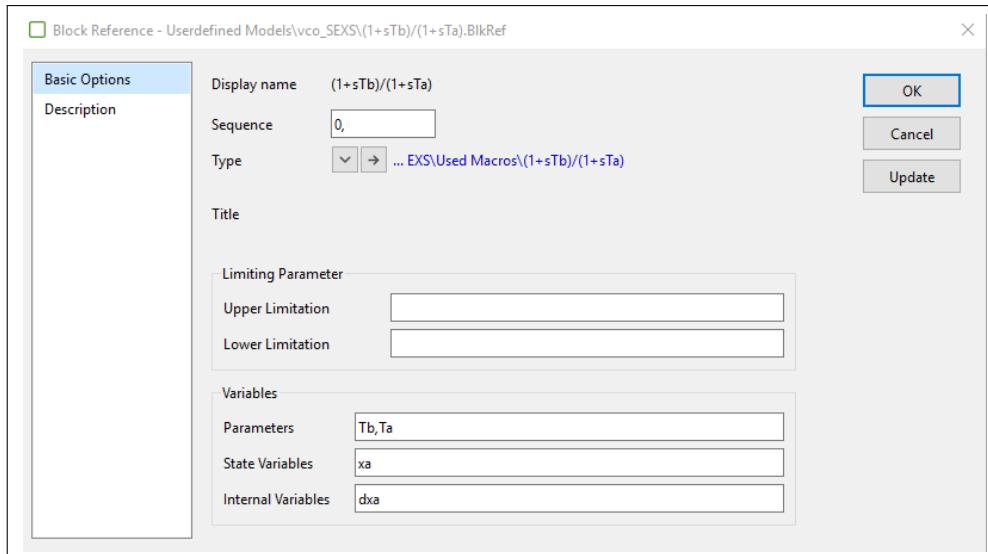


Figure 30.2.3: Block Reference dialog

Use the Select button ( $\rightarrow$  in Figure 30.2.3) to select a model definition. Predefined standard block diagrams for your usage are located in the folder *Database / Library / Standard Models*.

It is also possible to create a block in the graphical Block Definition by dragging Macros from the global library or project library into the drawing area of the Block Definition, using the Drag & Drop functionality.

### 30.2.1.2 Adding Calculation Blocks

#### **Summation Point**

Every dot can be used as an input to the summation point. The sign of signals at summation points can be changed by editing the summation point object. The “edit” dialog will pop up, where any connected input connection can be inverted. It should be noted that not all dots have to be used and only one dot can be defined as an output.

#### **Multiplier**

Every grey dot of this block can be used as an input or output of the multiplier. An output of three input signals will thus be:  $out=(in\_0*in\_1*in\_2)$ . It should be noted that not all dots have to be used and only one dot can be defined as an output.

#### **Divisor**

Every grey dot of this block can be used as an input or output for the divisor. The first input will be the numerator and thus will be divided by the second (and if existing, the third) input. The order of the signals will be clockwise beginning from the left. An output of three input signals will then be:  $out=(in\_0/in\_1/in\_2)$ . Note that not all dots have to be used and only one dot can be defined as an output.

#### **Switch**

Two input signals can be applied to this block, which will be connected to the output according to the position of the switch. Additionally a control signal has to be connected to the top, which will define the operation of the switch. If the control signal is 0.5 or less, the switch will stay in the displayed state, whereas a signal greater than 0.5 will cause the switch to change to the upper signal and the other way round. In the edit dialog the zero position of the switch may be altered.

### 30.2.1.3 Connecting Signals

After drawing and defining the block references, slots or other node elements, they can be connected with signal lines. After selecting the button from the graphical toolbox, a signal line is drawn by first clicking on a 'from' node (output of a block/slot), optionally clicking on the drawing surface to make a non-direct connection, and finally clicking on a 'to' node (input to a block/slot). The input and output terminals of common blocks and other node elements are depicted as coloured dots (see Figure 30.2.4).

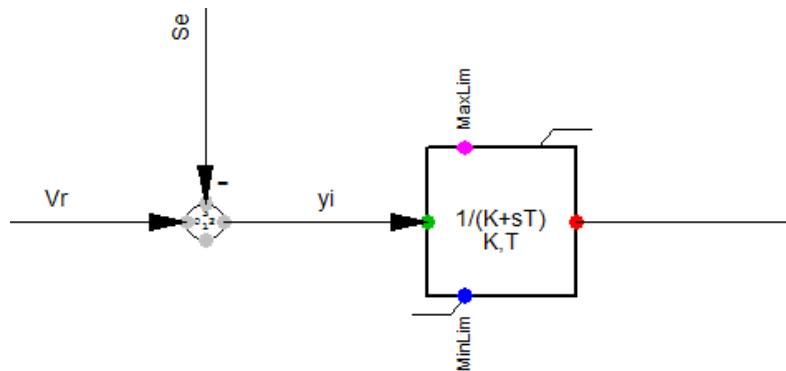


Figure 30.2.4: Block Signal Connections

- **Green:** Input
- **Red:** Output
- **Blue:** Min. Limitation
- **Pink:** Max. Limitation
- **Gray:** Every signal can be connected

The signal lines can also be edited in the corresponding dialog, which provides the possibility to change the name of the signal.

### 30.2.1.4 Multi-Signal Connections

Signals normally connect a single output parameter with a single input parameter. Especially in the case of three phase signals, as is often the case for voltage or current signals, multi-signal connections may be used.

A multi-signal is defined by writing two or more signal names together, separated by semicolons, e.g. "I\_A;I\_B;I\_C". In Figures 30.2.5 and 30.2.6, the multi-signal output and input of two Block Definitions are shown. Both blocks will show a single input or output connection point. They can be connected to each other by a single signal line, as illustrated in Figure 30.2.7.



Figure 30.2.5: Output Definition of Block1

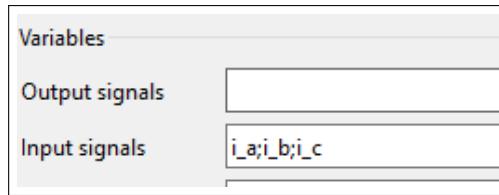


Figure 30.2.6: Input Definition

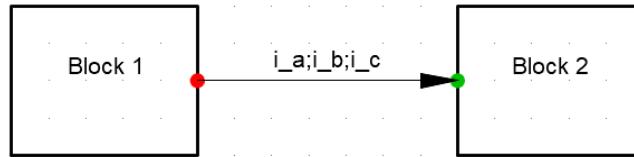


Figure 30.2.7: Multi-Signal Connection

**Note:** The number of variables and their order in the output signal must be equal to the number of signals in the input signal.

### 30.2.1.5 Block Diagram Input and Output Definitions

The composite block diagram normally has input, output and limiting signals of its own. Input signal points are defined by starting a new signal line on the left, top or bottom side of the frame enclosing block diagram. This will create a new input signal for the composite Block Definition.

New output signals are defined by ending a signal line by clicking on the right side of the enclosing rectangle frame.

Signals, which are thus connected to the rectangular frame, have the following meanings:

- connected to the left side: Input
- connected to the right side: Output
- connected to the bottom side: Minimum Limitation
- connected to the top side: Maximum Limitation

**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 assign the definition.

### 30.2.1.6 Resize

If a marked symbol has small black squares at its corners, it can be resized by left-clicking one of the squares, as can be seen in Figure 30.2.8. The cursor will change to a double diagonal arrow, and moving it (while holding down the left mouse button) resizes the object. Release the mouse when the new size is correct.

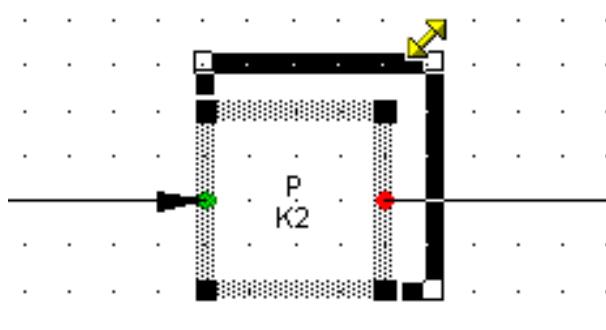


Figure 30.2.8: Resizing an Object

It is also possible to make the object(s) have a new size by clicking on one side of the marking box. The marked object(s) will only resize in one direction in that case. This is not possible for all objects. Some objects may only be resized with a fixed X/Y- ratio; some other objects cannot be resized at all.

#### 30.2.1.7 Additional Equations

After the internal structure of the block diagram has been defined graphically, the block diagram itself can be edited. This can be done without having to close the graphical representation of the block diagram. By left or double-clicking the enclosing rectangular frame, the block diagram edit dialog will pop up. This dialog will show all input, output and internal signals, as have been defined graphically.

On the Equations page, information and equations for the initialisation of the block can/has to be entered. Additionally, the name and the unit of the parameters to be defined in the common model can be specified (see also Section 30.3: User Defined (DSL) Models).

Additional DSL equations can be defined on the second page of the block diagram edit dialog.

## 30.3 User Defined (DSL) Models

System modelling for stability analysis purposes is one of the most critical issues in the field of power system analysis. Depending on the accuracy of the implemented models, large signal validity, available system parameters and applied faults or tests, nearly any result could be produced and arguments could be found for its justification.

A simple example illustrates this. In a 10 GW power system the expected steady-state frequency deviation when losing a fully loaded 2000 MW unit depends highly on the frequency dependency,  $K_f$ , of loads. Assuming a total system droop of 7% and a  $K_f$  value of 0, the steady-state frequency deviation will be approximately 700 mHz.

Now with a more realistic coefficient of  $K_f = 5\%/\text{Hz}$ , the steady-state frequency deviation is expected to be only 596 mHz. On the other hand, the frequency dependency might be slightly higher or lower, but the non-linear characteristics of hydro turbine efficiencies and steam valve non-linearities could be more relevant at a certain unit loading point. Consequently, as long as only one or two different loading scenarios are considered, average values with reasonable simple models may give acceptable results by tuning only some key parameters like the frequency dependency of loads or droop settings.

Thus system model structures and parameter settings are to be best evaluated against the following main criteria:

**System size** Large and small systems have different “key parameters”. Referring to the above ex-

ample, for a smaller power system the frequency dependency of loads is irrelevant, while in large systems such as UCTE or UPS/IPS, frequency dependency may cover the spinning reserve requirements totally.

**Unit size** Steady-state and transient behaviour of large units is more decisive for the overall system response than smaller units which might have a very negligible effect on the total system.

**System structure** Independent of system and unit size, the system structure may be more relevant than any other factor. This can be easily demonstrated when weak systems with a longitudinal geographical extension or appropriate substructures are analysed.

**System fault** Most relevant to system modelling considerations are the applied faults and related problems which are to be analysed. The analysis of system damping and PSS tuning will not necessarily require the boiler dynamics to be considered. On the other hand, load shedding optimisation and frequency restoration would not give appropriate results if mid- and long-term characteristics of relevant system elements are neglected.

**Study purpose** In general, for systems which are in the planning stage, typical models and parameters could be applied as long as there is no specific additional information available. However, a more detailed representation is necessary for system extensions, where a detailed model representation should form part of the performance specification. Special attention has to be paid to the analysis of operational problems and operation optimisation. For these cases, detailed modelling of the relevant components is critically important.

As soon as a detailed analysis and representation of system models is required, the subsequent questions to be asked are:

- How can the structures and parameters of the model be determined?
- Are IEEE models and additional manufacturers' block diagrams adequate and accurate?
- How could the available information be used within the power system analysis software?

The approach which is presented here and successfully applied in various projects can be called the "Advanced System Modelling Approach (ASMA)". Typical applications are:

- The analysis of controller problems and relevant malfunctions, especially under disturbance conditions;
- Optimisation of control parameter settings;
- Modelling of unconventional system structures and control concepts often found in industrial systems;
- Study applications for the design and specification phase of components and systems (e.g. power system stabiliser, generator and HVDC controllers).

For the ASMA approach, the following steps are critically important:

**Setup of system models** Based on the fundamental equations of engineering and physics, the basic algebraic and differential equations are to be set up according to the required degree of accuracy. In addition, all parameters such as time constants and gains which could be also derived from these basics, are to be calculated with the same degree of accuracy.

**Performance of system tests** In order to define all other parameters and, in particular, non-linear characteristics, system performance tests are the best method. In the majority of cases, frequency response tests will not permit the determination of any non-linear structure and its parameters. Special test procedures, which do not interfere with normal operation, have to be applied to focus on the steady-state characteristics, gains and time constants. These measurements are preferably executed with a highly accurate digital transient performance measurement system.

**System Identification** Non-linear, multi-input and multi-output system identification techniques are applied for system identification procedures. Typically, the mismatch between measured and identified data should be smaller than 2%.

**Comparison of measurements and simulations** Besides the analysis of subsystems and components, overall system performance is to be compared with the theoretical model for all relevant operating modes.

Of course, very strict application of the ASMA approach is not necessary for modelling relays and less complex or digital control functions, as these are clearly defined by their appropriate general and acceptance test documentation. However, independently of the analysed system, where the system representation cannot be matched to a classical IEEE or any other standard model, there is a substantial need for an easy to use and flexible method for the realisation of individual models.

### 30.3.1 Modelling and Simulation Tools

As already indicated, the most critical and decisive factor for reliable simulation results is the accuracy and completeness of system model representation for identification and simulation purposes. Methods for solving this task range from the classical and traditional way of using software which allows interfacing of user-defined models at the FORTRAN/C level - typically via connection lists - to the block-oriented approach which is based on the provision of predefined low-level block macros being connected at the case definition level.

In addition, most modern commercially available general purpose simulation tools may be used for flexible and specific system representation. Unfortunately, this approach does not adequately cover the special electrical system load flow characteristics.

In order to provide a very flexible modelling and simulation tool, which forms part of a stability program, a control system based simulation language was developed. The following describes the main features of the *DIGSILENT Simulation Language (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

### 30.3.2 DSL Implementation: an Introduction

The *DIGSILENT* Simulation Language is used to define new dynamic controllers which receive input signals from the simulated power system and which react by changing some other signals.

DSL itself can be looked upon as an add-on to the transient analysis functionality of *PowerFactory*. During the simulation, the model equations of the 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 transient simulation of the combination of the power system and its controllers.

The DSL main interfacing functions are:

**Signal input and output channels:** Any variable defined within the kernel (currently more than 2500) and in a DSL model, can be accessed in a read-and-write mode. Main and sub-address features are implemented allowing the access of any signal existing in the system or to build up complex structures such as hardware-based modules taking equipment “rack” and “function card” structures into account.

**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.

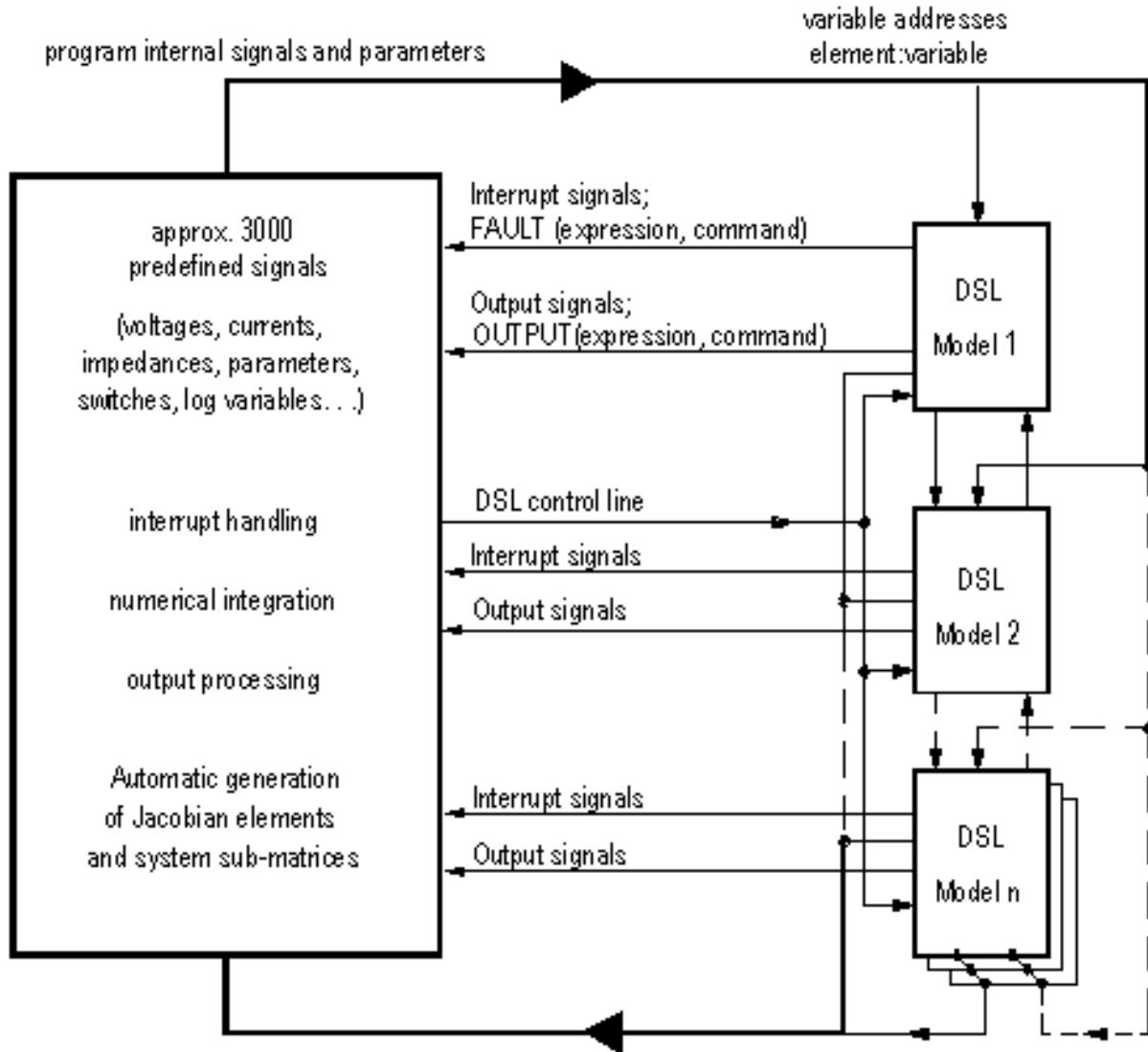


Figure 30.3.1: Structure of the *PowerFactory* DSL System

The structure of a DSL model is best explained by an example. This example considers a prime mover unit model of a simple hydro turbine. This DSL model has been defined graphically, and contains one embedded DSL macro. This embedded macro models a single integrator and is defined by programming it.

The basic method for designing new DSL models is as follows:

1. A set of basic DSL models is created. These models implement simple, “primitive” controllers like a “first order time lag” or a “PID” controller. The *PowerFactory* program is shipped with a large

number of these primitive controller models. New primitives are created by programming their differential equations and signal settings, using the DSL language.

2. The more complex controller is created graphically by drawing its block diagram. This kind of block diagram normally uses references other DSL models which are thus combined into a more complex controller. Controller references may be used to include DSL primitive models into the complex model, but may also refer to other graphically defined complex models. Highly complex controllers may thus be designed in a hierarchical way, by designing sub-models and sub-sub-models, where the DSL primitives form the lowest level. Section 30.3.3 (Defining DSL Models) describes these procedures in detail.

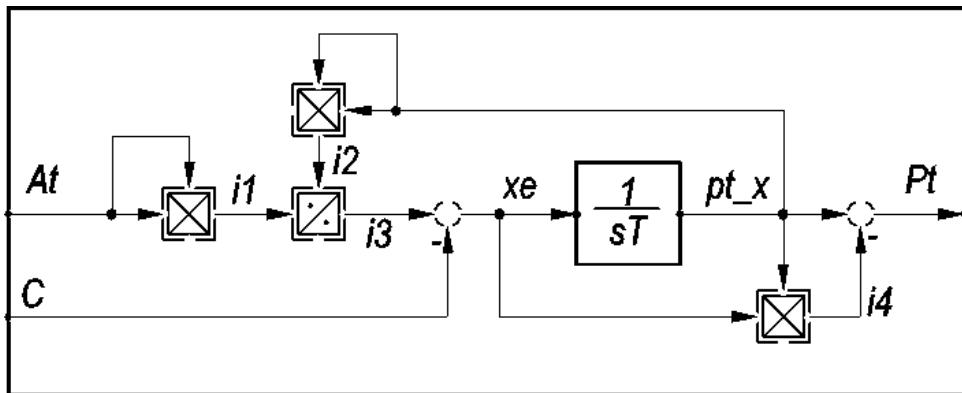


Figure 30.3.2: Diagram of a Simple Model of a Hydro Turbine

Figure 30.3.2 depicts the model definition that was used to define the hydro turbine model. The resulting DSL code, as shown in the output window when a graphical *Rebuild* (↻) is performed is:

```

1. model Pt = 'pmu_hydro'(At,C;x1;Ti;)
2. pt_x = 'I.BlkDef'(xe;x1;Ti;)
3. i3 = i1/i2
4. i1 = At*At
5. i2 = pt_x*pt_x
6. i4 = xe*pt_x
7. xe = i3-C
8. Pt = pt_x-i4

```

The line numbers have been added for readability. The corresponding Block Definition shows:

```

Output Signals : Pt
Input Signals : At, C
State Variables : x1
Parameter : Ti
Internal Variables

```

The example describes a simple hydro turbine model with the input signals A\_t and C and the output signal P\_t.

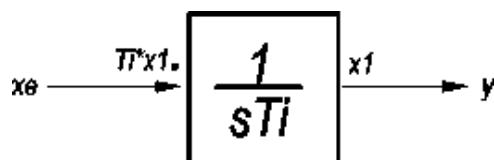


Figure 30.3.3: Graphical Representation of a DSL Model of an Integrator

Figure 30.3.3 depicts the graphical representation of the embedded primitive DSL model. This primitive

model is included in the hydro turbine (in line 2 of the definition of the hydro). The DSL primitive implements a single integrator and is programmed as follows:

```
1. model y = 'I' (xe;x1;Ti;)
2. [Ti] = 's'
3. limfix(Ti) = (0,
4. inc(x1) = y
5. inc(xe) = 0
6. x1. = xe/Ti
7. y = x1
```

Line 1 is generated by clicking on the **Equations** button in the block diagram dialog. Lines 2..7 were entered manually.

The Block Definition dialog was used to set the following:

```
Output Signals : y
Input Signals : xe
State Variables : x1
Parameter : Ti
Internal Variables
```

### 30.3.2.1 Parts of a DSL Model

Both example DSL models show the two basic parts of any DSL model, primitive or complex:

1. The interface definitions
2. The DSL model description

#### Interface description

The interface defines the model name, names of input and output signals, model parameters and state variables. These are shown in the output window in the model heading.

Example (line 1 from the hydro turbine model):

```
1. model Pt = 'pmu_hydro' (At,C;x1;Ti;)
```

The block diagram dialog further allows for the definition of limiting parameters and input signals, and the classification of the model as a linear model and/or as a DSL macro.

#### Model description

The model description describes the DSL model, based on the signals defined in the interface. The DSL description includes:

- Parameter descriptions: name and unit
- Allowed parameter ranges
- Initial conditions and functions which are used to calculate initial values.
- The algebraic relations which define the controller.

Example (the integrator):

```
2. [Ti] = 's' ! the unit of Ti is seconds
3. limfix(Ti) = (0,) ! Ti > 0
4. inc(x1) = y ! initially x1=y
5. inc(xe) = 0 ! initially xe=0
```

```
6. x1. = xe/Ti ! equation 1: deltax1 / deltat = xe/Ti
7. y = x1 ! equation 2: y=x1
```

### 30.3.2.2 Advanced 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.

In addition, inputs and outputs of model parameters, its organisation via windows menus etc. is also derived automatically from the DSL model.

### 30.3.3 Defining DSL Models

A new DSL model is created either by entering the DSL code in the equation part of a *Block Definition (BlkDef)* object, or by creating a new Graphical Block Diagram. Both methods will result in a Block Definition Object which holds the definition of the DSL model.

The Block Definition objects thus serve two purposes in the process of constructing a DSL model:

- They hold the definitions and parts of a graphically constructed composite Block Definition, and the diagram graphic which was used to define the model;
- They provide the surrounding in which a new “DSL primitive” or “primitive Block Definition” can be defined.

#### 30.3.3.1 Composite Block Definitions

To create a new composite Block Definition:

- Use the main menu entry *File → New* or **Ctrl-N** and then select the option *Block/Frame Diagram* from the New command dialog (*ComNew*).
- Use the *Insert New Graphic*  icon on the toolbar (of the graphics window) and select the option *Block/Frame Diagram*.

To access the dialog of the Block Definition (*BlkDef*), double-click on the frame box surrounding the diagram.

Complex Block Definition 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 primitives.

Although a complex Block Definition 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.

The graphical environment in which a complex block diagram is constructed, is not covered here. Refer to Chapter 9 (Network Graphics (Single Line Diagrams)) for more information.

### 30.3.3.2 Primitive Block Definitions

To create a primitive DSL Block Definition:

- Right-click on or inside a (library) folder in the active project in the Data Manager and select *New... → Block/Frame-Diagram* from the context-sensitive menu;
- Use the *New Object*  icon in the database manager and select Block Definition (BlkDef);
- Double-click a new/empty block reference in an open block diagram and then use the  button to select a Block Definition. Following this, use the  icon to create a new Block Definition inside the local library.

DSL primitives are the building blocks from which the more complex controller diagrams are composed. A DSL primitive, for example, might implement a low pass filter, which may then be used to graphically construct more complex controllers which include this kind of filter.

Unlike transformers or other power system components, which may be looked upon as "power system primitives", a DSL primitive is only referred to by a complex block diagram and may thus be used in more than one complex DSL model at the same time.

### 30.3.3.3 Block Definition Dialog

When creating a primitive DSL model or by double-clicking on the frame of a composite Block Definition, a dialog will appear, where input and output variables, parameters, state variables and limiting signals can be defined. Furthermore, additional equations, initial conditions of variables as well as names and units of parameters can be inserted.

#### ***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 Block Definition 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 30.6.1 (C Interface).
- *MATLAB M-file model:* Enabling the "Matlab" tag will show an input dialog, where a *MATLAB (\*.m)* file can be defined with which the Block Definition can communicate during a simulation. For more information about the *MATLAB* interface see Section 30.6.3 (MATLAB Interface).
- *Macro flag:* Check this flag if the block definition is meant to be a DSL macro (refer to Section 30.4.9 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 block definition is shown in the block diagrams.

*Variables pane:*

- *Input and output signals* have to be defined for internal use inside the Block Definition. 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 common model dialog and can then be specified. The parameter defined in the Block Definition will automatically be inserted in the Block Reference. The names of the parameters can be different in the Block Reference and in the Block Definition. Only the order must be identical.
- *Internal variables* are only used inside the Block Definition 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.

*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. In case the two checksums are different, a corresponding information message will be shown.

### **Basic Options page - Advanced tab**

*Initialisation options pane:*

- *Automatic calculation of initial conditions*: *PowerFactory* can calculate the initial conditions automatically. However, if no sequence is found (because of, for example, deadlock situations) there will be an error message.
- *Partial initialisation in case of deadlock*:
- *Configuration Script*: A configuration script (DPL based) can be referenced here. The script should reside within the Block definition object (*BlkDef*) and will be run during the execution of the *Calculation of Initial Conditions* command (*ComInc*). More information on its usage is provided in Section [30.4.6.2](#).

*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 [30.3.1](#). 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 common models, the level of the referenced block definition will decide the model behaviour both at runtime as well as for the syntax checks. Consequently, the block definition which makes use of the specific DSL macro will decide (via its own *Level* parameter) the DSL *Level* of all DSL equations in the model.
- *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.

DSL Level	Description
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 <code>lim</code> function being precise in time
4	DSL Level 3 plus <code>event</code> function using boolean expression as condition: <code>event(boolean expression, ...)</code> , see Section 30.5.2. DSL Level 4 has been introduced with <i>PowerFactory</i> version 13.2.
5	DSL Level 4 plus additional DSL syntax checks and improved warnings. DSL Level 5 has been introduced with <i>PowerFactory</i> versions 15.2.9, 2016 SP4 and 2017.
6	DSL Level 5 plus newly developed additional DSL functions <ul style="list-style-type: none"> <li>• <code>picontrol</code> with variable non-windup limits,</li> <li>• <code>reset</code>,</li> </ul> and improvements in behaviour of DSL functions <ul style="list-style-type: none"> <li>• <code>loopinc</code>, <code>intervalinc</code> and <code>newtoninc</code>,</li> <li>• <code>limstate</code> and <code>limstate_const</code>,</li> <li>• <code>picdro</code> and <code>picdro_const</code>,</li> <li>• <code>picontrol_const</code>,</li> <li>• and all DSL functions having min/max limits (internal check of max not being smaller than min).</li> </ul> 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 has been introduced with <i>PowerFactory</i> version 2020.

Table 30.3.1: List of DSL levels

*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 available 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 available 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 30.6.1 (C Interface).

#### **Block Definition dialog - buttons**

There are several buttons on the right side of the dialog:

*Check:*

"Check" will verify the model equations and output error messages if errors have occurred. Otherwise the following message will occur:

Check '\TestUser.IntUser\Windparks.IntPrj\Library

```
\Block Definitions\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 the current Block Definition DSL equations (including the equations in the used macros) to the output window.

#### *Pack:*

Pack will copy all DSL models (macros) used by a composite model definition to the folder “Used Macros” inside the Block Definition. In this way there will now be references to other projects or libraries outside the model. Beware: any further changes in the macro library have no influence; the macros are copied and no longer linked to the library. Therefore, if an error occurs in a specific macro it must be fixed separately in each packed block.

#### *Pack-> Macro:*

This command will reduce the entire model (including DSL blocks and additional equations and macros) into one DSL model containing only equations. All graphical information will be lost. It should be noted that this command is irreversible.

#### *Encrypt:*

The *Encrypt* button is available after the *Pack → Macro* command has been successfully executed. The *Encrypt* command encrypts all equations inside the model, such that the equations are no longer available to the user. This way, a model can be shared with any third party without disclosing any confidential information that may be contained within its internal structure. It should be noted that this command is irreversible and that no decrypt function is available.

---

#### **Note: Encrypting a DSL Model**

To encrypt a DSL model:

- 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 do the encryption of the model;
  - Activate the project;
  - Open the dialog window of the block definition you want to encrypt;
  - On the right side press **Pack**: this copies all used macros into a sub folder of the block definition;
  - Then press **Pack->Macro**: this removes the graphics and packs all equations into one block definition object. Macro references are removed afterwards.
  - Afterwards, **Encrypt** should be active and can be press to encrypt the model.
  - In addition it is recommended to delete the Graphic folder of the Frame to hide the graphical information.
- 

#### **Block definition dialog - Equations page**

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 30.3.4 shows the additional equations for the DSL model of the basic first-order limiter block.



Figure 30.3.4: Block Definition dialog, *Equations* page

#### **Block definition dialog - Description page**

A general purpose Description page is available for each Block definition.

#### **Block definition dialog - Version page**

A Version page is available for each Block definition.

*Copyright* field: - This field allows to set a watermark/prove the ownership of a model. The field is password protected and can be customized 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 block definition 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;
- Block definition 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).
- Block definition objects which are imported from a *PowerFactory* version equal to or greater than *PowerFactory* 2019 inherit the *Copyright* field information.

The global 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.

## 30.4 The *DIGSILENT* Simulation Language (DSL)

The DSL language is used to program models for electrical controllers and other components used in electric power systems. As for any other simulation or programming language, a special syntax is provided for the model formulation. This syntax is explained in the following sections.

### 30.4.1 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 [30.5 \(DSL Reference\)](#).
- 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'
```

### 30.4.2 General DSL Syntax

**Line length:** The maximal 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 remaining line to be interpreted as a comment. Comments are removed 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)
```

### 30.4.3 DSL Variables

A DSL model may use five different types of variables:

**Output signals:** Output signal variables are available as input signals to more complex DSL models.

**Input signals:** 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.

**State variables:** State variables are time-dependent signals generated and used within the DSL model itself.

**Parameters:** Parameters are ‘read only’ numbers which are set to alter the behaviour of the DSL model.

**Internal variables:** Internal variables are defined and used in the DSL model to ease the construction of a set of DSL equations.

The following rules may be helpful when interpreting warning and error messages:

- A state variable may not be simultaneously used as a state variable *and* an output variable; if required, the use of an assignment like  $y=x1$  is recommended.
- All parameters are real numbers.
- A special parameter “array\_iiii” (with up to 4 digits i), with  $2^*iiii$  elements is provided to define characteristics (see procedure “lapprox”).
- Only the derivatives of state variables can be assigned an expression.

### 30.4.4 DSL Structure

DSL models are constructed in three parts:

- The interface part, which states the model name, title, classification and variable set. This part is set in the first page of the block diagram dialog;
- Definition code;
- Equation code.

The definition and equation code form the actual controller network definition and are treated in the next sections.

### 30.4.5 Definition Code

**Definition code** in the equation part of a DSL model is used to define parameter properties and initial conditions.

#### 30.4.5.1 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 wide.

*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
```

#### 30.4.5.2 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 “)” braces exclude the minimum or maximum value from the interval; the “[” and “]” braces 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 initiali-

sation. Its usage is encouraged for performance reasons whenever `varnm` is constant throughout the simulation.

### 30.4.6 Initial Conditions

#### 30.4.6.1 Direct Setting of Initial Conditions

##### `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`.

#### 30.4.6.2 Configuration scripts for initialisation

A DPL based *Configuration Script* can be optionally added to any DSL model. 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.

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 30.4.1. It can be set up by following these steps:

- Creating a DPL script within the Block Definition (`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 Block Definition (`BlkDef`) object, using the corresponding field

“Configuration Script” within *Basic Options page* → *Advanced tab* of the Edit dialog of the Block definition object.

- Creating a new DSL common 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 common model contains a DPL script that references the remote script of the Block Definition. 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 is sharing the same Block Definition. 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*.

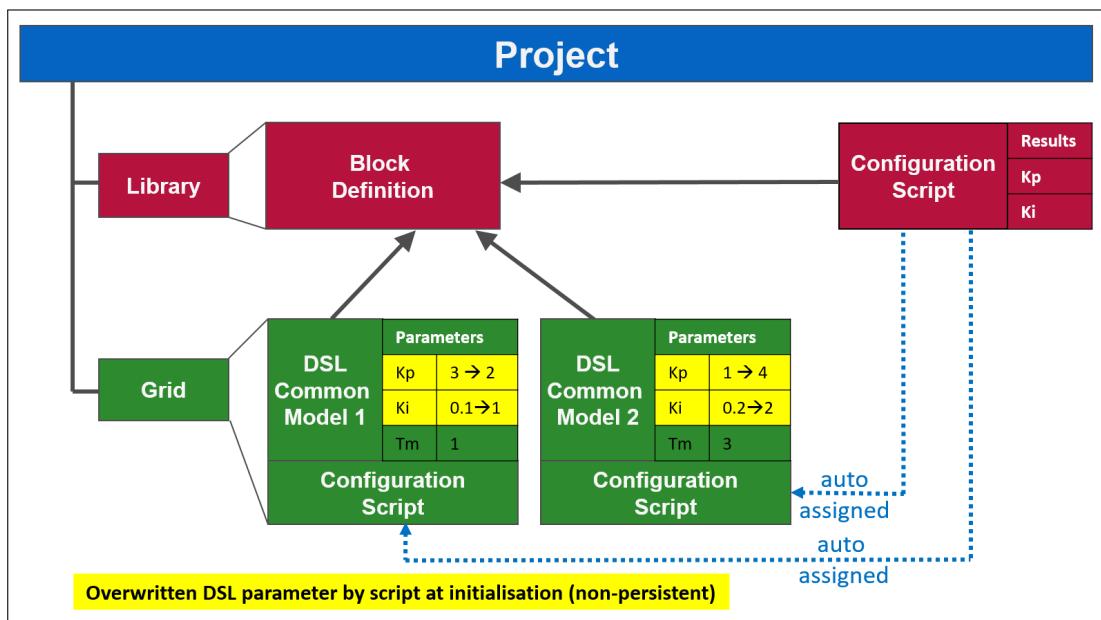


Figure 30.4.1: Configuration Script functionality and inter-operation

#### Notes:

- The scripting language used for the Configuration Script is *DIGSILENT 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 common 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;
  - Switch to the page “Simulation RMS” or “Simulation EMT” (depending on the case), choose the *Variable Set* to be “Calculation Parameter”;
  - 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 30.4.1 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 Common 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 common model may be configured with individual values for the two parameters. Consequently, the parameters *Kp* and *Ki* of “DSL Common Model 2” will have been assigned different values. Note also that, at the end of the simulation, all modified parameters are automatically set back to their initial values.

#### 30.4.6.3 Iterative calculation of initial conditions

Three functions are available for determining initial values iteratively: `loopinc`, `intervalinc`, `newtoninc`.

These functions 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, described in Section 30.6.1.

---

##### **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

s = 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)**

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 value for the parameter whose initial value is sought

iter = the maximum allowed number of iterations

eps = the maximum allowed absolute error between initexpr and the parameter whose initial value is sought.

Example:

```
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)
```

This example shows 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 (like 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.

#### 30.4.6.4 Initialisation logic for multiple DSL blocks containing inter-dependent variables

Complex initialisation procedures involving multiple DSL blocks are supported.

#### 30.4.7 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 transfer functions between the input and output signals. These transfer functions may range from simple linear, single-input single-output functions, to highly complex non-linear, non-continuous, multi-input, multi-output functions.

DSL is used to describe the direct relationships 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 thus split up into a set of single order equations by the introduction of additional state variables.

#### 30.4.8 Equation Statement

The equation statements are used to assign expressions to parameters, thus relating all parameters 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 variables are of type floating point, even if assigned to a boolean expression, in which case the value will be 0.0000 or 1.0000.
- When a variable z is used in a logical expression (i.e.  $y = \text{not}.z$ ), the logical 1 of z is tested by evaluating ( $z > 0.5$ ):

$y1 = \text{not}.z$  is interpreted and equal to  $y1 = (z < 0.5)$

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
```

**30.4.9 DSL Macros**

A DSL macro is a predefined DSL model, complex or primitive, which is meant to be included in higher level DSL models. The block diagram edit dialog offers a 'Macro' classification option which can be set to mark the model as a macro.

A DSL macro is included in a higher level DSL model either by creating a block reference in the block diagram graphics or by its explicit inclusion in a DSL equation.

Syntax:

***varnm1, varnm2,... = macroname(i1, i2,...; s1, s2,...; p1, p2,...; i1, i2,...)***

Assigns the output signals of the DSL macro macroname to the variables *varnm1*, *varnm2* , ... Assigns the input signals of DSL macro to the variables *i1*, *i2*,... The macro uses the state variables *s1*, *s2*,... the parameters *p1*, *p2*,... and the internal variables *i1*, *i2*,...

Example: *P1, P2 = '\User\I.BlkDef' (i1,i2;s1,s2;T1,T2)*

This example assigns to P1 and P2 the output of DSL model *User\I.BlkDef*.

Macro calls are not supported within expressions, even if they only have one output variable.

Correct example:

```
y = my_macro(x1, s1, p1, i1) !
```

Incorrect example:

```
y = 3 * my_macro(x1, s1, p1, i1) + 4
```

which should be replaced by:

```
y1 = my_macro(x1, s1, p1, i1) y = 3 * y1 + 4
```

### 30.4.9.1 DSL Internal 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.

### 30.4.9.2 DSL Models

In general, there are two basic types of DSL models possible:

1. Models of electrical devices such as generators, loads or HVDC systems. These models are characterised by their principal output signal “complex device current”, which is injected to the electrical grid at a certain busbar. However, in addition to the electrical device currents, there may be any other variable defined as an output signal. A summary of the available variables of each element can be seen in the [Technical References Document](#).
2. Models with output signals which are not directly injected to the electrical network (general devices). Among these types of models are prime mover units, voltage controllers, relays, calculation procedures, etc.

### 30.4.10 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 [30.5.2](#).
- **reset(*var*, *rst*, *val*)** generates DSL model internal events (user defined). Further information is found in Section [30.5.2](#).
- **output(*boolexp*, *message\_string*)** outputs a message during the simulation. Further information is found in Section [30.5.2](#).

---

**Note:** The events are accessed or created by opening the edit dialog of the common 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).

---

### 30.4.11 DSL Modelling Examples

#### 30.4.11.1 Integrator Model in DSL

Continuous equation:

$$y_{out} = y_0 + \int y_{in} dt \quad (30.6)$$

Rewriting the continuous equation using the state variable x:

$$\frac{d}{dt} y_{out} = y_{in} \quad (30.7)$$

$$\begin{cases} \frac{d}{dt} x = y_{in} \\ y_{out} = x \\ inc(y_{out}) = y_0 \end{cases} \quad (30.8)$$

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
```

### 30.4.12 Example of a Complete DSL Model

#### Thermal Double Reheat Turbine with Steam Storage

Controller Model:

```
model pt,ptmw =
'pmu_1 '(at,sgn,cosn,ngnum;x1,x2,x3,x4;Thp,
Tip,Tlp,alflp,Tspi)
[T1] = 's'
limfix(T1) = [0,)
limfix(alflp) = [0,1]
vardef(alflp) = ;'High pressure turbine ratio';
limfix(alflp) = [0,1-alflp]
vardef(alflp) = ;'Low pressure turbine ratio ';
vardef(Tspi) = 's';'Boiler capacity time constant';
limfix(Tspi) = (0,)
vardef(Thp) = 's ';'High pressure turbine time constant ';
vardef(Tip) = 's ';'First reheater time constant ';
vardef(Tlp) = 's ';'Second reheater time constant '

inc(x1) = y/K
inc(xe) = y/K
inc(x4) = 1.0
inc(at) = pt
```

```

inc(steamflow0) = pt
inc(ylp) = pt
x1. = selfix(T1>0, (xe-x1)/T1, 0)
y = K*selfix(T1>0, x1, xe) ! if T1=0 => y=xe
steamflow = at*x4
x4. = (steamflow0 - steamflow)/Tspi ! boiler
yhp = PT1(steamflow;x1;Thp) ! high pressure part
yip = PT1(yhp;x2;Tip) ! medium pressure part
ylp = PT1(yip;x3;Tlp) ! low pressure part
pt = yhp*alfhp + ylp*alflp+ yip*(1.0-alfhp-alflp)
ptmw = pt*sgn*cosn*nge ! only for output purposes

```

The used macro 'PT1' is defined as:

```

model y = 'PT1'(xe;x1;K,T1,)
x1. = selfix(T1>0, (xe-x1)/T1, 0)
y = K*selfix(T1>0, x1, xe) ! if T1=0 => y=xe
inc(x1) = y/K
inc(xe) = y/K
[T1] = 's'
limfix(T1) = [0, )

```

## 30.5 DSL Reference

### 30.5.1 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>	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>	$\pi = 3.141592\dots$	$\text{pi}() = 3.141592\dots$
<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 30.5.1: DSL Standard Functions

**Remark:**

- Names of DSL standard functions are reserved. Note that DSL model variables may not have any reserved name.

### 30.5.2 DSL Special Functions

DSL Function	Short description
<code>aflipflop</code>	“Analog” flip-flop function
<code>balanced</code>	Function returning the balanced/unbalanced network representation
<code>delay*</code>	Delay function
<code>event</code>	Multi-purpose simulation event function
<code>flipflop</code>	Logical flip-flop function
<code>gradlim_const</code>	Gradient limiter function
<code>invlapprox</code>	Inverse lapprox function
<code>lapprox</code>	Linear approximation based on a single dimensional array
<code>lapproxext</code>	Linear approximation based on a single dimensional array (function extension)
<code>lapprox2</code>	Linear approximation based on a two dimensional array
<code>lastvalue*</code>	Function returning the last valid value of a signal
<code>lim_const</code>	Nonlinear limiter function with constant limits
<code>lim</code>	Nonlinear limiter function
<code>limfix</code>	Function used to print a warning message if a parameter is outside limits at initialisation
<code>limits</code>	Function used to print a warning message if a parameter is outside limits
<code>limstate_const</code>	State variable limiting function with constant limits
<code>limstate</code>	State variable limiting function
<code>movingavg*</code>	Moving average filter implementation
<code>output</code>	Function used for generating output window messages
<code>picdro_const</code>	Logical pick-up/drop-off function with constant pick-up/drop-off arguments
<code>picdro</code>	Logical pick-up/drop-off function
<code>piccontrol_const</code>	Function used in PI controller implementations with constant non-windup limits
<code>piccontrol</code>	Function used in PI controller implementations with variable non-windup limits
<code>reset</code>	Function used to reset state variables and constant internal variables
<code>rms</code>	Function returning the RMS/EMT simulation type
<code>sapprox</code>	Spline approximation based on a single dimensional array
<code>sapprox2</code>	Spline approximation based on a two dimensional array
<code>select_const</code>	Conditional function with constant true/false expressions
<code>select</code>	Conditional function
<code>selfix_const</code>	Conditional function applied to constants at initialisation with constant true/false expressions
<code>selfix</code>	Conditional function applied to constants at initialisation
<code>time</code>	Returns the current simulation time
<code>vardef</code>	Function assigning a text description and unit to a specific variable
<code>file</code>	Obsolete function
<code>fault</code>	Obsolete function
<code>mavg</code>	Obsolete function

Table 30.5.2: 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 [30.3.1](#).
  - 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(t)* 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 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 [29.5](#) 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 (common 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 common 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 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 common 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 common 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 changes. 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)
```

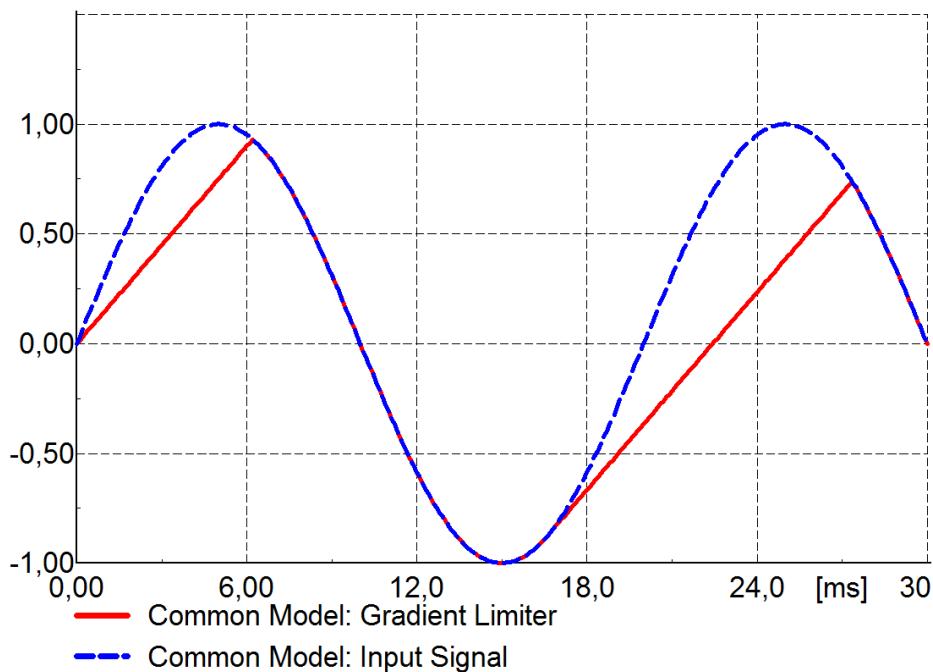


Figure 30.5.1: Gradient limiter example

## invlapapprox

[\[back\]](#)

***invlapprox (y, array\_iiii)***

***invlapprox (y, oarray\_iiii)***

Inverse lapprox with array.

**Arguments:**

- *x* - 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 common model. The *oarray\_iiii* is used if the lookup array is to be defined externally of the DSL model, using *IntMat* objects.

**Return value:** output *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 common 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 common model (n.b. not within the block definition) 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 common 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 common model (n.b. not within the block definition) 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 common 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 common 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 common model which defines variable “oarray\_K”. A combination of internal and external arrays (i.e. mixing array\_ and oarray\_ variables) within a single block definition / common 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 common 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 common 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 block definition / common 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:  $y = yval(\text{first point}) + xe - xval(\text{first point}) * dy/dx$  (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 common 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 initialization.

***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(boolexpr) 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

The outfix(*boolexpr*,...) function is a variant of the output(*boolexpr*,...) function which is evaluated only at initialisation. Its usage is encouraged for performance reasons whenever *boolexpr* evaluates to a constant throughout the simulation.

#### **Arguments:**

- boolexpr* - Expression to be evaluated as true or false.
- message\_string* - String delimited by single quotes

**Return value:** None.

**picdro\_const**

[\[back\]](#)

#### **picdro\_const(*boolexpr*, *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:**

- boolexpr* - 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 *boolexpr* = 1 (i.e. true), for a duration of at least *Tpick* seconds
- changes from 1 to 0, if *boolexpr* = 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 (*boolexpr*, *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 30.5.2 and described in IEEE 421.5 (2005) standard. In the figure,  $K_p$  and  $K_i$  are constant parameters and `input` is an input signal. Its usage is encouraged (as opposed to [piccontrol](#)) for simulation performance reasons whenever the `min` and `max` arguments evaluate to time-independent constants.

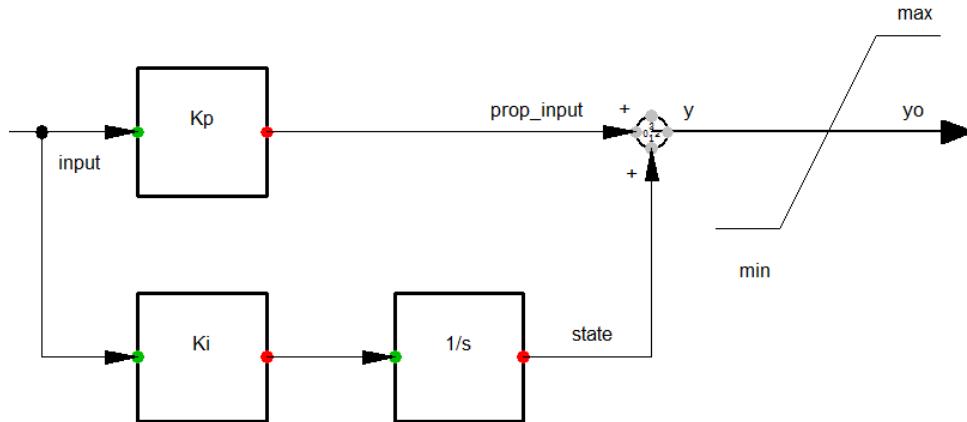


Figure 30.5.2: PI Controller Structure acc. to IEEE 421.5

#### **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 * Kp`)

**Return value:** `yo`, calculated as below:

$$yo = \begin{cases} \text{min}, & \text{if } (\text{prop\_input} + \text{state}) < \text{min}. \\ \text{max}, & \text{if } (\text{prop\_input} + \text{state}) > \text{max}. \\ \text{prop\_input} + \text{state}, & \text{otherwise.} \end{cases}$$

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 30.5.2, 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 30.5.2 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 30.3.1 for more information). For lower levels use [event](#) function instead.
- It is allowed to use multiple *reset* functions targetting 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 common 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(boolexpr,x,y)**

Fixed switch function, returning  $x$  throughout the simulation if  $boolexpr$  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:**

- $boolexpr$  - 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* throughout the simulation if *boolexpr* is true at initialisation, otherwise *y*.

**Example:**

```
xramp.=selfix_const(T1>0, 1/T1, 0.0) !to avoid division by zero
```

**selfix**

[\[back\]](#)

**selfix (*boolexpr*, *x*, *y*)**

Fixed switch function, returning *x* throughout the simulation if *boolexpr* is true at initialisation, else *y*. 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* throughout the simulation if *boolexpr* 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 block definition.

**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.

### 30.5.3 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.

## 30.6 Interfaces for Dynamic Models

### 30.6.1 C Interface

*PowerFactory* provides the possibility of compiling DSL Block Definitions into C source files. The 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. The 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 'C models'. Currently, the Microsoft®Visual Studio 2012 IDE is supported. *PowerFactory* provides compiled models of 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').

#### 30.6.1.1 C Interface Usage

Any DSL Block Definition may be compiled into a DSL C Interface model. As with any DSL model, it is recommended that the *Calculation of Initial Conditions* of a DSL common model representing the DSL Block Definition 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 Block Definition; and
2. Compilation of source files into a DLL using an external compiler.

To create the C source files for an existing DSL Block Definition (which must be a non-macro, highest-level Block Definition), the following steps are required:

- Open the *Advanced* tab of the Block Definition dialog as shown in Figure 30.6.1;
- 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').

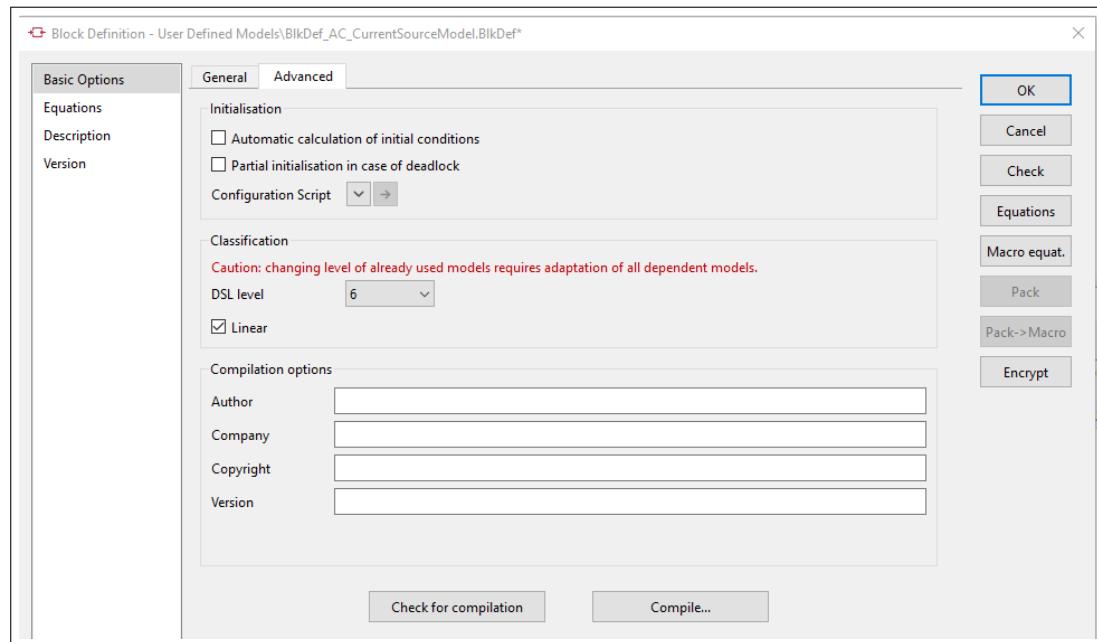


Figure 30.6.1: Block Definition dialog, *Basic Data* page, *Advanced* tab

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 Block Definition 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 correct 32-/64-bit architecture is selected when building the DLL in order to comply with *PowerFactory* requirements as follows:

- A 64-bit *PowerFactory* version requires a 64-bit compiled DLL;
- A 32-bit *PowerFactory* version requires a 32-bit compiled DLL.

After creating the DLL, the file can be readily used in *PowerFactory*. On the *General* tab of the Block Definition 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 Block Definition (shown in Figure 30.6.2) 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.

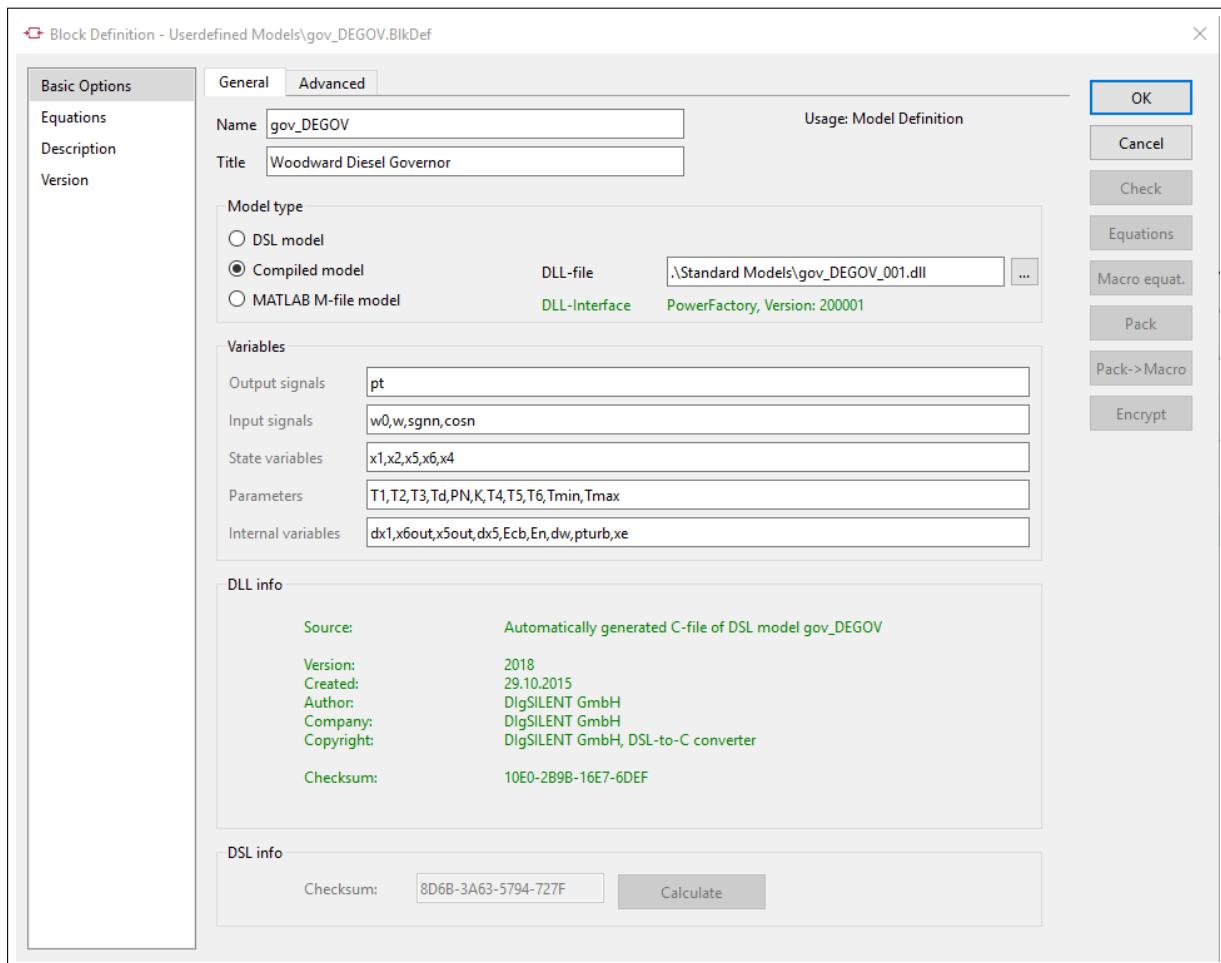


Figure 30.6.2: Compiled model options

The model can be integrated in a Composite Frame in the same manner as a typical Block Definition (as described in Section 30.2.1).

More information about the C Interface including the description of the C Interface Model and examples, is available on the *External C Interface for dynamic models*, which can be accessed from the menu *Help* → *Additional Packages*.

## 30.6.2 External C Interface acc. to IEC 61400-27-1

It is possible to directly interface external models compliant with IEC 61400-27-1 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 Annex F specifies 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 core equations (i.e. binary code) can be used in multiple simulation environments.

### 30.6.2.1 IEC Interface Usage

The user needs to make sure that the correct 32-/64-bit architecture is selected when building the DLL in order to comply with *PowerFactory* requirements as follows:

- A 64-bit *PowerFactory* version requires a 64-bit compiled DLL;
- A 32-bit *PowerFactory* version requires a 32-bit compiled DLL.

As shown in Figure 30.6.3 the user must assign the DLL-file path in the corresponding field of the *Basic Data page* → *General tab* of the Block Definition. *PowerFactory* automatically loads the DLL and displays relevant information. The list of used variables (outputs, inputs, parameters, 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 lists the model description (*StaticExtSimEnvCapi* → *ModelDescription*), the simulation type (*StaticExtSimEnvCapi* → *EMT\_RMS\_Mode*) and the used sample time (*StaticExtSimEnvCapi* → *Fixed-StepBaseSampleTime*). The *Version* field states the model version (*StaticExtSimEnvCapi* → *ModelVersion*). The *Created* field provides the model creation date (*StaticExtSimEnvCapi* → *ModelCreated*). The *Author* field provides the name of the model creator (*StaticExtSimEnvCapi* → *ModelCreator*).

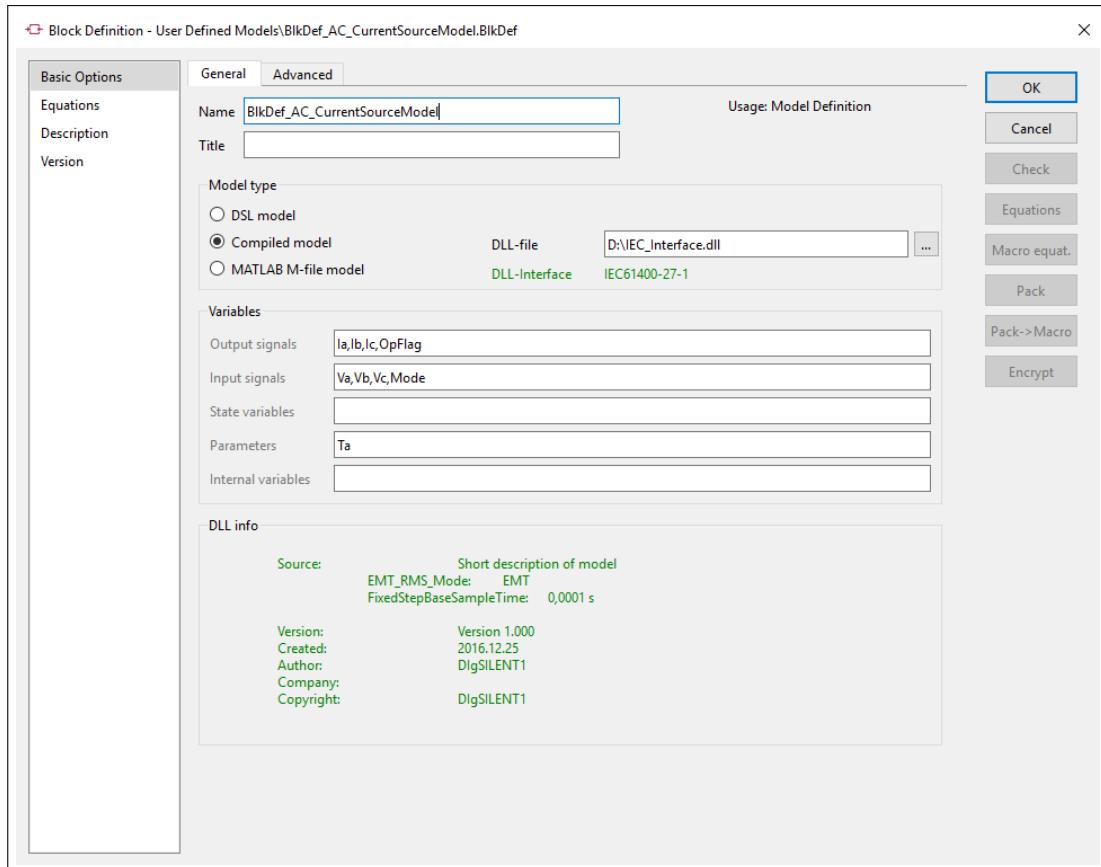


Figure 30.6.3: Compiled model options

The model can be integrated into a Composite Frame in the same manner as a typical Block Definition (as described in Section 30.2.1).

Under *Basic Data page* → *Advanced tab* of the Block Definition the field *Sample Time* is available (with unit in seconds). By default, the value of  $-1$  will ensure that the pre-defined sample time (*StaticExtSimEnvCapi* → *FixedStepBaseSampleTime*) is used in the simulation. A positive value of the field *Sample Time* will override the pre-defined setting and use this specific value as model sample time within the simulation.

**Note:** It is important to note that the calculation of initial conditions procedure in *PowerFactory*'s RMS and EMT simulation is only partly supported by the IEC 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 compliant model. It is important to note that, as with any other externally compiled models (e.g. based on digedyn or digexfun interfaces), the IEC 61400-27-1 compliant models are not supported by the Modal Analysis Toolbox. The only exception is made by compiled models developed based on *DlgSILENT*'s C Interface. (Further information on the DSL to C Interface can be found in section 30.6.1.)

### 30.6.3 MATLAB Interface

In addition to building controller models using the *DlgSILENT* Simulation Language (DSL), it is possible to connect to MATLAB models via a *PowerFactory*-MATLAB interface. A DSL model (object class *BlkDef*) is defined in *PowerFactory* and set up to have a link to a MATLAB .m file. The DSL model must have at least one output and at least one state variable. The MATLAB .m file returns the vector *t* and the matrices *x* and *y*, where *t* represents a time-vector, *x* represents a matrix of state-variable values,

and  $y$  represents a matrix of output values. From these, *PowerFactory* calculates the derivatives of the state variables and outputs. The numerical integration is conducted in *PowerFactory*. *PowerFactory* calls MATLAB with every time step. MATLAB does not run a simulation in parallel with *PowerFactory*, it simulates only one time step and returns two rows in  $t$ ,  $x$  and  $y$ , which correspond to the initial and the final times of that time step. The matrix  $x$  has as many columns as there are state variables, and the matrix  $y$  has as many columns as there are outputs. *PowerFactory* calculates the derivatives and the outputs at the beginning of the time step, and proceeds with the integration. In the following example the initial time is 0 s and the final time is 0.01 s. There are two state variables, and two outputs.

$$t = \begin{bmatrix} 0 \\ 0.01 \end{bmatrix}; x = \begin{bmatrix} 1.02 & 2.1 \\ 1.03 & 1.9 \end{bmatrix}; y = \begin{bmatrix} 10 & 2 \\ 11 & 3 \end{bmatrix} \quad (30.9)$$

*PowerFactory* calculates the derivatives from the time step and the initial and final values of the state variables, e.g., and obtains the outputs of the MATLAB model at the beginning of the time-step from the  $y$ -matrix, e.g.

$$\frac{dx_1}{dt} = \frac{(1.03 - 1.02)}{0.01 - 0} \quad (30.10)$$

, and obtains the outputs of the MATLAB model at the beginning of the time-step from the  $y$ -matrix, e.g.  $y_1 = 10$ .

To use the MATLAB interface, it must be installed on the same computer as *PowerFactory*. When the time-domain simulation in *PowerFactory* is initialised, it will start an instance of MATLAB the same version that was used last.

The following section provides an example of integrating a MATLAB model with *PowerFactory*. In the example a voltage controller is implemented first using a *PowerFactory* model (*ElmVco\_16*) and subsequently using an implementation in MATLAB Simulink. This example can also be found in the **Knowledge base** accessible from the *DlgSILENT* Support area <https://www.digsilent.de/index.php/support.html>.

### 30.6.3.1 Example Implementation of Voltage Controller

In this example the grid consists of two generators, one load and one line, as shown in Figure 30.6.4.

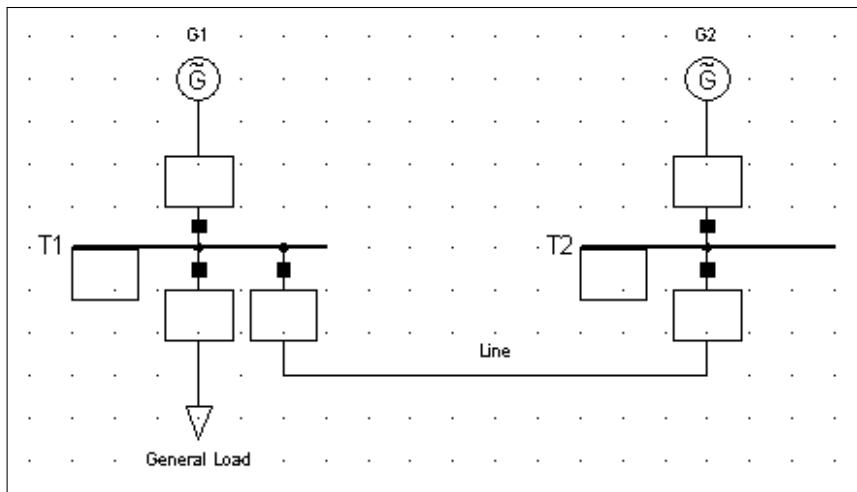


Figure 30.6.4: MATLAB example grid

The simulation event is defined for the load, where the reactive power is increased after 0.5 seconds.

The complete example contains three files:

1. Matlab Example.dz is a *PowerFactory* file.
2. VCOtype16.m is a MATLAB M-file.  
This file is an interface to the Simulink model, and it is used as a middle layer in the communication between *PowerFactory* and Simulink.
3. vcotype16mod.mdl is a Simulink model and contains Simulink implementation of VCO type 16.

### 30.6.3.2 Built-in Model

In the base study case, the voltage controller models are represented by the built-in models VCO type 16 (**ElmVco\_16**). The built-in VCO type 16 inside *PowerFactory* is one excitation control system with simplified exciter. Both composite models use the AVR inside the IEEE-frame from the global library. The generators have different VCO parameters set.

In Figure 30.6.5 the edit dialog of the *ElmVco* with the parameters of the AVR can be seen.

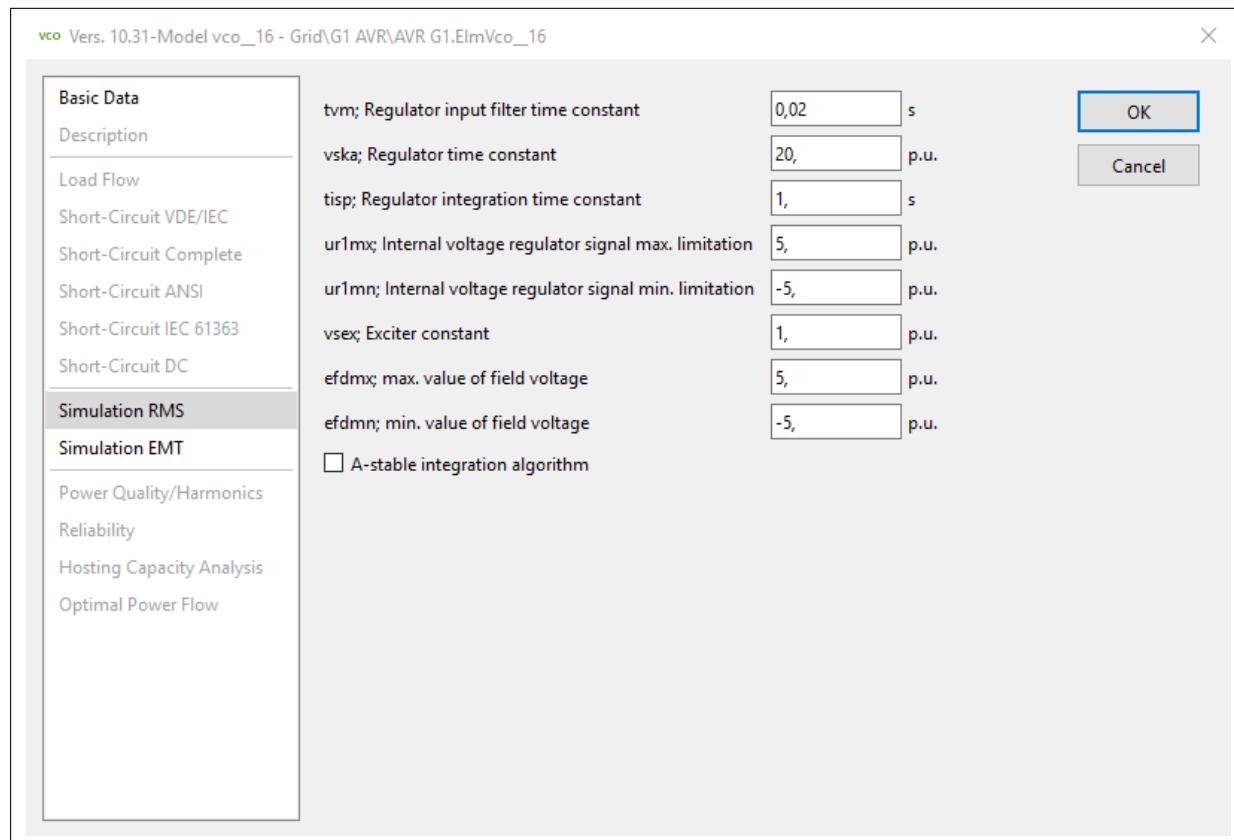


Figure 30.6.5: Parameters dialog of the voltage controller

The model representation of the **ElmVco\_16** is indicated in Figure 30.6.6.

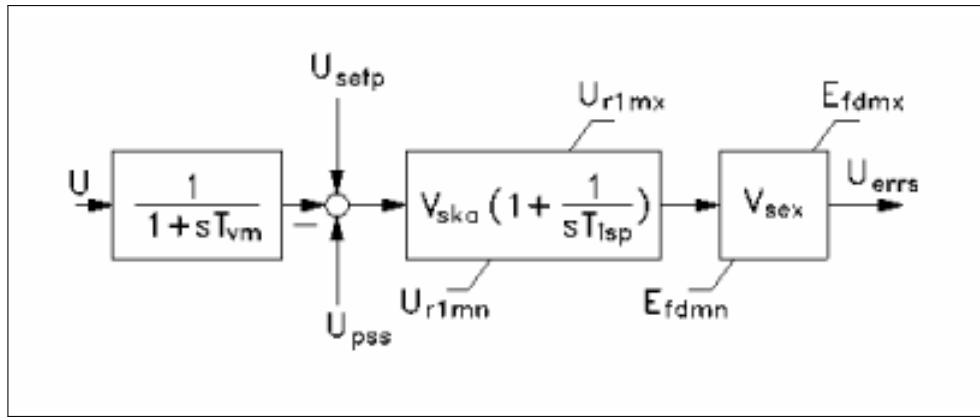


Figure 30.6.6: Parameters plot of the voltage controller

The plots resulting from the simulation (Figure 30.6.11) show busbar voltages and excitation voltage for both generators. The results are stored in results files located under the “Results” folder of the relevant study case.

### 30.6.3.3 MATLAB Model

In the second study case “Matlab” which is a modification of the base case, VCO type 16 is modelled inside the Simulink package, instead of using a built-in model. The MATLAB console is started automatically when running the simulation.

To implement a MATLAB model into a current project in *PowerFactory* it has to be included into a frame similar to a DSL model definition. This procedure is described in detail in the Section 30.6.3 (MATLAB Interface). First a slot inside the frame has to be created, where the controller model should be inserted. This is done exactly like for implementing built-in models or common models. Then a Block Definition *BlkDef* has to be created inside the library. Instead of programming the transfer function using the DSL code, there can now the definition of the MATLAB code be imported.

This can be done in the dialog of the Block Definition. When creating a primitive DSL model in the library by:

- right-clicking a or inside a (library) folder in the active project in the Data Manager and selecting *New... → Block/Frame - Diagram* from the context menu.
- using the *New Object* icon ( in the database manager and selecting *Block Definition (BlkDef)*
- double-clicking an new/empty block reference in an open block diagram and then use the button to select a Block Definition. Then The icon can be used to create a new Block Definition inside the local library.

Now open the dialog of the new *BlkDef*:

- by double-clicking on the frame of a composite Block Definition
- by double-clicking the definition in side the library or on its icon

Here input and output variables, parameters, state variables and limiting signals have to be defined. Instead of inserting the equations to describe the different function blocks, a MATLAB file *\*.m* can be selected, when the option Matlab is activated.

The edit dialog of the Block Definition including the parameter definition and the selected file can be seen in Figure 30.6.7 for the mentioned example.

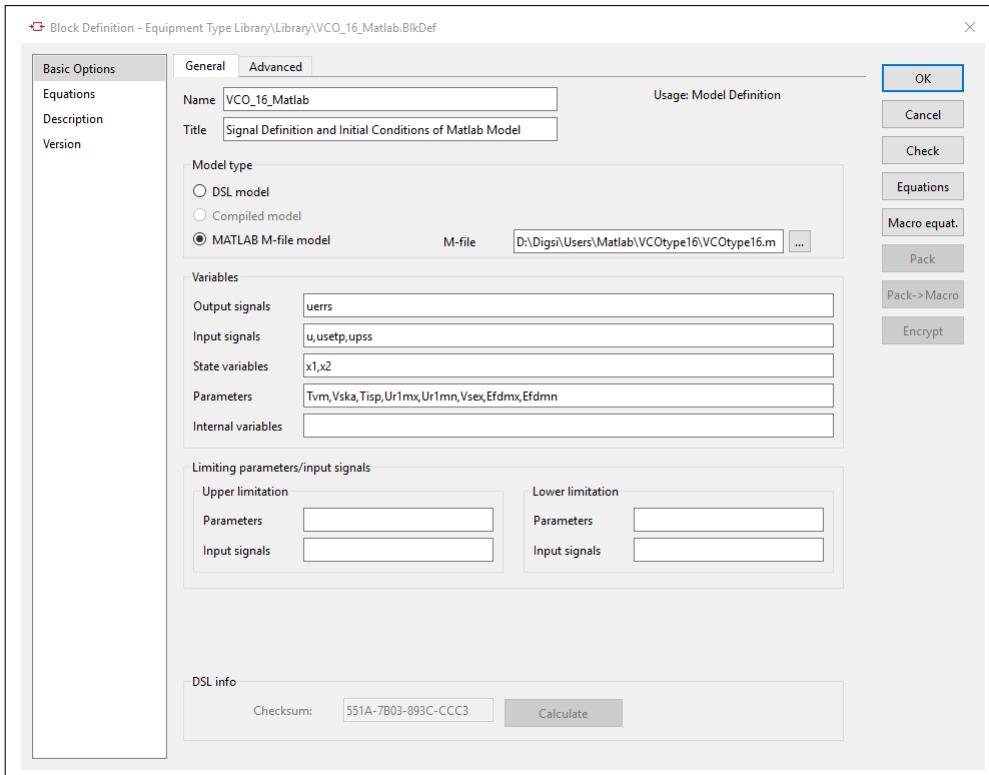


Figure 30.6.7: Composite model using a special frame

The model representation of the **ElmVco\_16** in the MATLAB Simulink package is shown in Figure 30.6.8

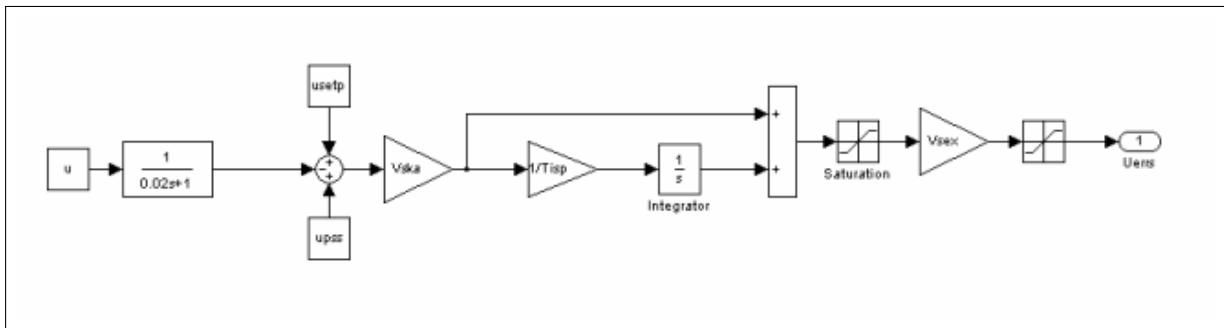


Figure 30.6.8: Parameters plot of the voltage controller

When the Block Definition is specified, a DSL model has to be created first. As described in Section 30.1.3 (The Common Model), the common model element (*ElmDsl*, *dsl*) is the front-end object for all user-defined Block Definitions. This means that all user-defined transient models including built-in elements or MATLAB models cannot be used other than through a common model.

The common model then combines a model or Block Definition with specific set of parameter values. The edit dialog of the DSL element now looks different to the built-in *ElmVco*. From Figure 30.6.9 can be seen, that this dialog is similar to the normal DSL models. All time constants and other parameters are the same as for the built-in VCO models.

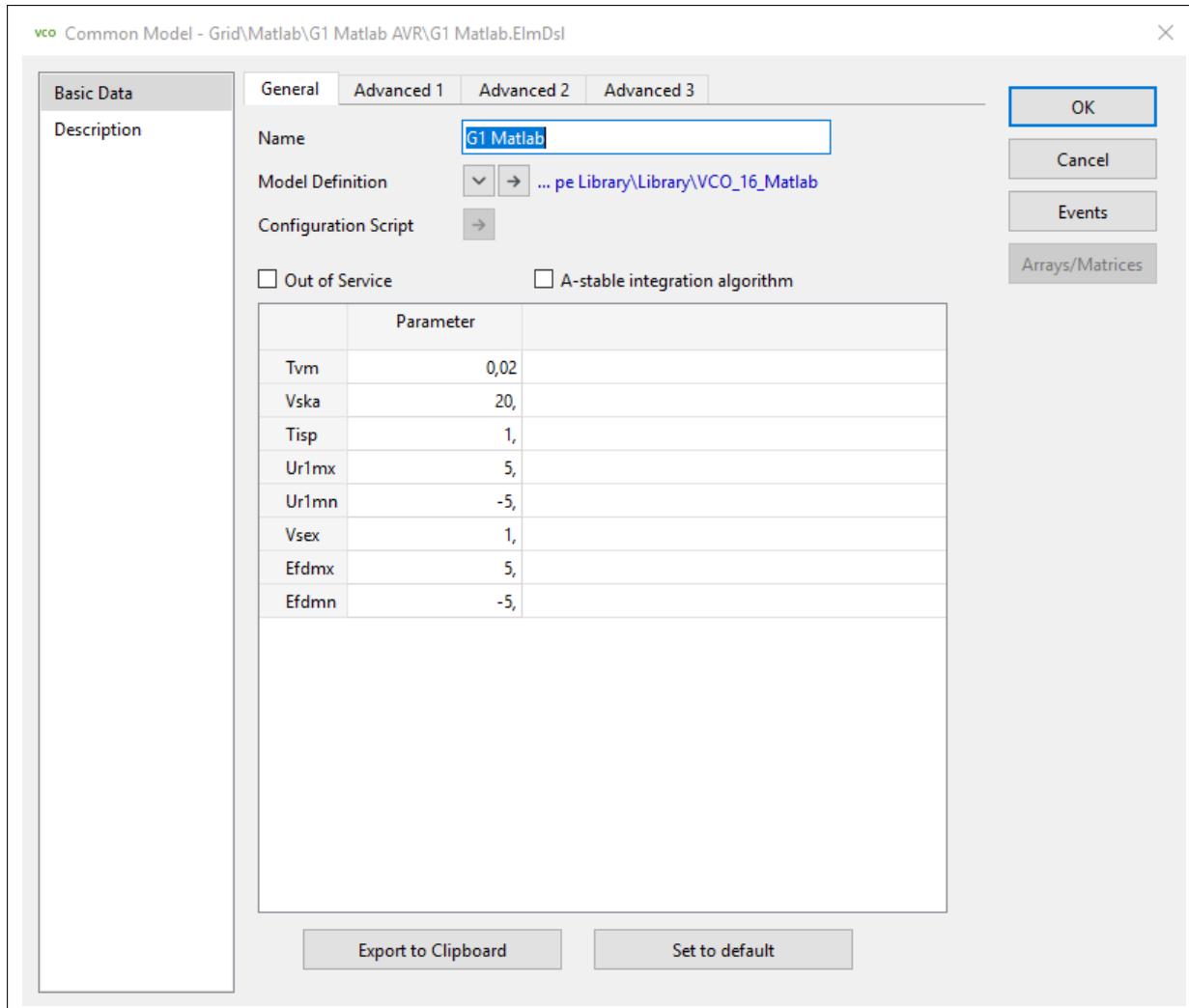


Figure 30.6.9: Parameters dialog of the *MATLAB* voltage controller

Figure 30.6.10 shows the composite model using the special frame with the generator 'G1' and the Matlab-AVR inserted into the slots.

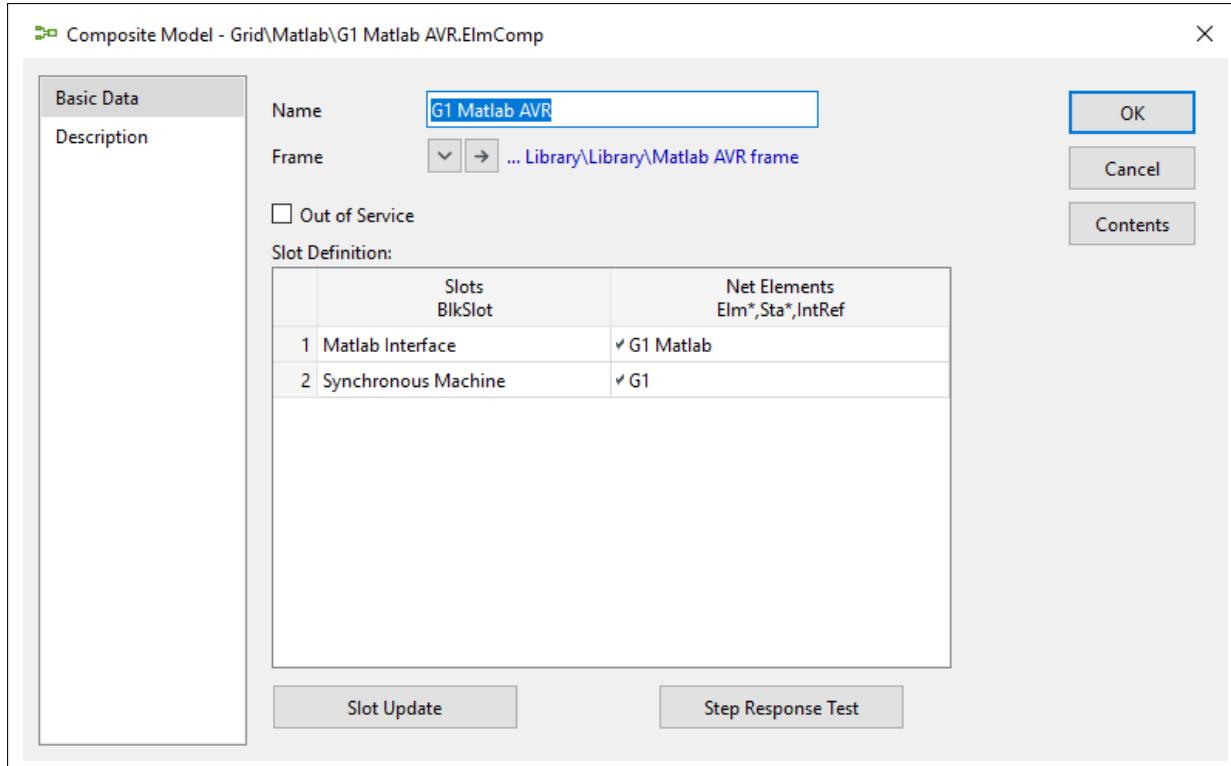


Figure 30.6.10: Composite Model using a special frame

These results from the simulation of the reactive power step using the built-in VCO model (dotted curves) and using the MATLAB representation (solid curves) can be seen in Figure 30.6.11

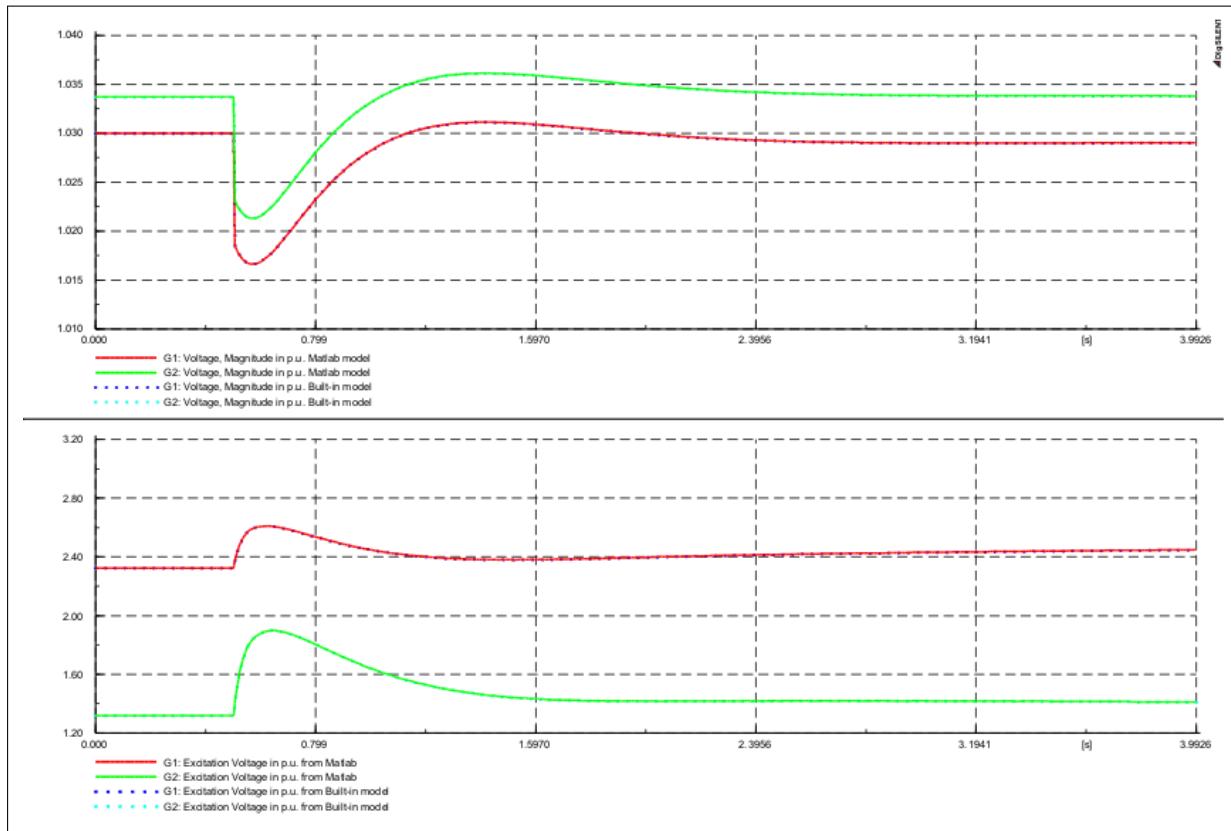


Figure 30.6.11: Results of the transient simulation with the Built-In model

### 30.6.3.4 The MATLAB File

The MATLAB file **VC0type16.m** is an interface configuration for the Simulink model, stored in the file **vcotype16mod.mdl**, and the *PowerFactory* DSL model. There the input and output signals, the parameters and the state variables are defined, as described below. The transfer function is specified.

The contents of this file is listed here:

```
function [t, x, y] = VC0type16
global U Tvm Usetp Upss Vska Tisp Urlmx Urlmn Vsex Efdmx
Efdmn ve1 x1 x2
options = simget('VC0type16mod');
options = simset('InitialState', [x1,x2]);
[t, x, y] = sim('VC0type16mod', [], options);
```

*PowerFactory* inserts the following variables into the *MATLAB* workspace:

```
U, Tvm, Usetp, Upss, Vska, Tisp, Urlmx, Urlmn, Vsex, Efdmx, Efdmn, ve1, x1, x2
```

Those variables are necessary to successfully run the Simulink model. There are three input signals (*U*, *Usetp*, *Upss*), one output signal *Uerr* and two state variables *x1* and *x2*.

In each step of the *PowerFactory* simulation the Simulink model is completely evaluated. State variables ('InitialState') are assigned to Simulink model in each step of the simulation. For *PowerFactory* it is a simple function call:

```
[t, x, y] = VC0type16.
```

*PowerFactory* uses only one Simulink model for both generators. To avoid limitation of Simulink, which allows only one instance of the model running at the same time, *PowerFactory* must send all parameters in the each step of the simulation.

To find appropriate equations for the initial conditions you need to understand the construction of the transfer function blocks in **Simulink**. To obtain this understanding you can replace the variables with actual numbers in the *MATLAB* Simulink model, set the initial conditions, run it for a few seconds and monitor the outputs of all transfer functions to see whether the model initialised correctly.

The *MATLAB* Simulink model (.mdl) and the interface file (.m) file may not have the same name.

The order of the state variables in the interface file's statement "options = simset('InitialState', [x1, x2, .....])" is important; the order of the elements in the vector [x1, x2, ...] must be the same as in the state variable vector constructed internally by *MATLAB*. To determine the order of the *MATLAB* state variable vector the user may use the command "[sizes,x0,xstring]= ModelName" in the *MATLAB* workspace, where ModelName is the name of the Simulink model (without the .mdl extension and without inverted commas). The output of the string variable xstring contains the names of the dynamic blocks in the **Simulink** model in the desired order. In the case of the above example the first state variable is in the measurement block and the second state variable is in the integrator:

```
xstring =
... 'VC0type16_model/Measure/State Space'
... 'VC0type16_model/Integrator'
```

The names of the variables in the 'Initial conditions' fields in the masks of the **Simulink** model dynamic blocks is irrelevant.

The initial conditions are set within *PowerFactory*. Also, for the purpose of *PowerFactory*'s model checking mechanisms, the state derivatives equal to zero

The **Simulink** solver parameters are set to integrate over one small time step, e.g. start time = 0, end time = 0.01, and step size = 0.01.

The y-matrix returned by *MATLAB* contains the output variables. If more than one output variable is defined in the **DSL** model, then those are sorted alphabetically before assigning the outputs from *MATLAB*. For example, if there are two outputs “uerrs” and “output”, then the value from the first column of the y-matrix is assigned to “output” and the value from the second column is assigned to “uerrs”.

### 30.6.3.5 Additional notes

DlgSILENT *PowerFactory* calls MATLAB using the programme identification keys “Matlab.Application” and “Matlab.Application.Single”. *PowerFactory* will start that same *MATLAB* installation which was used last.

Additional information on the calling of *MATLAB* is available on  
<http://www.mathworks.com>

# 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.

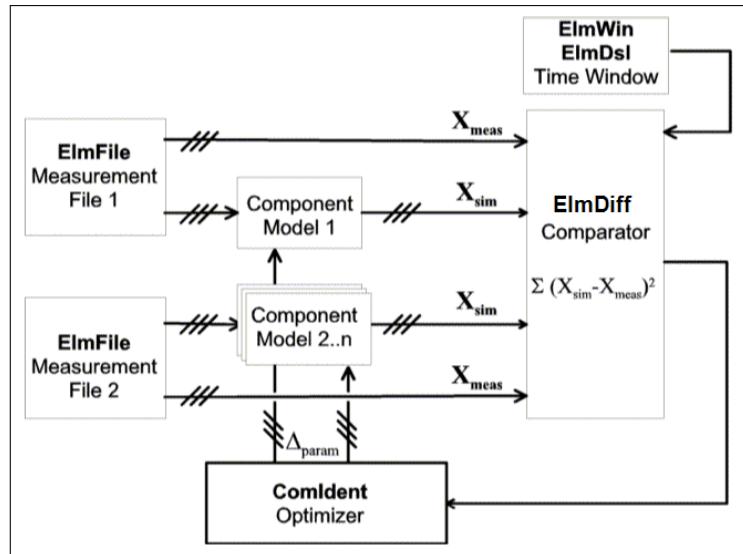


Figure 31.1.1: The identification principle

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* toolbar which is accessed 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 two types supported:

- Load Flow
- 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 (→).

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 (either load flow or dynamic simulation, depending on the selected calculation type). 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 common 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 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* 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

### 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  $\mathbb{R}^d$ , 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 *Comldent* 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 maintainance 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 maintainance 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](#).

**Line searches 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 simplex 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 Optimization Without the Lipschitz Constant  
by Jones, Pertunen and Stuckman,  
(Journal of Optimization 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 emphasize 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:

$$\begin{aligned}
 H_0 &= 1_{d \times d} \quad (\text{identity matrix}) \\
 s_n &= -H \nabla f(x_n) \\
 x_{n+1} &= \text{result of linesearch in direction } s_n \\
 \varrho_n &= \frac{1}{\langle \nabla f(x_{n+1}) - \nabla f(x_n), x_{n+1} - x_n \rangle} \\
 H_{n+1} &= H_n - \varrho_n \left( (x_{n+1} - x_n)(\nabla f(x_{n+1}) - \nabla f(x_n))^T H_n \right) \\
 &\quad + H_n (\nabla f(x_{n+1}) - \nabla f(x_n)) (x_{n+1} - x_n)^T \\
 &\quad + \varrho_n^2 \left( \left\langle \nabla f(x_{n+1}) - \nabla f(x_n), H_n (\nabla f(x_{n+1}) - \nabla f(x_n)) \right\rangle + \frac{1}{\varrho_n} \right) (x_{n+1} - x_n) (x_{n+1} - x_n)^T
 \end{aligned}$$

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 [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/Eigenvalues Analysis command 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 and at every 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 of 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 using numerical, iterative algorithms. The representation of the electrodynamic network model is equivalent to the representation used for the balanced RMS simulation, except for the general load model, for which the frequency dependencies are neglected.

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 several (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 stable.

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-angle
s:psie	p.u.	Excitation-Flux
s:psiD	p.u.	Flux in D-winding
s:psix	p.u.	Flux in x-winding
s:psiQ	p.u.	Flux in Q-winding

Table 32.1.1: Variables accessible for eigenvalue calculation

### When are modal analysis results valid?

A modal analysis can be started when a balanced steady-state condition is reached in a dynamic calculation. Normally, such a state is reached by a balanced load-flow calculation, followed by a calculation of initial conditions. However, it is also possible to do a balanced 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 initial conditions of all elements are calculated first (assuming that the calculation is initialised from a load-flow rather than during an RMS simulation). 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. *PowerFactory* automatically linearises all relevant system elements because eigenvalue calculations need linearised models.

### 32.2.1 Modal Analysis Command - Basic Options

There are two possible calculation methods for the Modal Analysis; the method and its associated options are selected on the *Basic Options* page.

#### 32.2.1.1 QR/QZ-Method

The QR/QZ-Method is the 'classical' method. It calculates all of the system eigenvalues.

The following models are supported by the QZ-Method:

- The Asynchronous Machine (*ElmAsm*);
- The PWM converter (*ElmVscmono*, *ElmVsc*);
- DFIG (*ElmAsmsc*);
- DC machine (*ElmDcm*);
- DC line (*ElmLne* with type set to DC);
- Complex load;
- DC shunt;
- DC surge arrester;
- DC valve (*ElmValve*);
- DC series reactor;

### 32.2.1.2 Selective Method (Arnoldi/Lanczos)

The selective modal analysis also known as the Arnoldi/Lanczos method only calculates a subset of the system eigenvalues around a particular reference point. Often this method is used in very large systems when using the QR/QZ-method could be very time consuming. It is especially useful if the user knows the target area of interest for the eigenvalues. This option needs more configuration as explained below.

#### Complex reference point (RP)

Reference point on the real-imaginary plain for the selective method.

#### 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 32.2.1. 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.

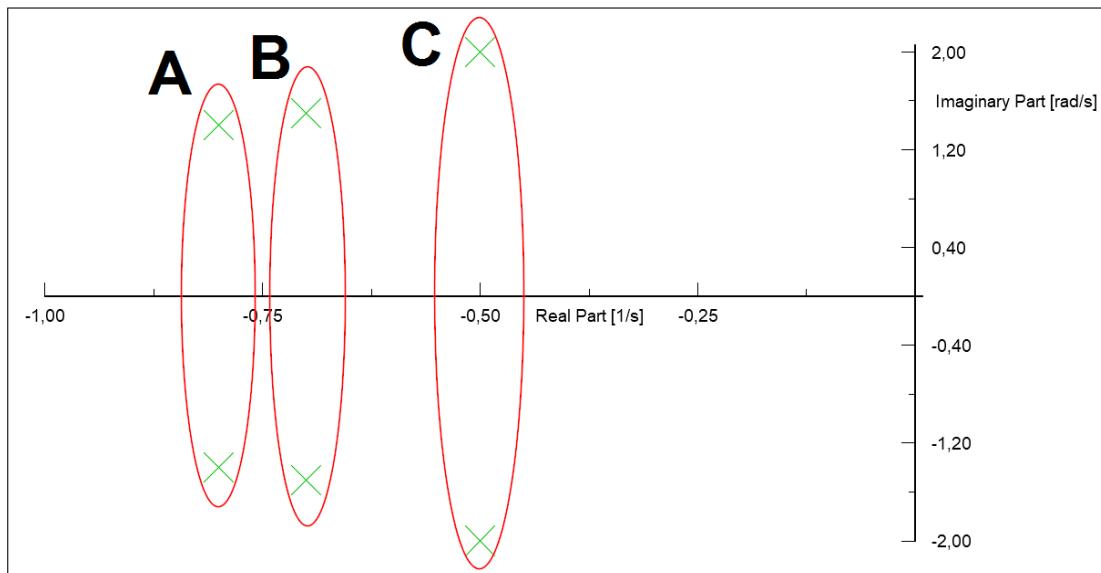


Figure 32.2.1: Illustration of different eigenvalue selection methods

### Number of Eigenvalues

This parameter limits the total number of eigenvalues calculated by the selective method. An eigenvalue pair is defined as *one* eigenvalue mode for this calculation.

#### 32.2.1.3 Initial Conditions

The *Edit* button ( ) is a reference (pointer) to the *Calculation of Initial Conditions* command that is used by the Modal Analysis command, accessed through the button from the Simulation RMS/EMT toolbar.

The eigenvalue analysis is only available with balanced RMS simulation method. More information about the options in the Calculation of Initial Conditions command can be found in Chapter 29: RM-S/EMT Simulations, Section 29.3.

### 32.2.2 Modal Analysis Command - Advanced Options

This section explains the options available on the *Advanced Options* page of the modal analysis command.

#### 32.2.2.1 Calculate

There are three check boxes here:

- 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.

The *Controllability*, *Observability* and *Participation Factors* for any mode can be visualised using the *Mode Phasor Plot* or *Mode Bar Plot* described in Section 32.3.2.

#### 32.2.2.2 Omit Eigenvalues

In this part of the command it is possible to restrict the eigenvalues shown.

- if magnitude is larger than: all eigenvalues of modulus greater than this value will not be shown.
- if magnitude is smaller than: all eigenvalues of modulus smaller than this value will not be shown.

#### 32.2.2.3 Initialisation of Arnoldi Iteration

This selection is active if the Calculation Method is set to *Selective Method*. According to this selection the start vector of the iterative algorithm is chosen. The user may choose to use a randomly chosen vector or a standard unit vector to initialise the Arnoldi algorithm.

### 32.2.2.4 Algorithm

This selection is activated if the QR/QZ-Method is selected.

**QR method using system reduction:** this method uses a system reduction in order to solve a standard eigenvalue problem by the QR method. All models including the selection of elements in section [32.2.1.1](#) are supported by this calculation.

**QZ method for generalised system:** this method solves the generalised eigenvalue problem directly by using the QZ method. Again, there are no restrictions on the models in the system. It may be used in rare cases when the system reduction method described above fails.

### 32.2.2.5 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 [30.6.1](#), 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 [30.5.2](#). 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.

### 32.2.2.6 Advanced tab

The options are available only when *Selective Method (Arnoldi/Lanczos)* is chosen.

- Identification of Eigenvalues: consider identical if distance smaller. Left and right eigenvectors are calculated separately in *PowerFactory*. This parameter specifies the tolerance for eigenvalues when its eigenvectors are associated.

If the QR/QZ-Method is chosen, the user can set the check box to directly construct the A Matrix as in version 13.2. This option is from an older version of *PowerFactory* and we suggest that it should no longer be used.

## 32.2.3 Modal Analysis Command - Output

The matrices used for the Modal Analysis can be exported to a Matlab readable file format (sparse matrix). The user can select to which the Matlab files 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  $\mathbf{J}$  ( $n + m$  by  $n + m$ ) 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 **A<sub>mat</sub>**: file **Amat.mtl**;
- System eigenvalues: file **EVals.mtl**;
- Left eigenvectors: file **IEV.mtl**;
- Right eigenvectors: file **rEV.mtl**;
- Jacobian matrix **J**: File **Jacobian.mtl**;
- **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 **J** and the system **A<sub>mat</sub>** matrices is provided within two text documents named **VariableTolidx\_Jacobian.txt** and **VariableTolidx\_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 **A<sub>mat</sub>**.

- To verify the *j*'th right eigenvector, the following applies:

$$\mathbf{A}_{\text{mat}} \cdot \mathbf{rEV}(\text{j-th column}) = \mathbf{Evals}(\text{j-th column}) \cdot \mathbf{rEV}(\text{j-th column}) \quad (32.12)$$

Matlab code (returns the difference between the left and right terms):

```
» Amat*rEV(:,j)-Eval(j)*rEV(:,j)
• To verify the j'th left eigenvector, the following applies:
```

$$\mathbf{IEV}(\text{j-th column})^* \cdot \mathbf{A}_{\text{mat}} = \mathbf{Evals}(\text{j-th column}) \cdot \mathbf{IEV}(\text{j-th column})^* \quad (32.13)$$

Matlab code (returns the difference between the left and right terms):

```
» (conj(IEV(:,j))' * Amat)'-Eval(j)*conj(IEV(:,j))
```

---

**Note:** The left eigenvectors require an additional conjugation before being used

---

- To verify the eigenvalues, the following commands need to be entered:

```
» D=eig(Amat)
» D=eig(Amat,M)
```

**QZ and Arnoldi Methods:** these two calculation methods do not generate the system matrix  $\mathbf{A}_{\text{mat}}$ .

- To verify the right eigenvectors, the following applies, where  $\mathbf{D}$  is a diagonal matrix containing the eigenvalues:

$$\mathbf{J} \cdot \mathbf{rEV} = \mathbf{M} \cdot \mathbf{rEV} \cdot \mathbf{D} \quad (32.14)$$

Matlab code to verify the j-th right eigenvector (returns the difference between the left and right terms above):

» Jacobian \*rEV(:,j)-M \*rEV(:,j) \*Evals(j)

- To verify the left eigenvectors, the following applies, where  $\mathbf{D}$  is a diagonal matrix containing the eigenvalues:

$$\mathbf{IEV}^* \cdot \mathbf{J} = \mathbf{D} \cdot \mathbf{IEV}^* \cdot \mathbf{M} \quad (32.15)$$

Matlab code to verify the j-th left eigenvector (returns the difference between the left and right terms above):

» transpose(conj(IEV(:,j))) \*J - EVals(j) \*transpose(conj(IEV(:,j))) \*M;

- To verify the eigenvalues, the following command needs to be entered:

» EIGEN=eig(Jacobian,M)

## 32.3 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.

### 32.3.1 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 ( 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 32.3.1). The variables and the corresponding equations for an eigenvalue  $\lambda = \sigma + j\omega$  are shown in table 32.3.1.

Name	Unit	Equation
Real part	1/s	$\sigma$
Imaginary part	rad/s	$\omega$
Magnitude	1/s	$\sqrt{\sigma^2 + \omega^2}$
Angle	deg	$\text{atan}2(\sigma/\omega)$
Damped Frequency	Hz	$ \omega/2\pi $
Period	s	$ 2\pi/\omega $
Damping	1/s	$-\sigma$
Damping Ratio		$-\sigma/\sqrt{\sigma^2 + \omega^2}$
Damping Time Constant	s	$1/ \sigma $
Ratio A1/A2		$e^{j2\pi\sigma/\omega}$

Table 32.3.1: Variables calculated for each eigenvalue

The screenshot shows a software interface titled "Modal/Eigenvalue Analysis Results - Eigenvalues". The main area is a table with 10 rows, each representing a mode. The columns are labeled: Name, Real part (1/s), Imaginary part (rad/s), Magnitude (1/s), Angle (deg), Damped Frequency (Hz), Period (s), Damping (1/s), Damping Ratio, and Damping Time Constant (s). The modes are numbered Mode 00001 to Mode 00012. The last row, Mode 00012, has its Damping Time Constant value highlighted in blue.

Name	Real part 1/s	Imaginary part rad/s	Magnitude 1/s	Angle deg	Damped Frequency Hz	Period s	Damping 1/s	Damping Ratio	Damping Time Constant s
Mode 00001	-0.0556127	0,	0.0556127	180,	0,	0,	0.0556127	1,	
Mode 00002	-0.0556201	0,	0.0556201	180,	0,	0,	0.0556201	1,	
Mode 00003	-0.0558487	0,	0.0558487	180,	0,	0,	0.0558487	1,	
Mode 00004	-0.0584869	0,	0.0584869	180,	0,	0,	0.0584869	1,	
Mode 00005	-0.0599036	0,	0.0599036	180,	0,	0,	0.0599036	1,	
Mode 00006	-0.0599592	0,	0.0599592	180,	0,	0,	0.0599592	1,	
Mode 00007	-0.0665768	0,	0.0665768	180,	0,	0,	0.0665768	1,	
Mode 00008	-0.0666898	0,	0.0666898	180,	0,	0,	0.0666898	1,	
Mode 00009	-0.0674142	0,	0.0674142	180,	0,	0,	0.0674142	1,	
Mode 00010	-0.0674979	0,	0.0674979	180,	0,	0,	0.0674979	1,	
Mode 00011	-0.0675933	0,	0.0675933	180,	0,	0,	0.0675933	1,	
Mode 00012	-0.0678536	0,	0.0678536	180,	0,	0,	0.0678536	1,	

Figure 32.3.1: 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 32.3.2, 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 %.

Name	Object	State Variable	Mode Index	Mode: Real part 1/s	Mode: Imaginary part rad/s	Mode: Damped Frequ... Hz	Mode: Damping ratio
Northeast; NE_G1; speed	NE_G1	speed	26	-0,149619	6,23588	0,992471	0,023986
Northeast; NE_G2; speed	NE_G2	speed	26	-0,149619	6,23588	0,992471	0,023986
Northeast; NE_G3; speed					6,23588	0,992471	0,023986
Northeast; NE_G4; speed					6,23588	0,992471	0,023986
Northeast; NE_G5; speed					6,23588	0,992471	0,023986
Northeast; NE_G6; speed					6,23588	0,992471	0,023986
Northeast; NE_G7; speed					6,23588	0,992471	0,023986
Northeast; NE_G8; speed					6,23588	0,992471	0,023986
Northwest; NW_G1; speed					6,23588	0,992471	0,023986
Northwest; NW_G2; speed					6,23588	0,992471	0,023986
Northwest; NW_G3; speed					6,23588	0,992471	0,023986
Northwest; NW_G4; speed					6,23588	0,992471	0,023986
Northwest; NW_G5; speed					6,23588	0,992471	0,023986
Northwest; NW_G6; speed					6,23588	0,992471	0,023986
Southeast; SE_G1; speed					6,23588	0,992471	0,023986
Southeast; SE_G2; speed					6,23588	0,992471	0,023986
Southeast; SE_G5; speed					6,23588	0,992471	0,023986

Figure 32.3.2: 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.

#### 32.3.2 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 phasor 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 32.3.3.

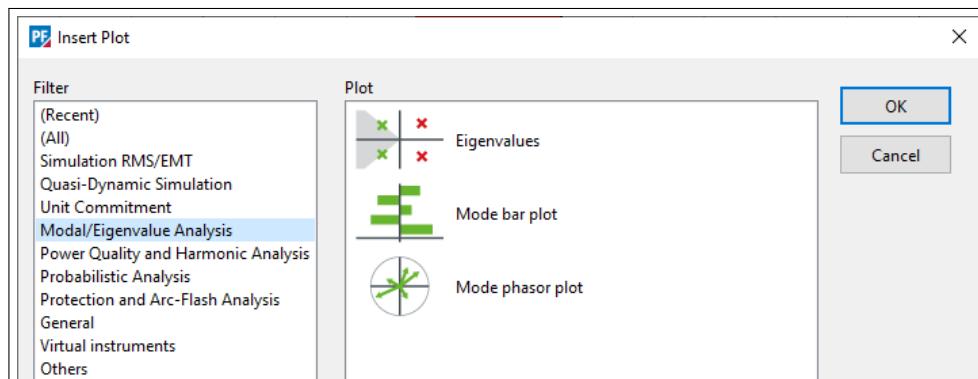


Figure 32.3.3: Selection of the Modal Analysis plots

### 32.3.2.1 Eigenvalues Plot

Eigenvalues plots are inserted by choosing *Eigenvalues plot* from the insert dialog shown in figure 32.3.3. 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 eigenvalues plot is shown in figure 32.3.4.

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, or the damped frequency 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 bring up a pop-up dialog where the index, the complex representation, the polar representation and oscillation parameters of the mode can be viewed.

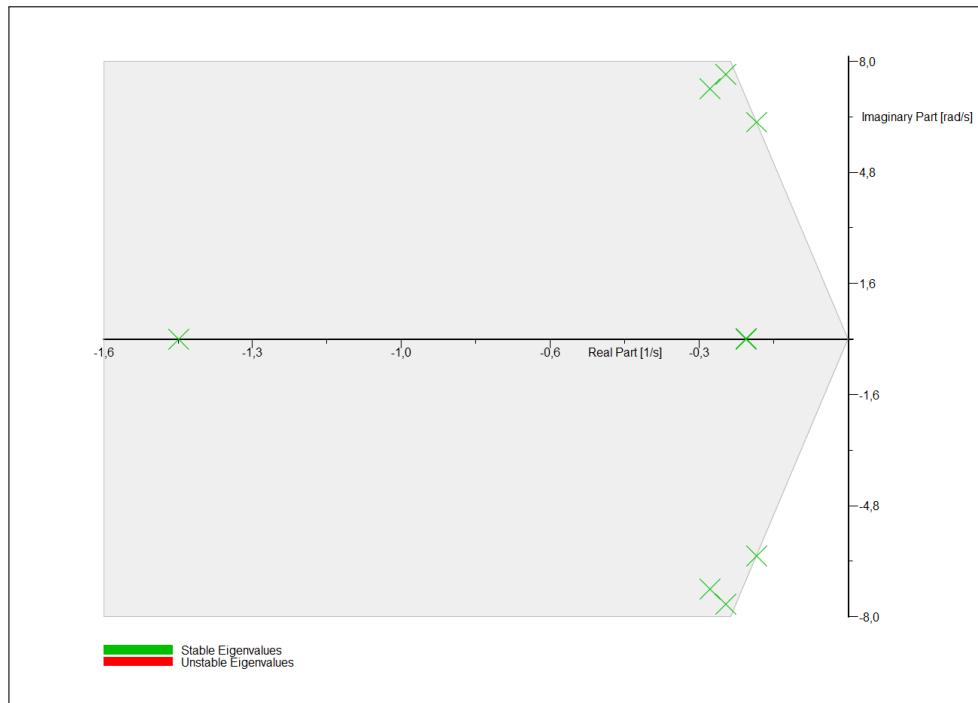


Figure 32.3.4: The Eigenvalues Plot

#### Editing the Eigenvalues Plot

All settings that control the appearance of the eigenvalues plot can be accessed by double clicking on an empty area of the plot. The options available are:

- Appearance: here the colour of the stable and unstable eigenvalues can be adjusted. Is also possible to decide whether to display the plot legend and the stability borders. The so-called *Stability Borders* option shades the area of the plot containing all the modes shown on the plot. It is not an *area of stability* as such.
- 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*.

- Scale: here the range of the plot (x and y axis limits) can be defined. Also by enabling the *Adapt Scale* option, the x and y axis tick marks will be displayed as integer values, rather than floating point numbers. For example, the axis marks will be 10.0, 20.0 and 30.0 rather than 9.7988, 19.5976 and 29.3964.

### Damping Ratio Constant

A constant showing a predefined damping ratio can be added into the plot by right clicking on the plot and choosing *Constant damping ratio line*. The use of this constant, shown in figure 32.3.5, 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*.

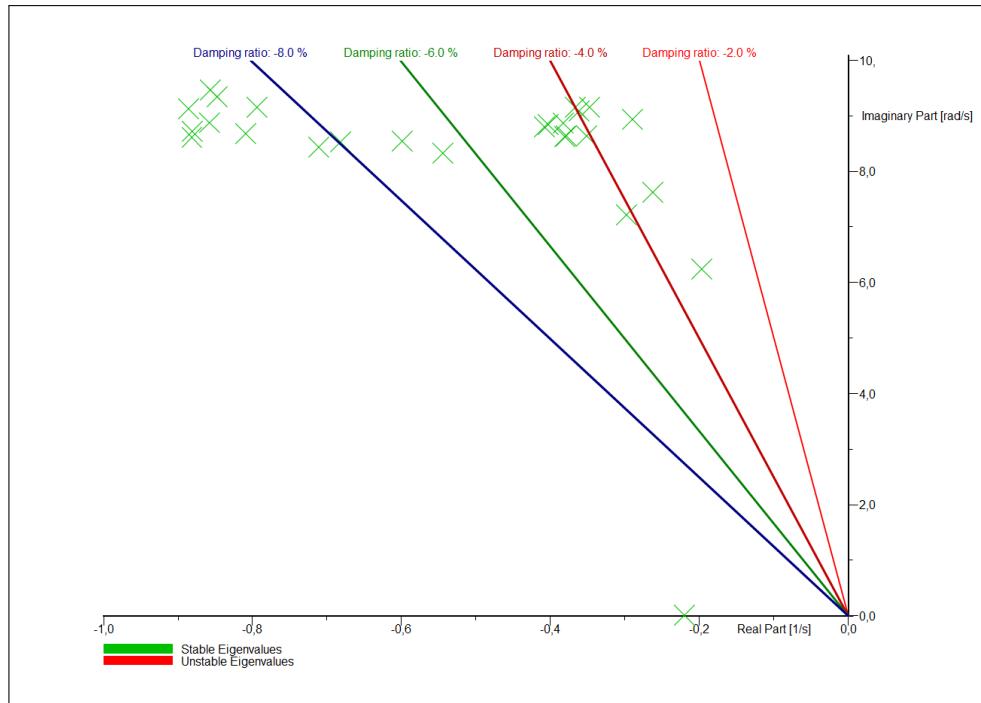


Figure 32.3.5: Damping Ratio Constants in Eigenvalues Plot

#### 32.3.2.2 Mode Bar Plot

Mode bar plots can be inserted by choosing *Mode bar plot* from the insert dialog shown in figure 32.3.3. When this option is selected, an edit dialog of the plot will open, with the following options available:

##### Editing the Mode Bar Plot

- 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 plot legend and also the annotation (value) for each bar.

- Scale: 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 *Bar limit* field.

### Interpreting the Mode Bar Plot

An sample Mode Bar Plot is shown in Figure 32.3.6. 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.

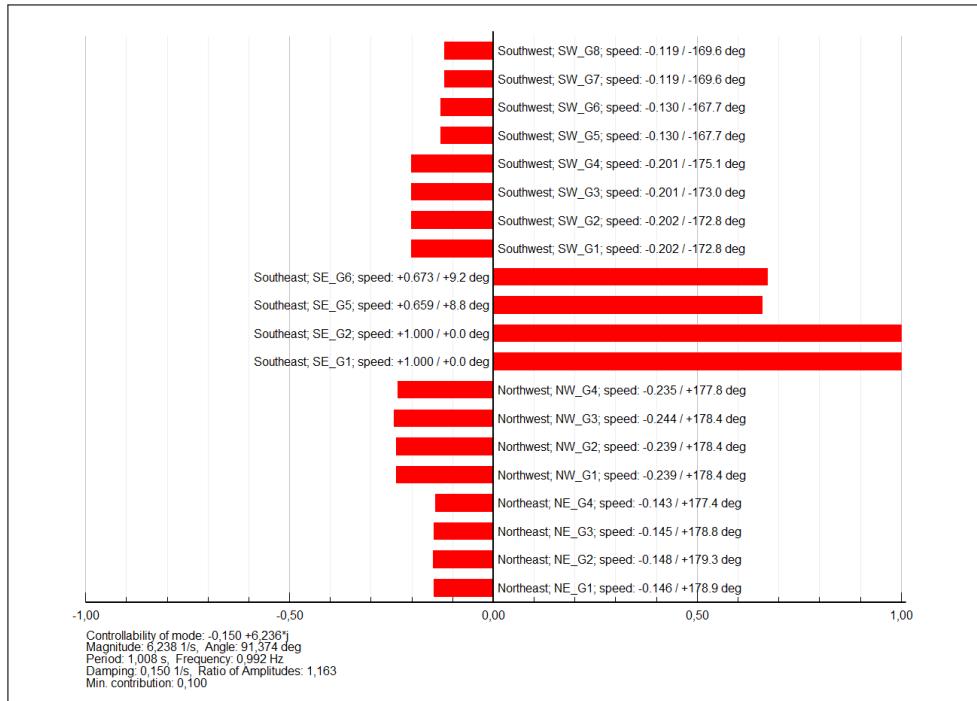


Figure 32.3.6: Example Mode Bar Plot

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* → *Controllability/Observability/Participation*.
- From the modal analysis modes result: right click on a mode and select *Show* → *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 48.4.6.

#### 32.3.2.3 Mode Phasor Plot

Mode phasor plots can be inserted by choosing *Mode phasor plot* from the insert dialog shown in figure 32.3.3. 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 Phasor Plot

All settings that control the appearance of the Mode Phasor Plot can also be accessed by double clicking on an empty area of the plot. The dialog that appears is very similar to the dialog for the Mode Bar Plot and the Mode Selection, Filter Options, Appearance and Scale 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 parameter entered. A cluster shares the same diagram colour.
- **Show only points:** if this parameter is enabled, the vectors will appear as points on the diagram rather than arrows.
- **Show unit circle:** the unit circle can be removed from the plot by disabling this option.
- **Auto scale:** if the *Auto scale* checkbox is ticked, then the plot is autoscaled. Otherwise, the plot radius can be specified in the corresponding *Radius* field.

### Interpreting the Mode Phasor Plot

An example of a mode phasor plot is shown in figure 32.3.7. The mode phasor 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).

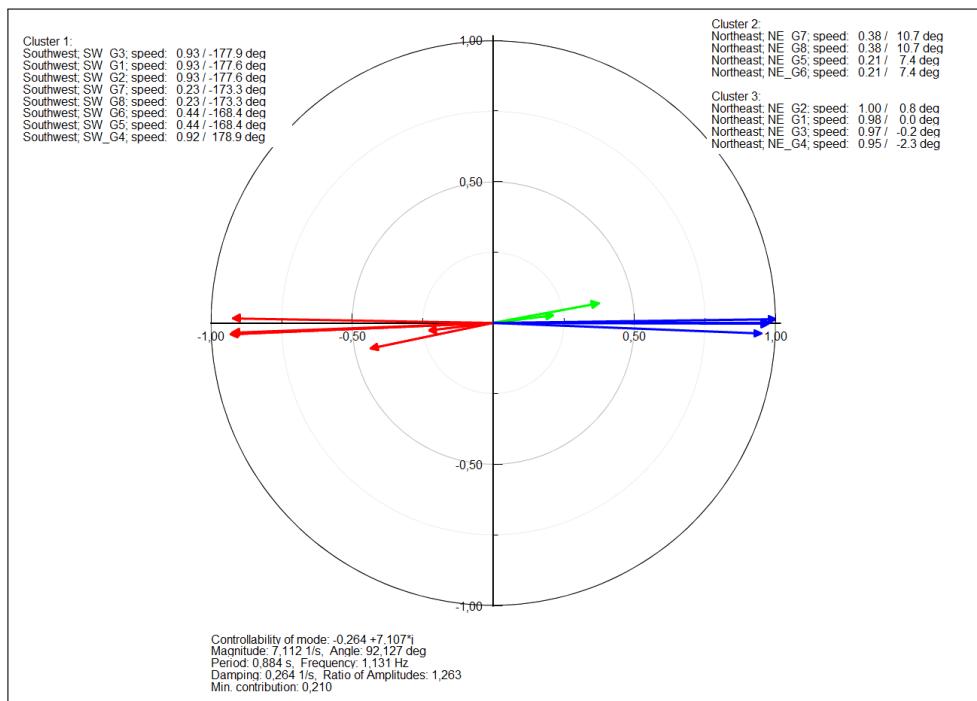


Figure 32.3.7: The Mode Phasor plot

The mode phasor plot can also be obtained in one of the following ways:

- From the eigenvalues plot: right click on a mode and select *Create phasor plot* → *Controllability/Observability/Participation*.
- From the modal analysis modes result: right click on a mode and select *Show* → *Phasor 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 48.4.6.

#### 32.3.2.4 Export a Modal Analysis Plot

Any of the modal analysis plots can be exported for use in an external software program. To export a modal analysis plot follow these steps:

1. From the main *PowerFactory* file menu, choose the option *File* → *Export Graphic*. A *Save As* dialog will appear.
2. Choose an appropriate File name and disk location and click *Save*.

---

**Note:** The process of exporting multiple plots can be automated using a DPL or Python script. More information is available in the DPL and Python References.

---

#### 32.3.3 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.

#### 32.3.4 Modal Analysis Results in the Output Window

The modal/eigenvalues analysis results can be displayed in the output window using the *Output of Results* command, accessible by clicking on *Output Calculation Analysis* icon () from the main toolbar or via the menu *Output* → *Output Calculation Analysis*.

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 32.3.8. 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.

---

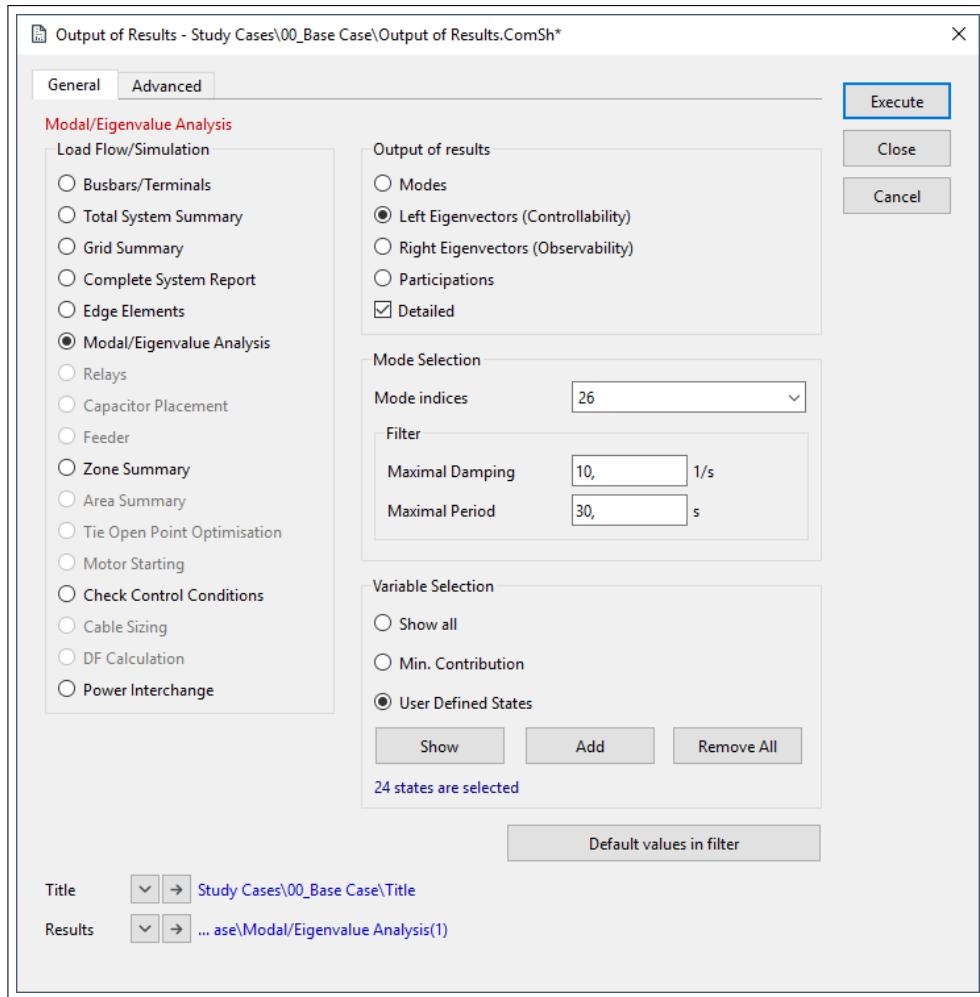


Figure 32.3.8: Output of Results Command

### 32.3.5 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 ().
  - From the *Modal Analysis* page choose the *Variable Set Calculation Parameter*.

- In the *Available Variables* window, scroll to near the bottom until you see the variables p\_mag (Participation, Magnitude) etc. Holding **shift** select this variable and all eight other variables down to rEVec\_mags (Observability, Magnitude signed).
  - Click on the variables or on the *right arrows* icon  between the *Available Variables* and *Selected Variables* windows.
  - 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 [32.3.1](#)) 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 *DIGSILENT* 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](#))
- Distance 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.

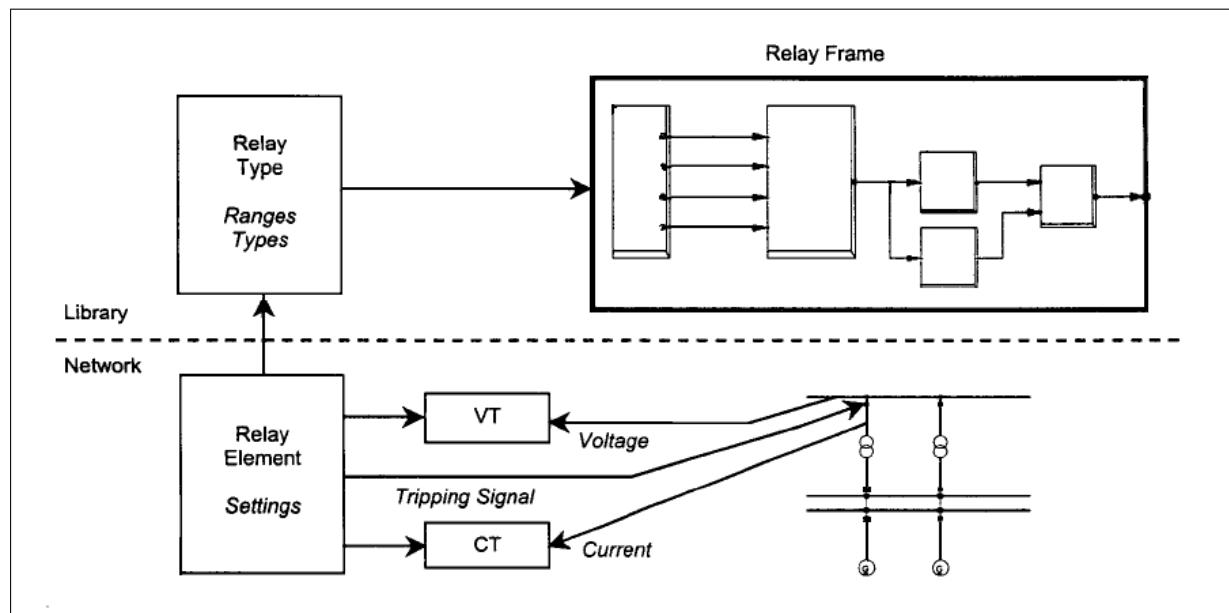


Figure 33.1.1: Modelling structure for protection devices

### 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 30.1.2 (The Composite Frame) 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.

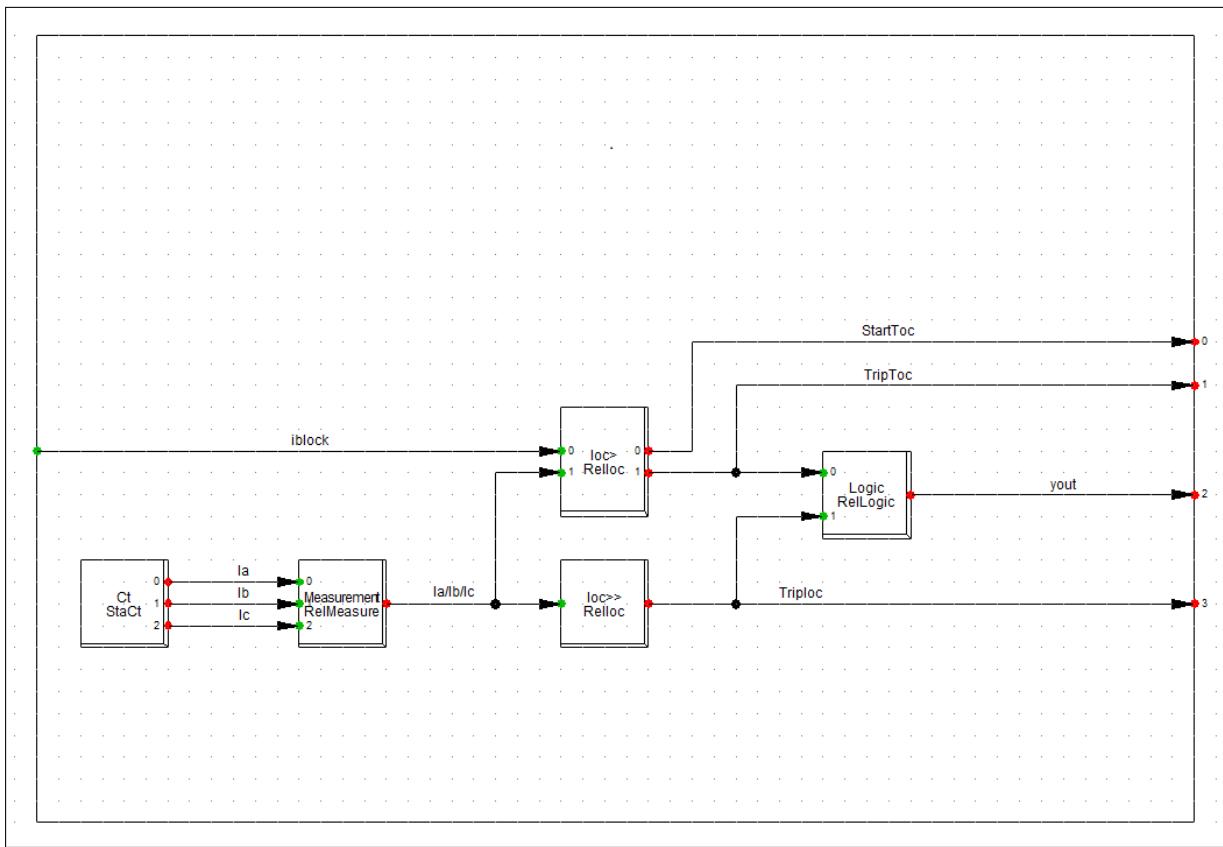


Figure 33.1.2: Typical relay frame

### 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.

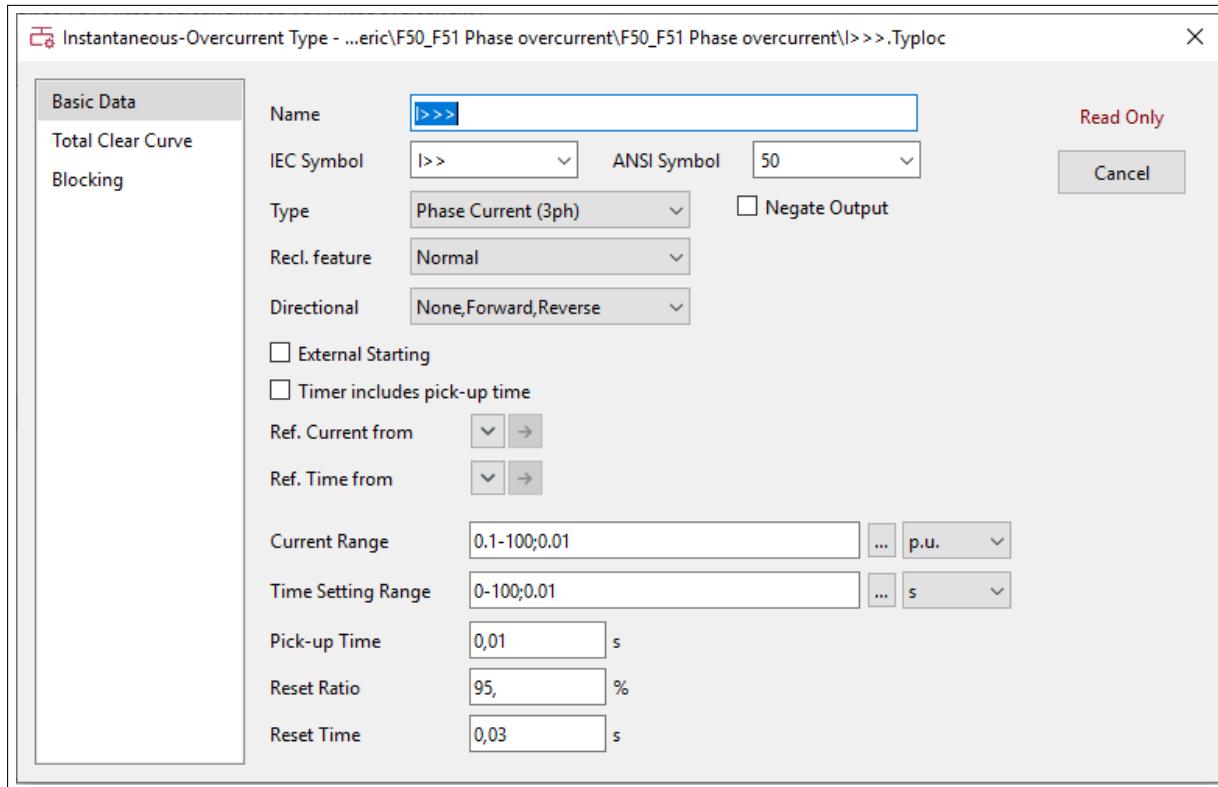


Figure 33.1.3: Type dialog of an instantaneous overcurrent block

### 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.

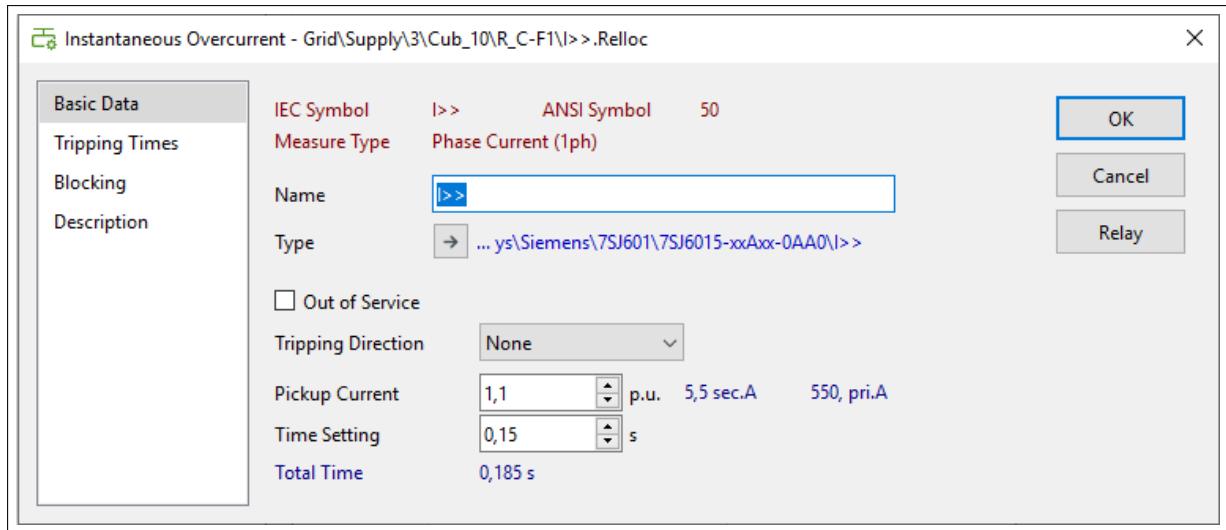


Figure 33.1.4: Element dialog of an instantaneous overcurrent relay

## 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 modelling of a protection scheme can be started, it is necessary to construct a model of the network to be protected. See Section 11.2.

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 24.13).

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.

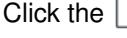
For “generic” relays, *PowerFactory* includes a global library containing some predefined generic relays, fuses and instrument transformers that can be used to design schemes without requiring specific details of a particular relay manufacturer’s range of products. This can be useful during the early 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 from the user download area on the *DlgSILENT* website. 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 sensitive menu.
  - (b) Choose *New Devices* → *Relay Model.../Fuse.../Current Transformer.../Voltage Transformer...*. A dialog for the chosen device will appear.
3. Right-clicking a switch symbol (Edit devices):
  - (a) Right-click a switch symbol in the single line graphic as illustrated in Figure 33.2.1. This will show a context sensitive menu.
  - (b) Choose the option *Edit 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 *Edit 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 11.2.7 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 sensitive menu will appear. See Figure 33.2.3 for an example substation showing possible locations where protection devices can reside.
  - (c) Choose *New Devices* or *Edit Devices* and following 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 33.2.2.

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 sensitive menu appears.

---

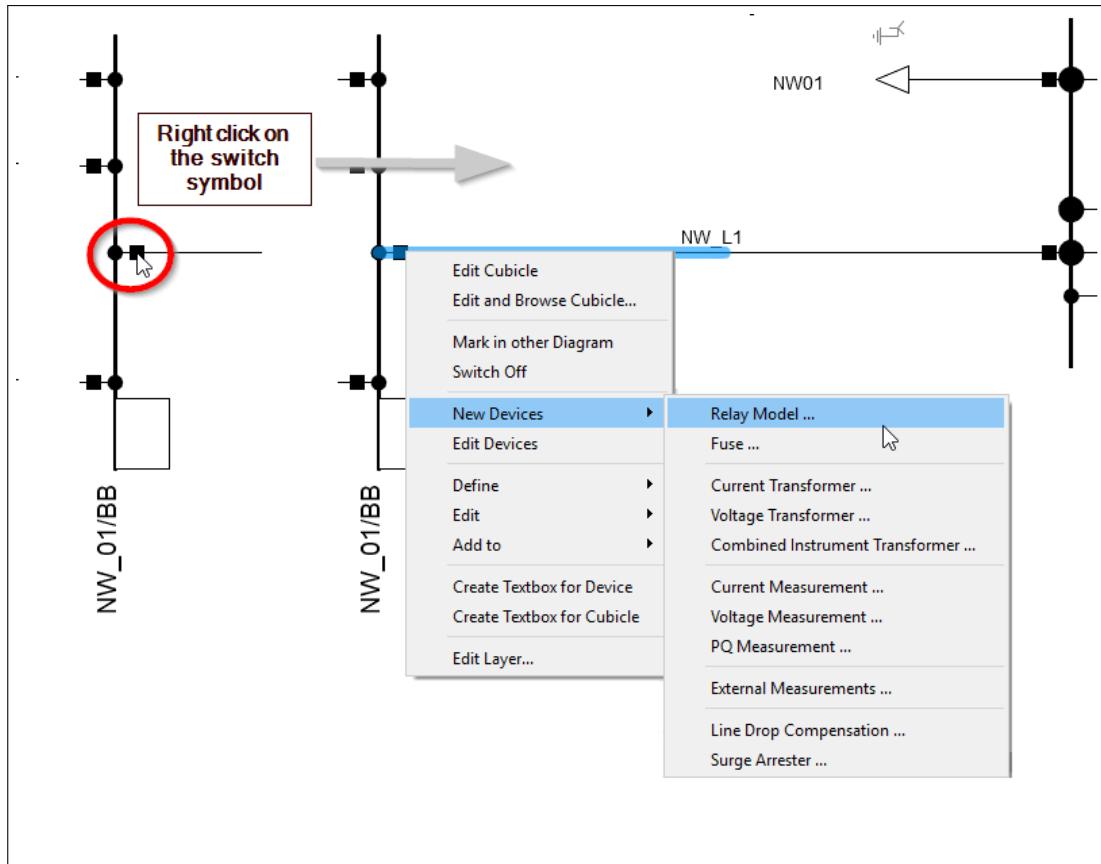


Figure 33.2.1: Adding a new relay to a single line diagram

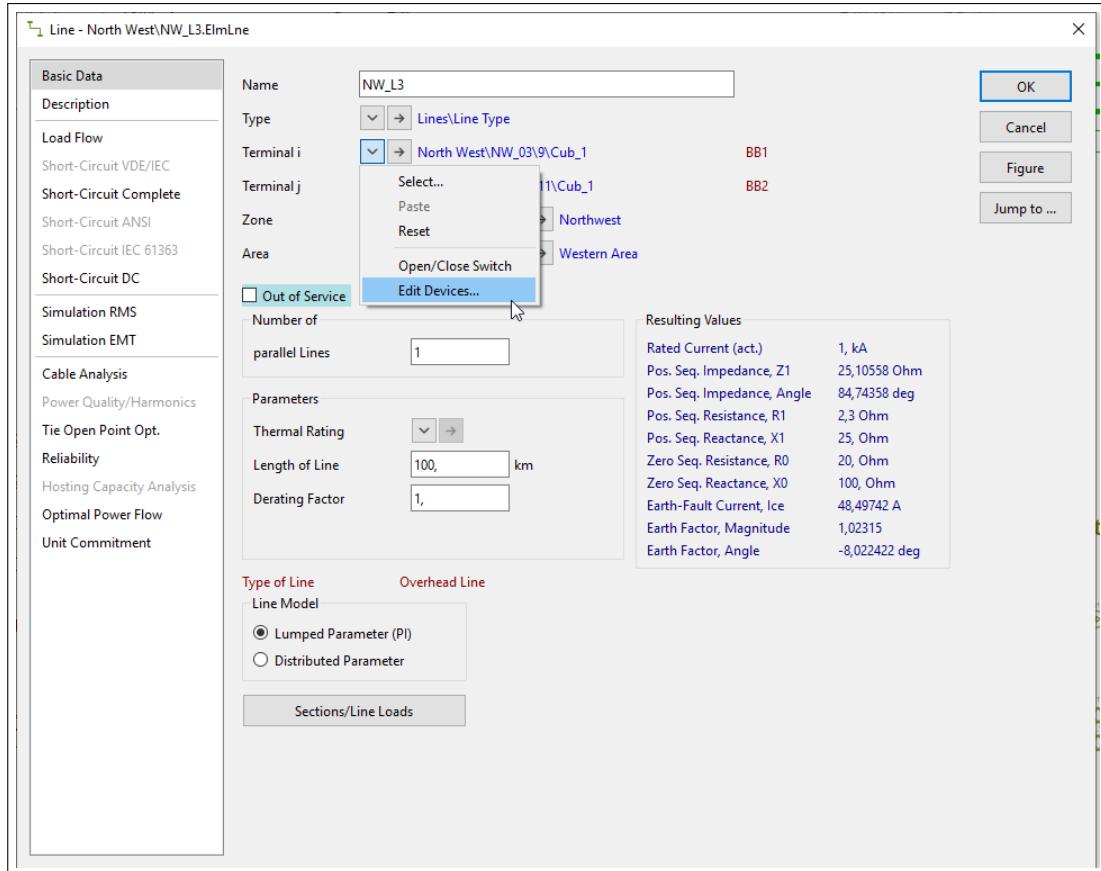


Figure 33.2.2: Editing line protection devices

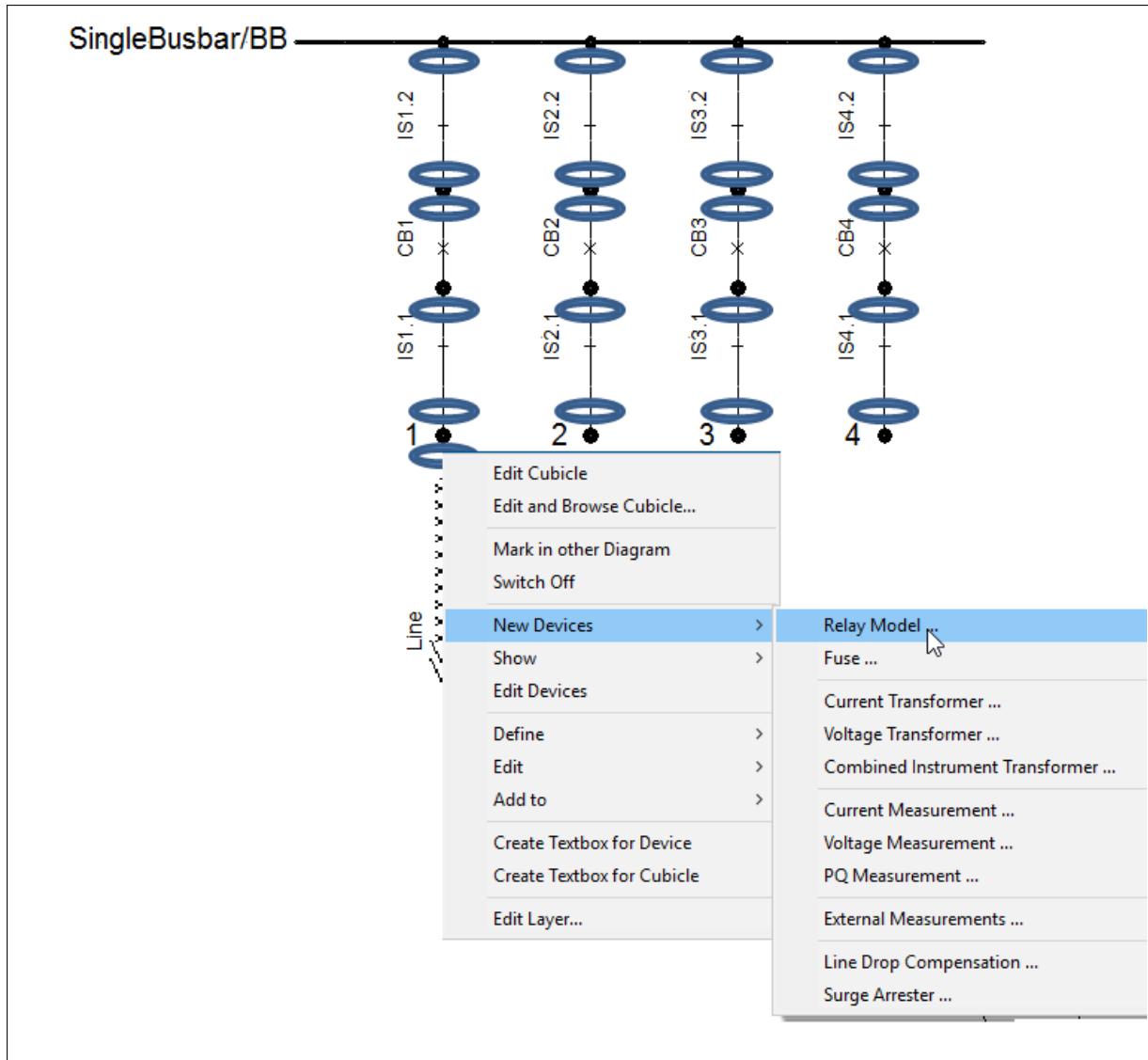


Figure 33.2.3: Adding a new protective device to a detailed substation graphic

### 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 11.6).

In the second instance, protection devices can be added to the model as described in the remainder of this section.

An example of a complete protection single line diagram is shown in Figure 33.2.4. In this diagram the protection relays are indicated with the “R” inside a rectangle, current transformers as a brown circle with the measured circuit underneath and voltage transformers as a brown circle with a semi-circle above and a line connecting to the measured bus. Black lines between the measurement transformers and the relays show the connection of the secondary side of the instrument to the relay.

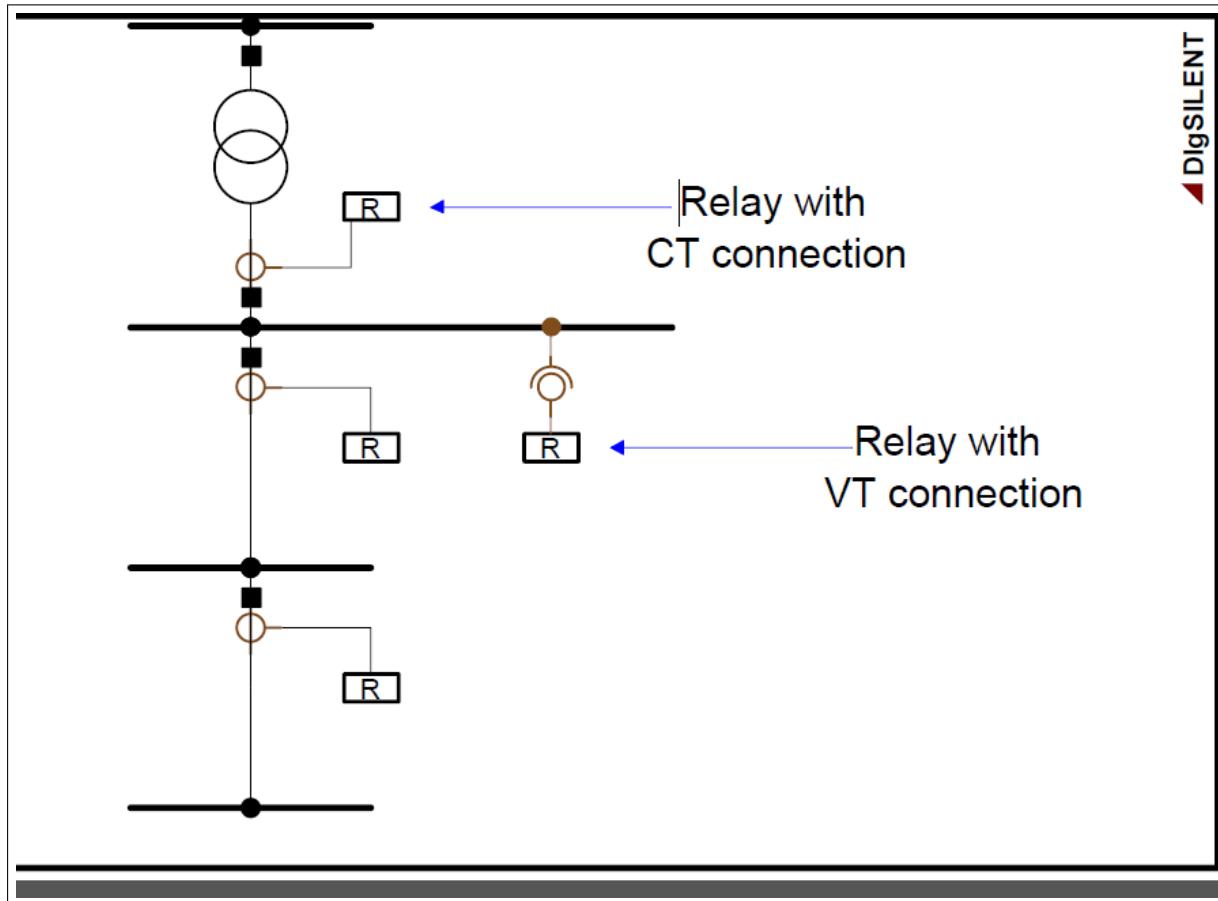


Figure 33.2.4: An example protection single line diagram

### 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 that the secondary side of the device will connect to, 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 that the secondary side of the device will connect to, 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 that the secondary side of the devices will connect to, 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, 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 object filter

To locate protection devices using the built-in object filters follow these steps:

1. Click the  icon from the main toolbar. The Network Model Manager, which lists all calculation relevant objects, will open.
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 sensitive 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 Global Type*. . . . A data page showing the global relay library will appear.
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 Project Type*. . . . A data page showing the local type library will appear.
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 [33.4.7.1](#).

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 *zemphshow* 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 [33.3.3.2](#) 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 sensitive 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  $I_{0\Delta}$  currents derived from the line currents. In this case the CT will interact with the  $I_{0\Delta}$ \_signals of the winding. With such a connection it is assumed that the CT will be single phase and measuring  $I_0$  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.

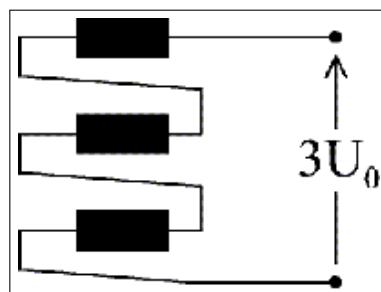


Figure 33.3.1: The open delta (O) winding connection

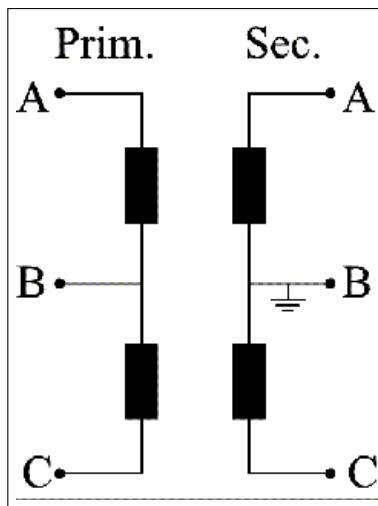


Figure 33.3.2: The V winding connection

#### 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 Project Type....*. 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 *New Devices* → *Fuse* .... 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 either:
- (a) *Select Global Type*. A dialog will appear showing you a library of built-in fuses where you can select an appropriate one; or
  - (b) *Select Project Type*. A dialog will appear showing you the local project library where you can choose a fuse type that you have created yourself or downloaded from the *DlgSILENT* website.
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.

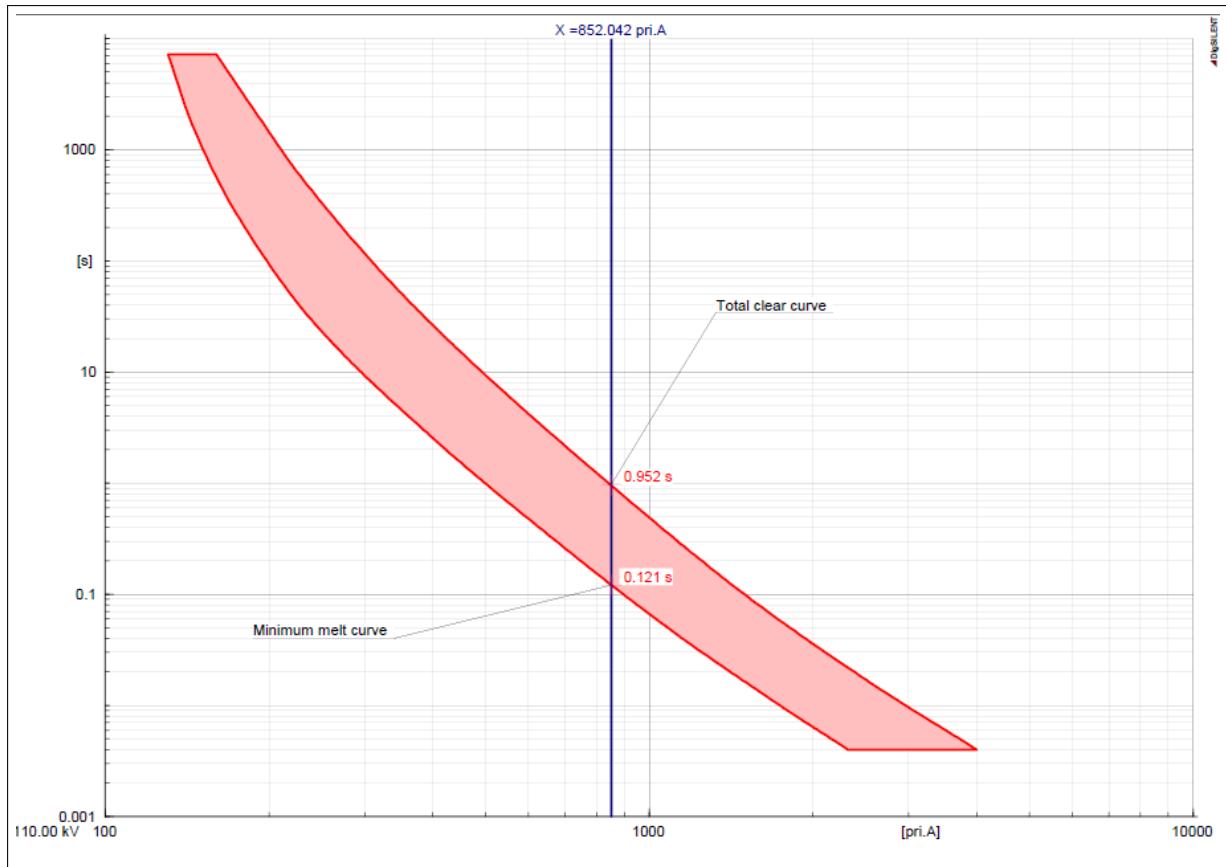


Figure 33.3.3: Fuse melt characteristics

### 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 “*I<sub>b</sub>*” 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 41 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 45.

### 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 *Show Graphic*, 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 section of the [Technical References Overview Document](#).

### 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.

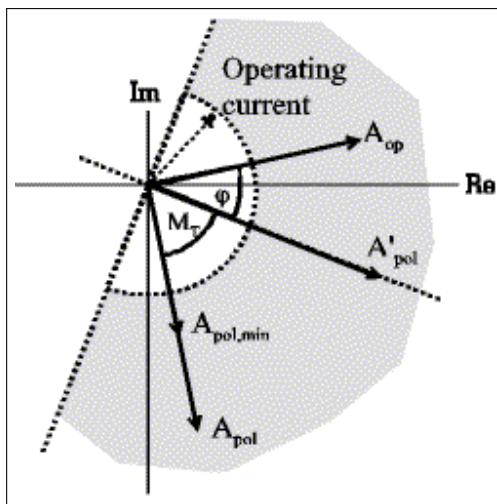


Figure 33.3.4: Directional relay principle diagram

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.

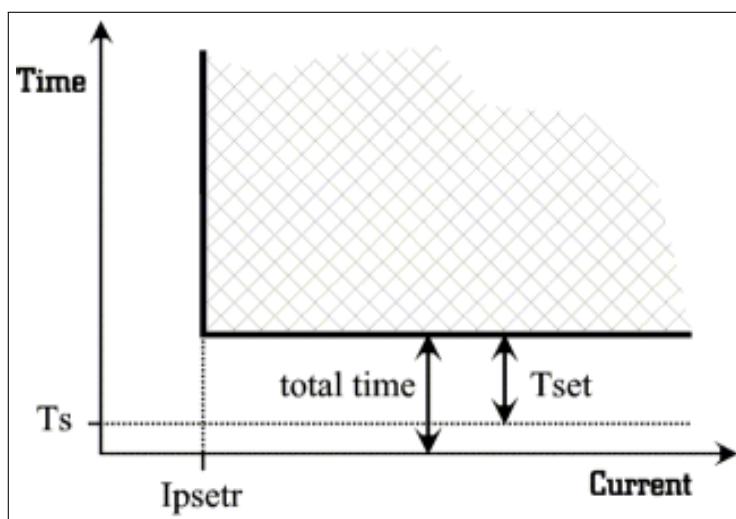


Figure 33.3.5: Instantaneous overcurrent tripping area

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.

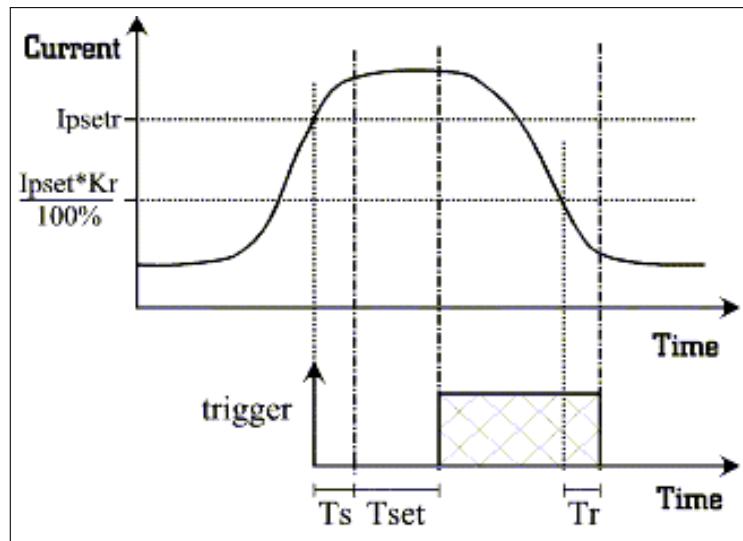


Figure 33.3.6: Instantaneous overcurrent timing diagram

#### 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.

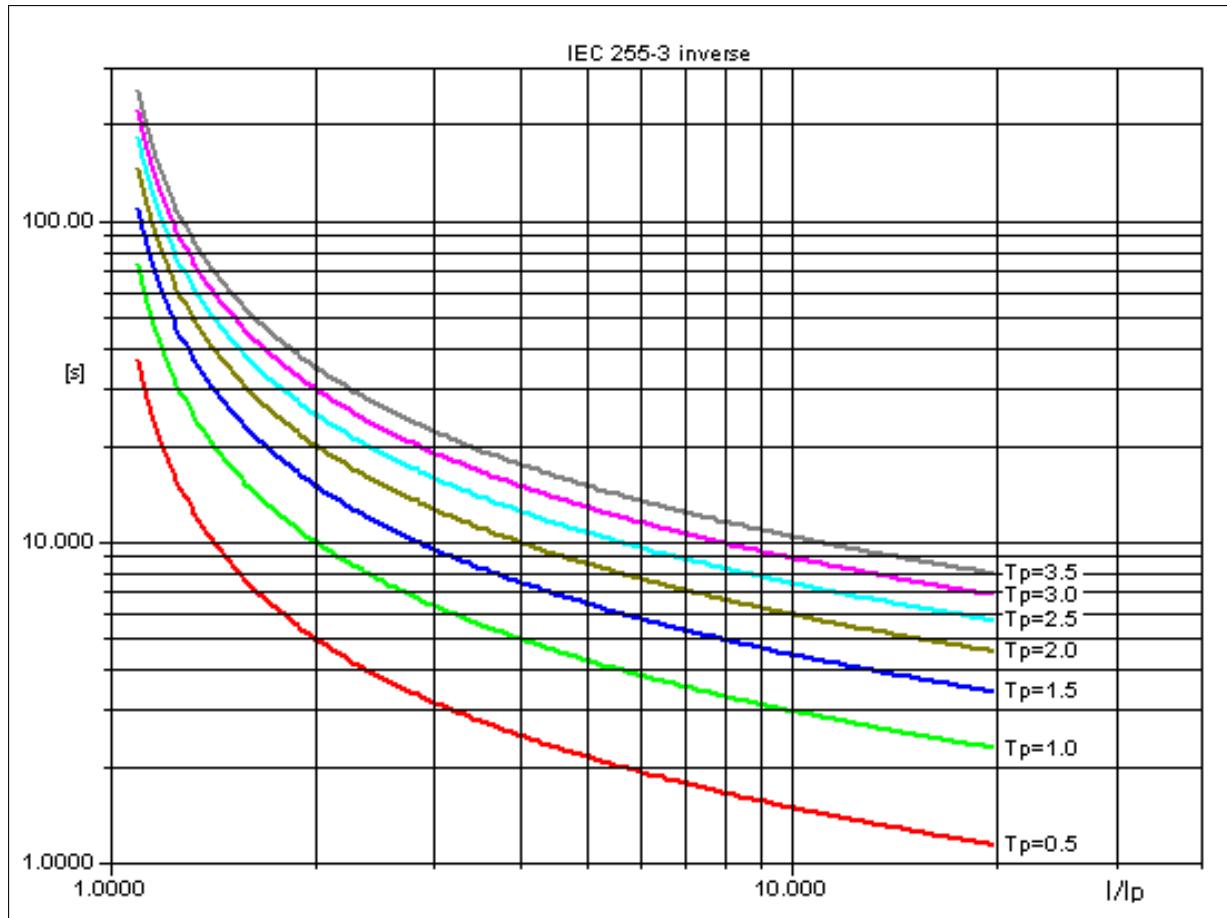


Figure 33.3.7: I-t curves for different time dials

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.

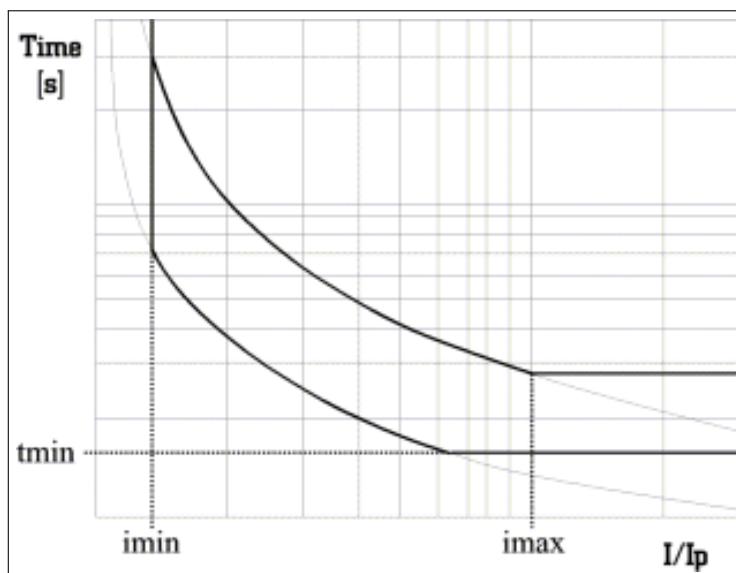


Figure 33.3.8: I-t curve limits

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 (*VisOcplot*) 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.

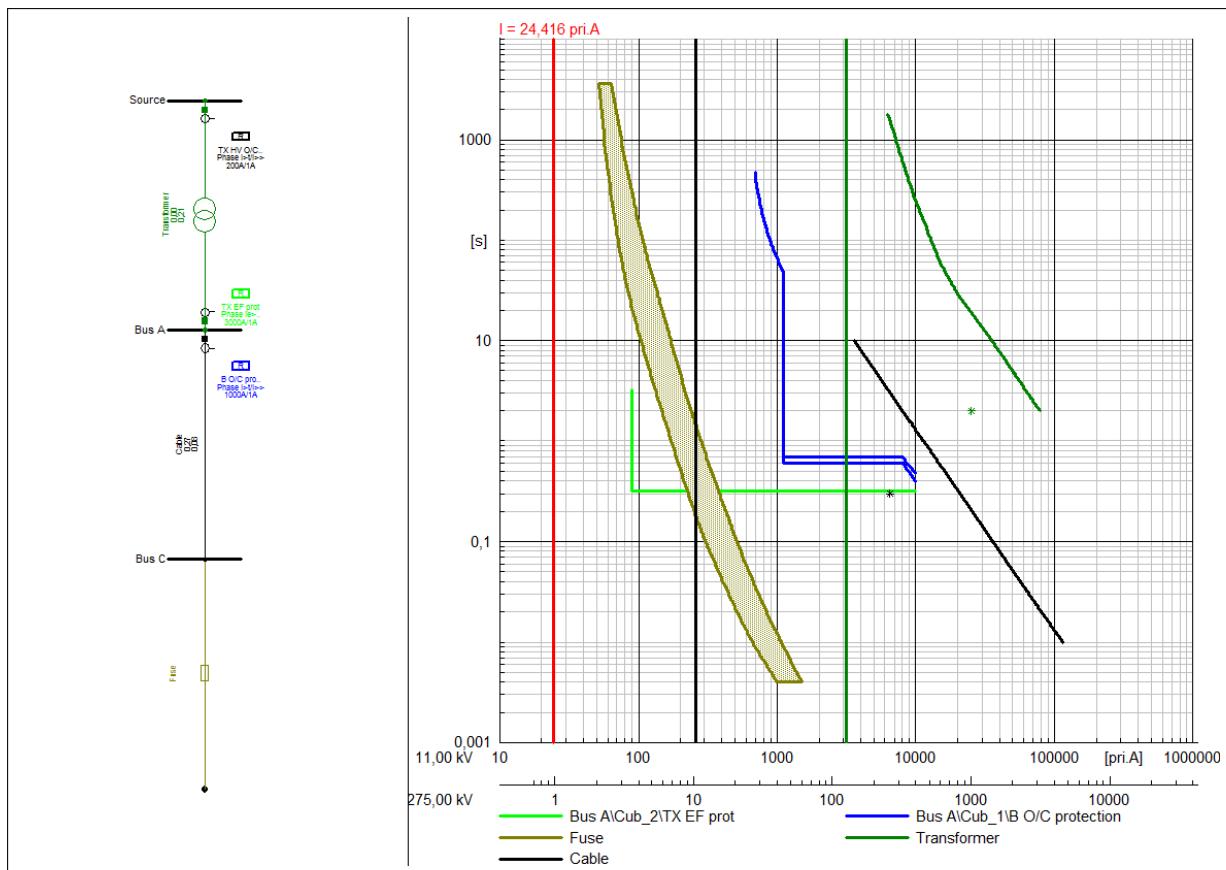


Figure 33.4.1: Time-overcurrent plot showing the auto-generated graphic for the protection path

### 33.4.1 How to create a time-overcurrent plot

There are four 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 fourth method shows 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 sensitive menu will appear.
- Select the option *Create 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 sensitive menu will appear.
- Select the option *Show → 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 sensitive menu will appear.
- Select *Show → Time-Overcurrent Plot*.

#### 4. From the protection path

- (a) Navigate to the protection path in the Data Manager.
- (b) Right-click the  icon. A context sensitive menu will appear.
- (c) Select *Show → Time-Overcurrent Plot*. Refer to Section 33.8 (The time-distance plot) for more information on defining paths. In this case the time-overcurrent plot will also show an auto-generated schematic of the path to the left of the diagram. This plot can also be manually adjusted. Refer to Section 33.4.7.

In methods 1-3, it is also possible to select the option *Add to Time-Overcurrent Plot* instead of *Show → 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 *Show → 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. Right-click the time-overcurrent plot. A context sensitive 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

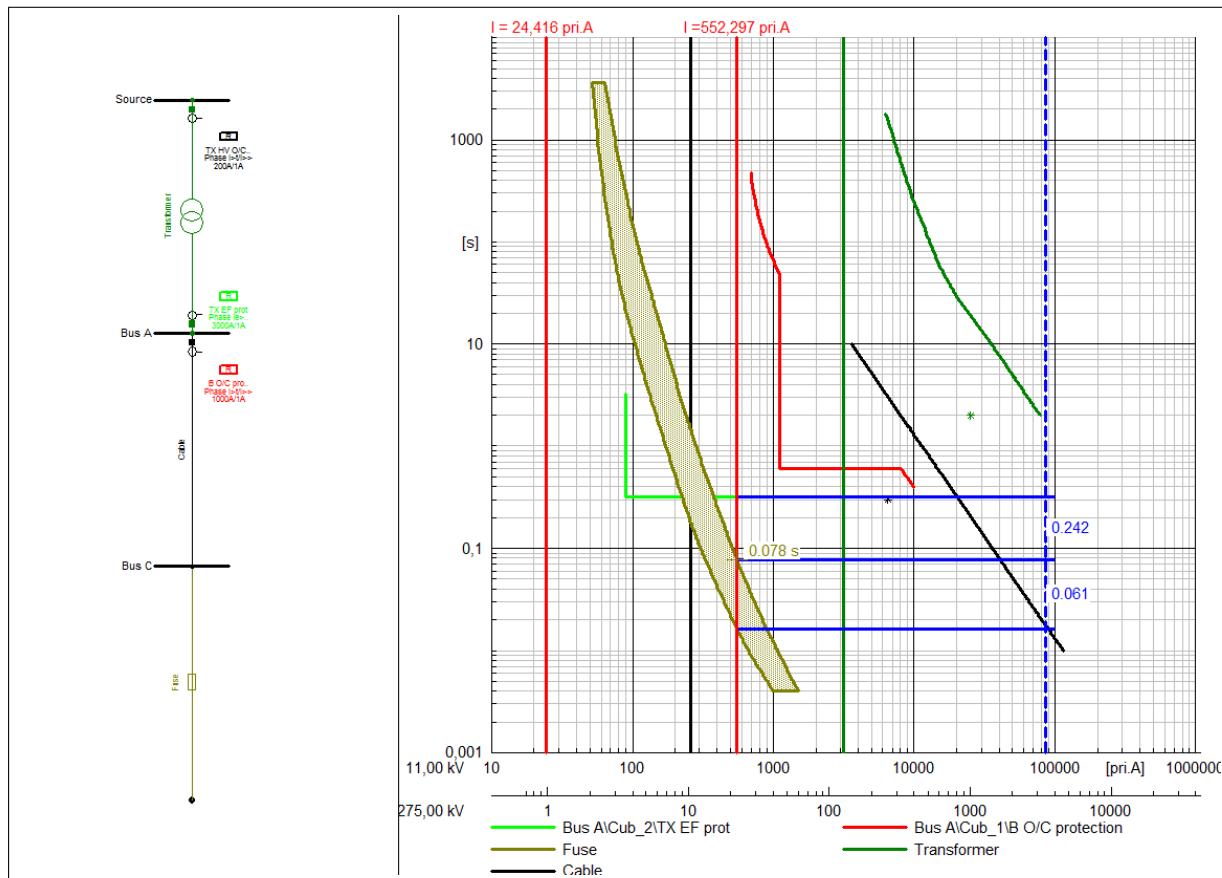


Figure 33.4.2: Time-overcurrent plot with grading margins displayed in blue

**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 33.4.9.2.

### 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 sensitive 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.

- (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 *Set Constant* → *x-Value*. 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.
    - (d) Press **OK** to add the line to the diagram.

### 33.4.6 Configuring the auto generated protection diagram

The auto-generated protection diagram that is created when a time-overcurrent diagram is generated from the protection path (see option 4 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 *Edit graphic 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 Overcurrent plot options

To access the time-overcurrent plot settings, either:

1. Right-click the time-overcurrent plot and select *Options*; or
2. Double-click the time-overcurrent plot and click **Options** underneath the Cancel button in the displayed dialog.

#### 33.4.7.1 Basic options page

The basic options page of the time-overcurrent options dialog shows the following:

**Current Unit.** The current unit may be set to either primary or secondary (relay) amperes.

**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.

**Characteristic.** This option defines whether the displayed curves also show the curves including the additional circuit breaker delays. The default option *All* shows both the minimum clearing time (not including the breaker delay) and the total clearing time (including the breaker delay). It is possible also to display just one of these curves. 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*.

**Recloser Operation.** The different recloser stages can be shown simultaneously or switched off in the diagram.

**Display automatically.** This option is used to select how the calculated load-flow or short-circuit currents will be displayed. Either the current lines, the grading margins, both or none may be selected.

**Consider Breaker Opening Time.** This option determines whether the relay characteristics will also include the breaker (switch) operating time.

**Voltage Reference Axis.** 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. **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. **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 33.4.9.2 for more information on grading margins when dragging the time-overcurrent characteristics.

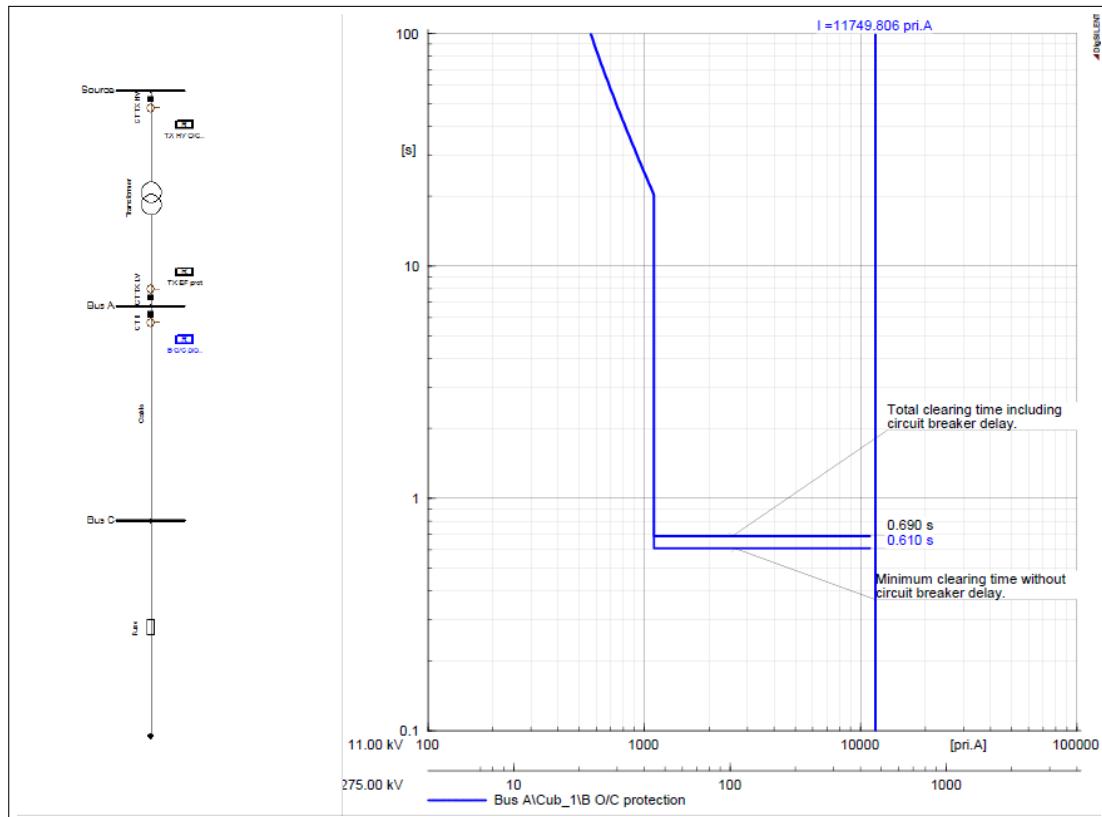


Figure 33.4.3: time-overcurrent plot showing an overcurrent characteristic including also the breaker delay time.

### 33.4.7.2 Advanced options page

**Drag & Drop Step Sizes.** These are used to set the step change in the relay settings when a time-overcurrent plot is dragged with a continuous time dial or pickup current. **Time Range for Damage Curves.** This option defines the maximum and minimum time limits for the transformer and line damage curves.

**'Colour for Out of Service' Units.** The characteristics for units that are out of service are invisible by default. However, a visible colour may be selected. **Brush Style for Fuses.** This defines the fill style for fuse curves when they have a minimum and maximum melting time defined.

**Number of 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.

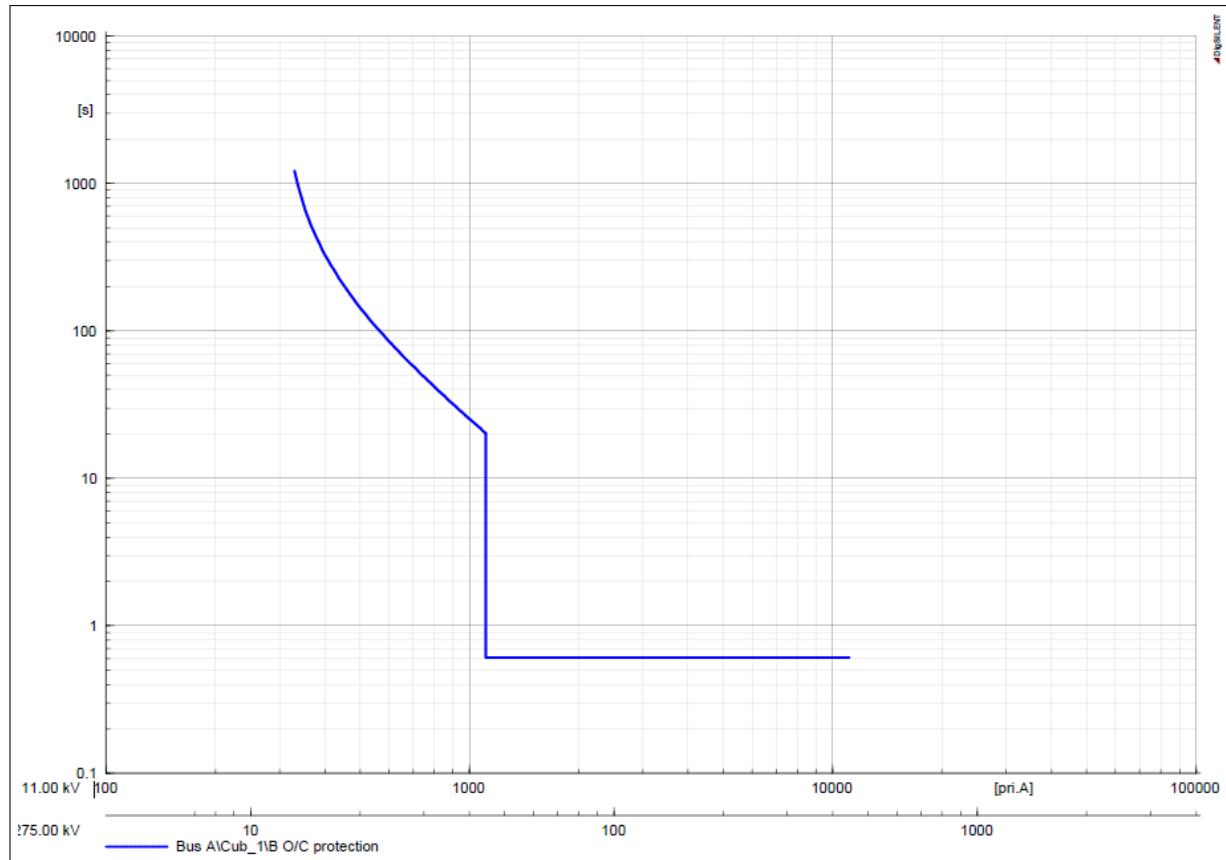
**Thermal Image, Pre-fault Current.** In some time-overcurrent relay characteristics, the tripping time is dependent on the pre-fault current. This box allows the user to enter a custom value for the pre-fault current, or to use the automatically calculated load-flow current.

### 33.4.8 Altering protection device characteristic settings from the time-overcurrent plot

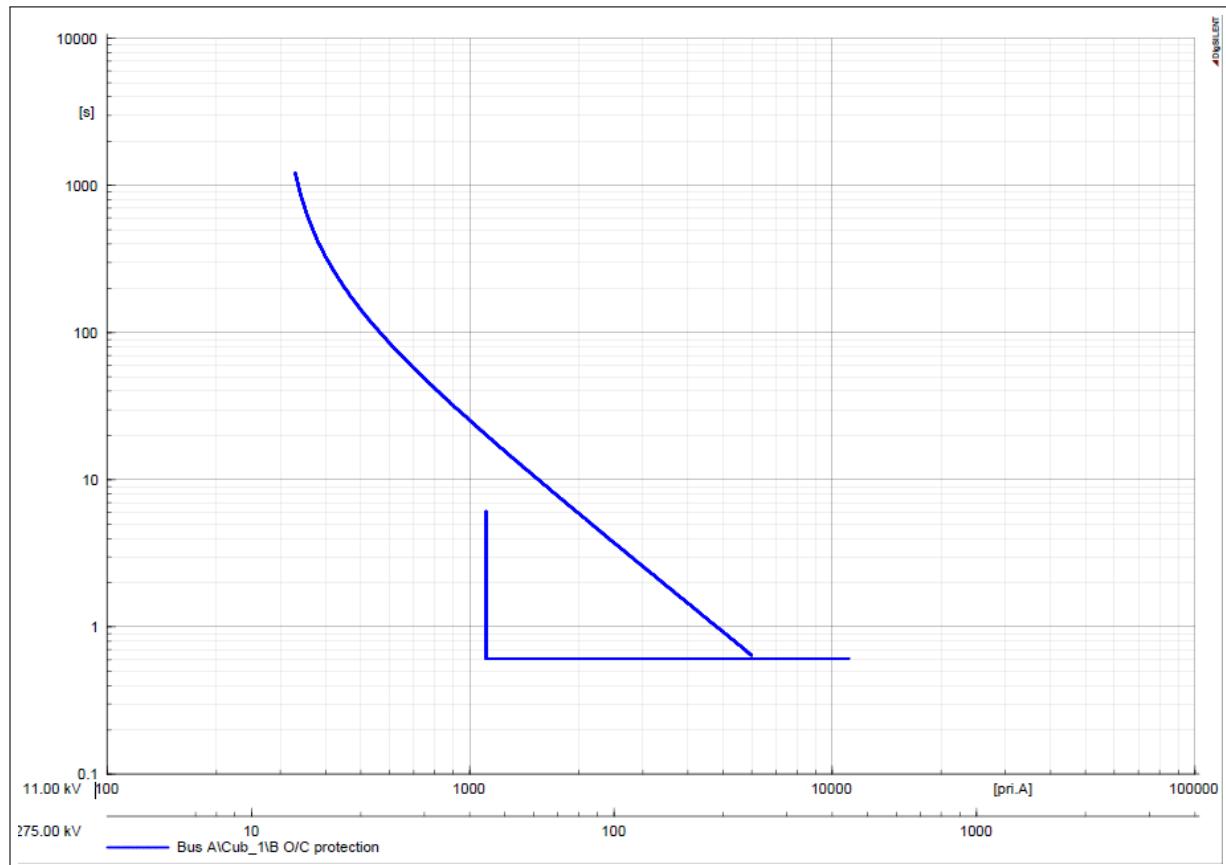
The time-overcurrent plots can be used to alter the relay characteristics graphically. This section describes various procedures used to alter such characteristics.

### 33.4.9 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) Unsplitted



(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 sensitive menu will appear.
  - (b) Select the option *Split*.
2. Method 2:
  - (a) Double-click the time-overcurrent plot avoiding any shown characteristics.
  - (b) On the lower table section of the displayed dialog *check* the *Split Relay* 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.

---

#### 33.4.9.1 Altering the sub-characteristics

The first step is to *Split* the relay characteristic. See Section 33.4.9. 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.

---

#### 33.4.9.2 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 grey 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 “1”, the new position as “2”, and the grading margins are labelled “a”.

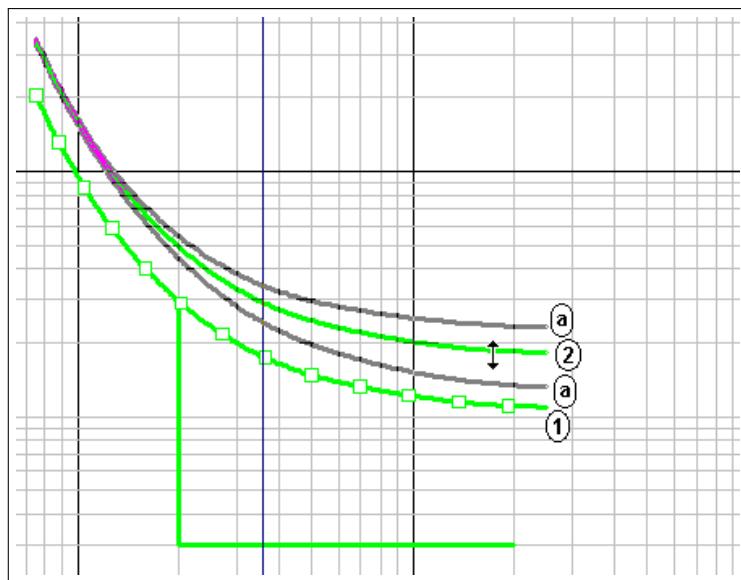


Figure 33.4.5: Grading margins when Moving a characteristic

### 33.4.10 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

#### 33.4.10.1 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 sensitive menu will appear.
  - (b) Select (*Show → 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 sensitive menu will appear.
  - (b) Select (*Add → Transformer Damage Curve / Conductor/Cable Damage curve / Motor starting curve*). A dialog with options for configuring the damage curve will appear. See Sections [33.4.10.2](#), [33.4.10.3](#) and [33.4.10.4](#).

### 33.4.10.2 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.

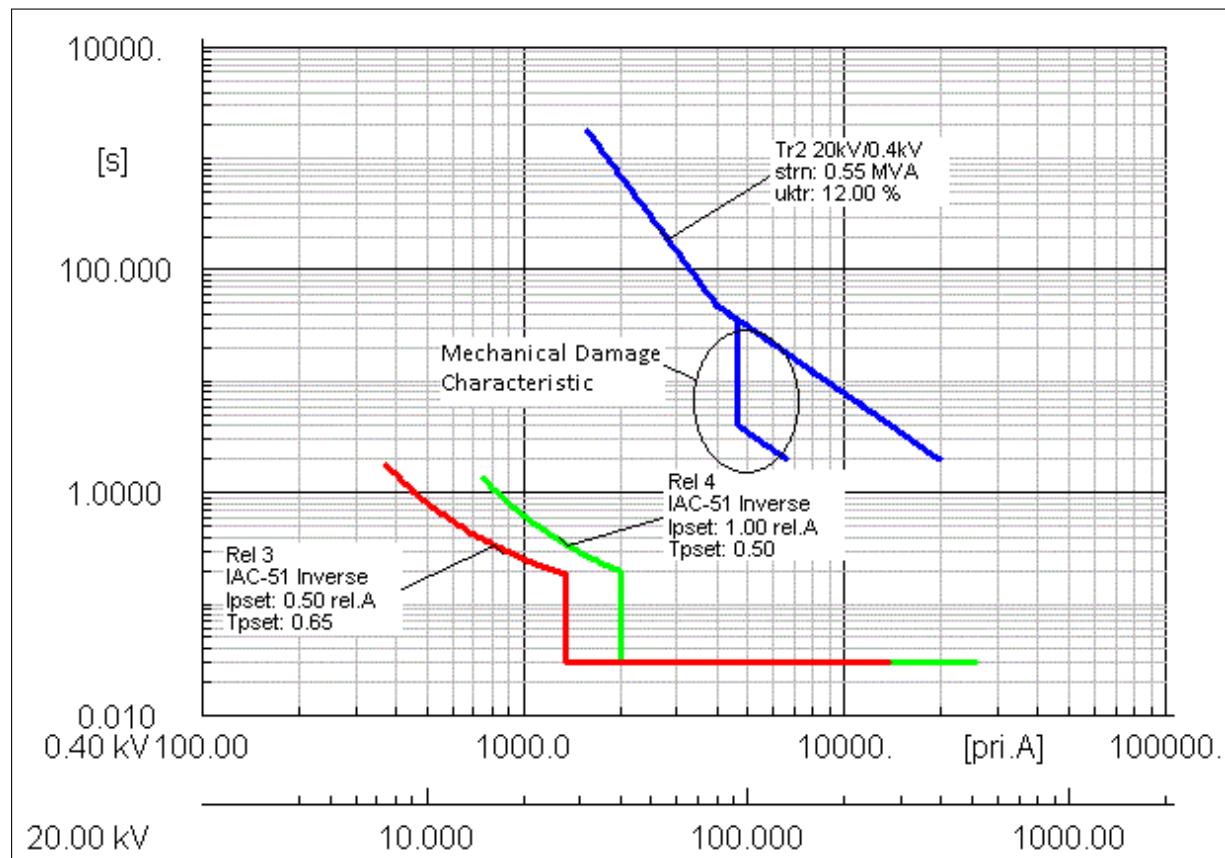


Figure 33.4.6: Transformer damage curve

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	$S_{rat} \leq 0.5\text{MVA}$	$S_{rat} \leq 0.5\text{MVA}$
Category II	$S_{rat} \leq 5.0\text{MVA}$	$S_{rat} \leq 1.667\text{MVA}$
Category III	$S_{rat} \leq 30.0\text{MVA}$	$S_{rat} \leq 10.0\text{MVA}$
Category IV	$S_{rat} > 30.0\text{MVA}$	$S_{rat} > 10.0\text{MVA}$

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:

$$I_1 = I_{rat} \cdot \frac{1}{u_k}; t_1 = 2,0s \quad (33.2)$$

$$I_2 = I_{rat} \cdot \frac{c_f}{u_k}; t_2 = \frac{K}{I_2^2} = \frac{I_1^2 \cdot t_1}{I_2^2} = \frac{2,0s}{c_f^2} \quad (33.3)$$

$I_3 = I_2$ ;  $t_3$  = 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.

#### 33.4.10.3 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** [27]:

$$k = c_1 \cdot \sqrt{\ln\left(1 + \frac{\theta_f - \theta_i}{c_2 + \theta_i}\right)} \quad (33.10)$$

**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>	<b>90</b>	<b>250</b>

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.

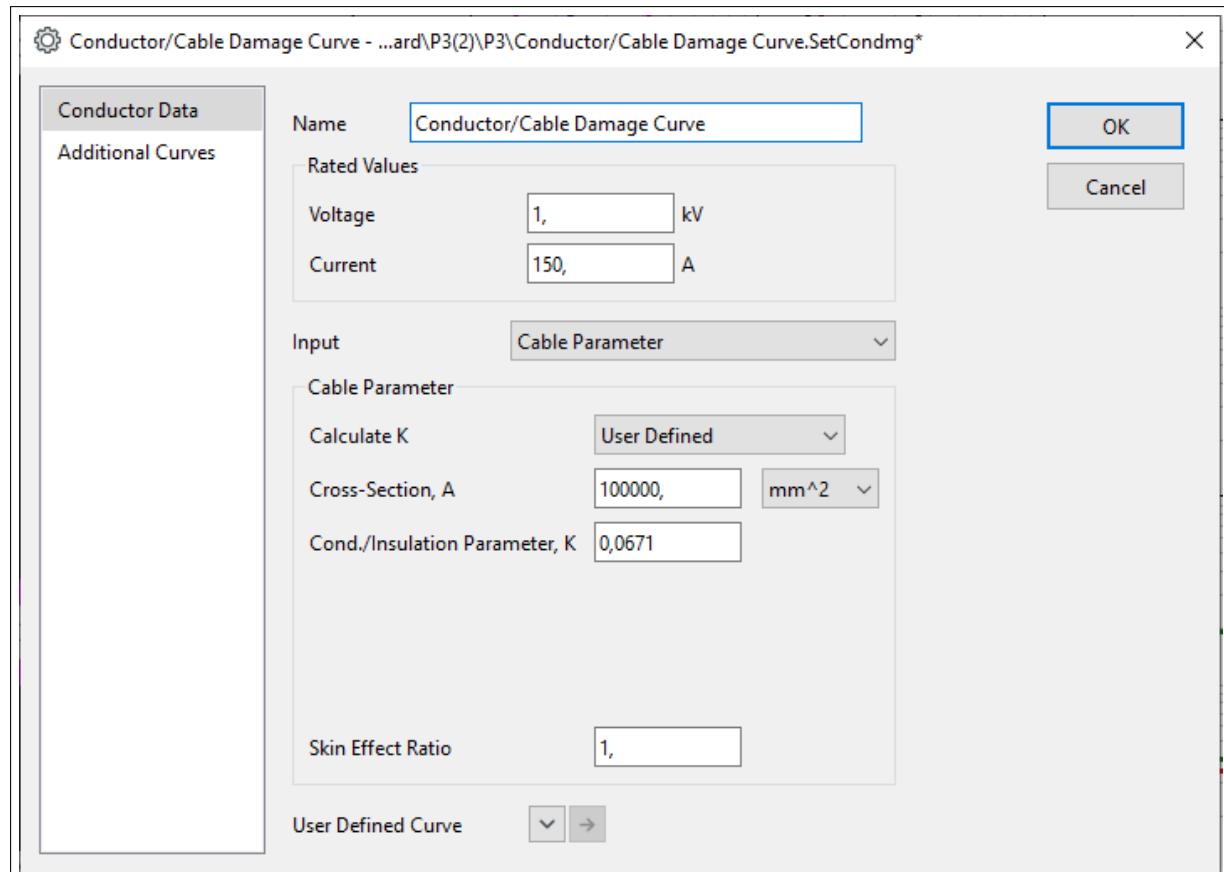


Figure 33.4.7: Conductor/Cable damage curve

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:

$$\frac{I_E}{I_N} = \sqrt{\frac{1 - \left(\frac{I_0}{I_N}\right)^2 \cdot e^{-\frac{t}{k}}}{1 - e^{-\frac{t}{k}}}} \quad (33.13)$$

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]

#### 33.4.10.4 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.

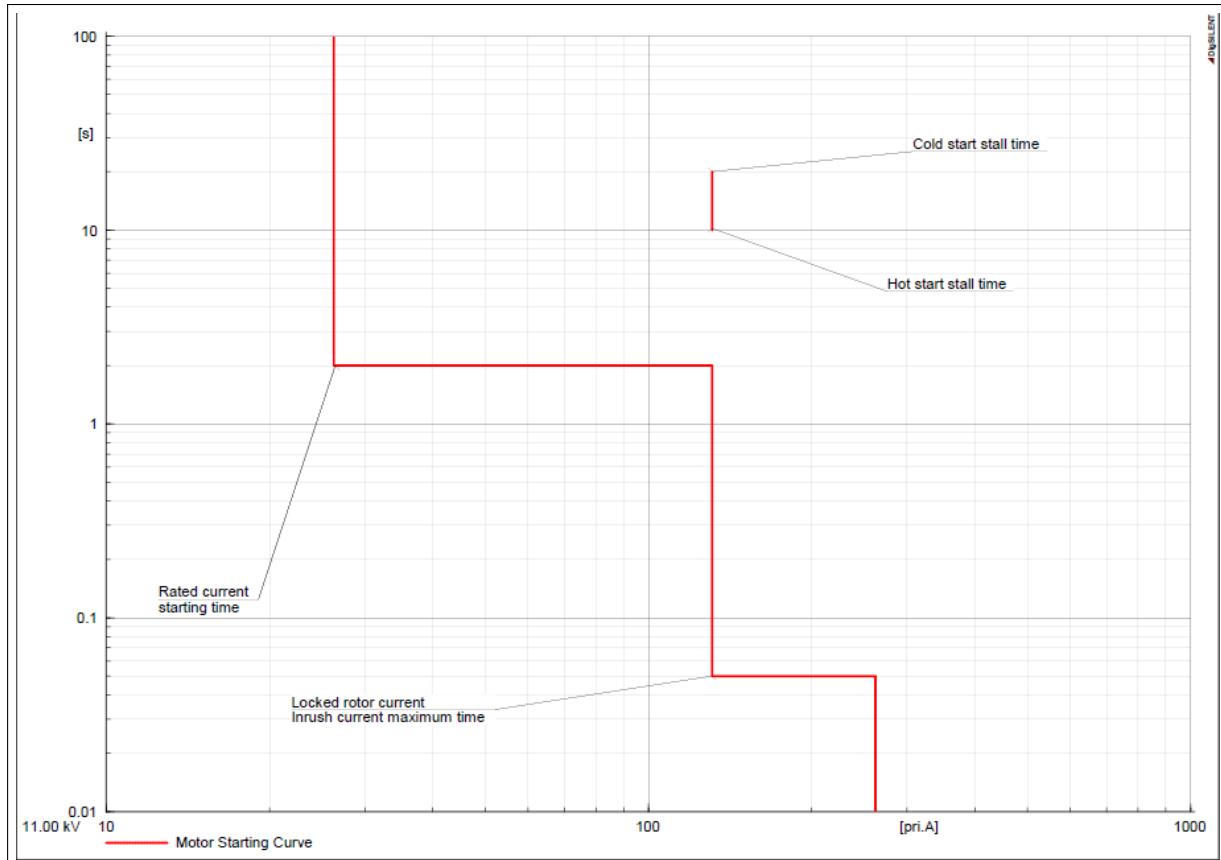


Figure 33.4.8: The motor start curve

#### 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]
$\frac{I_{tr}}{I_{nom}}$	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 section of the [Technical References Overview Document](#).

#### 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.

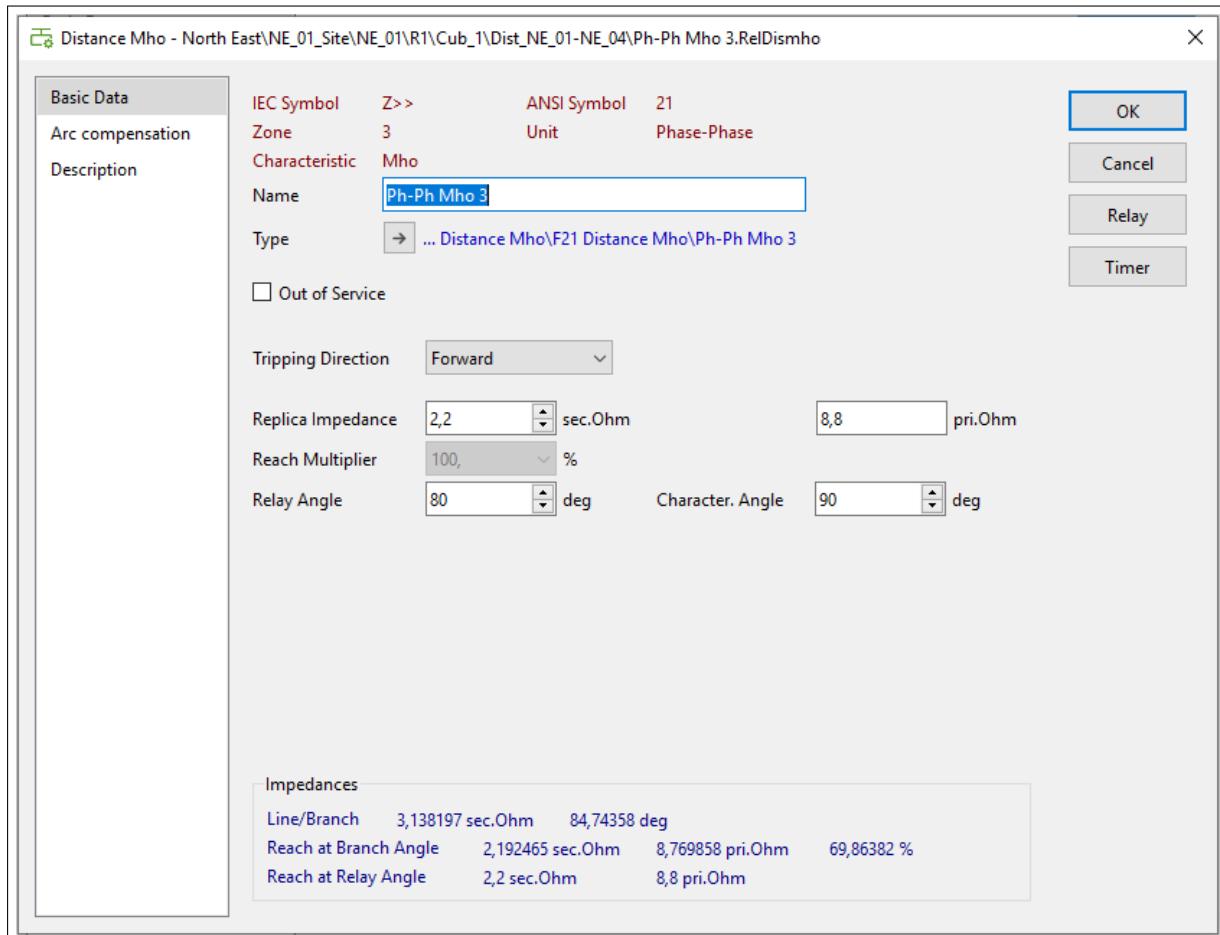


Figure 33.5.1: Distance mho block

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 -R resistance and the Rt 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.

---

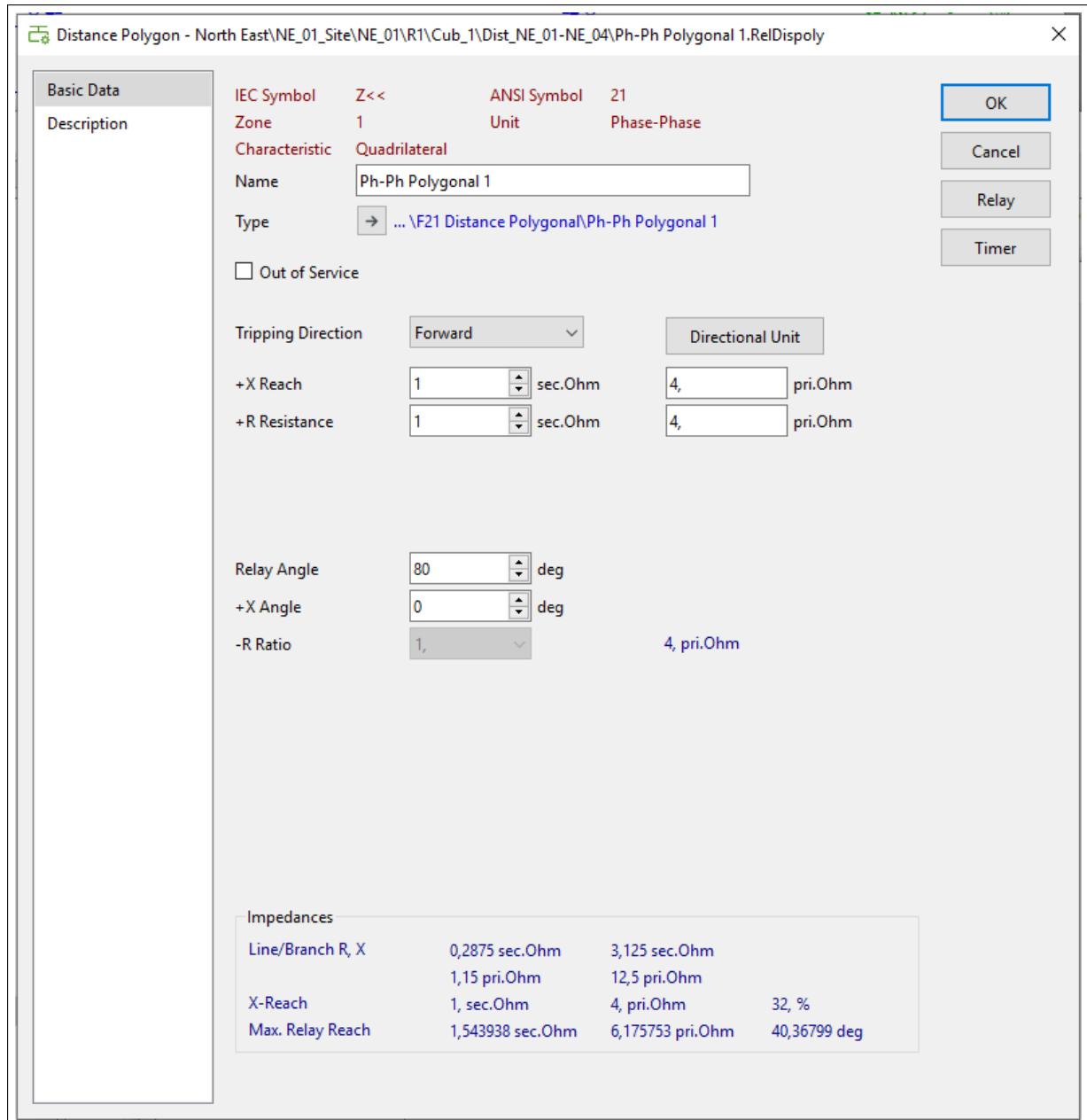


Figure 33.5.2: Distance polygon block (Polygonal (Beta))

### 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.

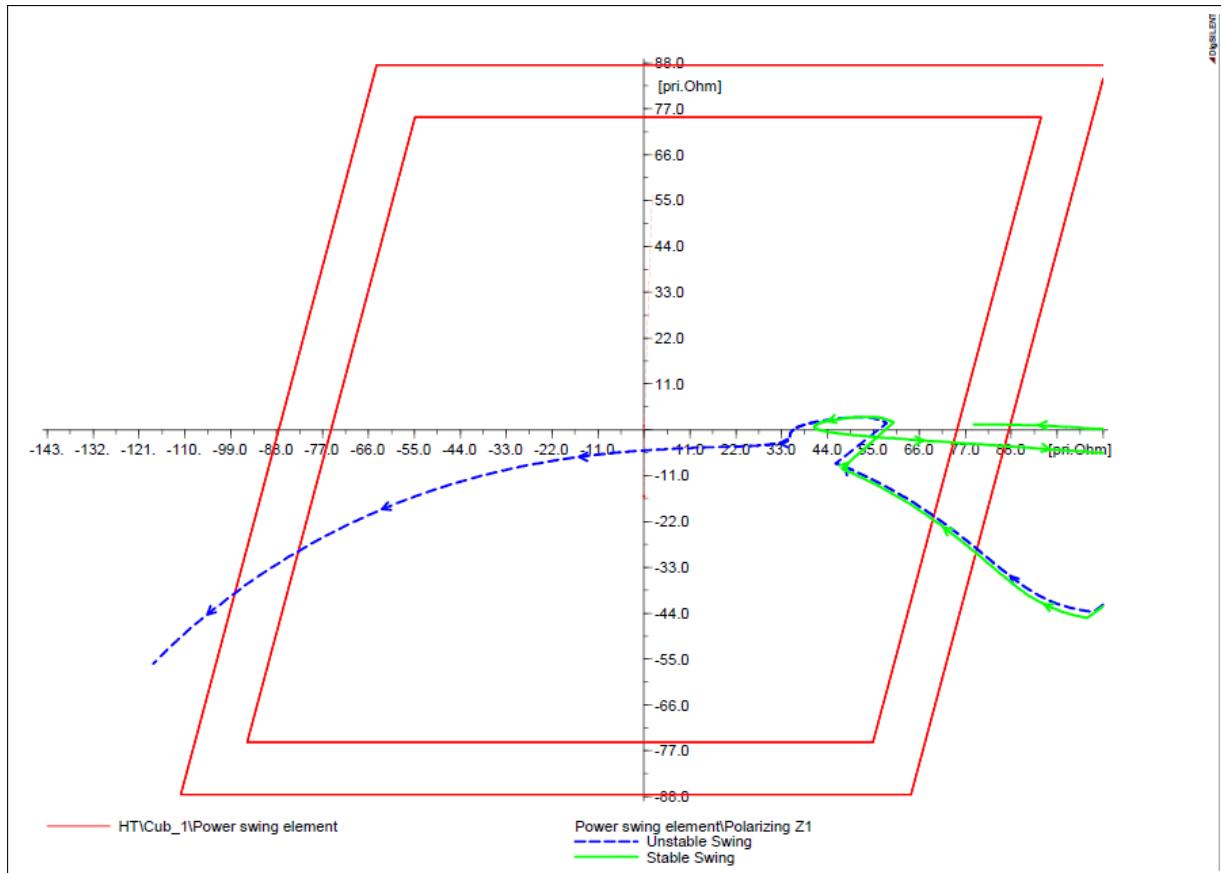


Figure 33.5.3: Stable (green) and unstable (blue) power swings

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 sensitive menu will appear.
- (b) Select the option *Create 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 sensitive menu will appear.
- (b) Select the option *Show → 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 sensitive menu will appear.
- (b) Select *Path... → R-X Plot* from the context-sensitive menu. *PowerFactory* will create an R-X diagram on a new page showing the active characteristics for all relays in the selected path.

In the first two methods, it is also possible to select the option *Add to R-X Plot* instead of *Show → R-X Plot*. This will open a list of previously defined R-X Plots from which any one can be selected to add the selected device characteristics to.

### 33.6.2 Understanding the R-X diagram

An example R-X diagram with two relays is shown in Figure 33.6.1. Shown on the plot is:

- The active zone impedance characteristics for each relay.
- The impedance characteristic of the network near the relay location - shown as a dashed line.
- The location of other 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 bolts on the plot and also as values within the coloured legends).
- The detected fault type as determined by the starting elements (shown in the coloured legend). Note, this is not enabled by default, see Section 33.6.3.5 for instructions how to enable this.
- The tripping time of each zone (shown in the coloured legend). Note this is not enabled by default, see Section 33.6.3.5 for instructions how to enable this.
- The overall tripping time of each relay (shown in the coloured legend).

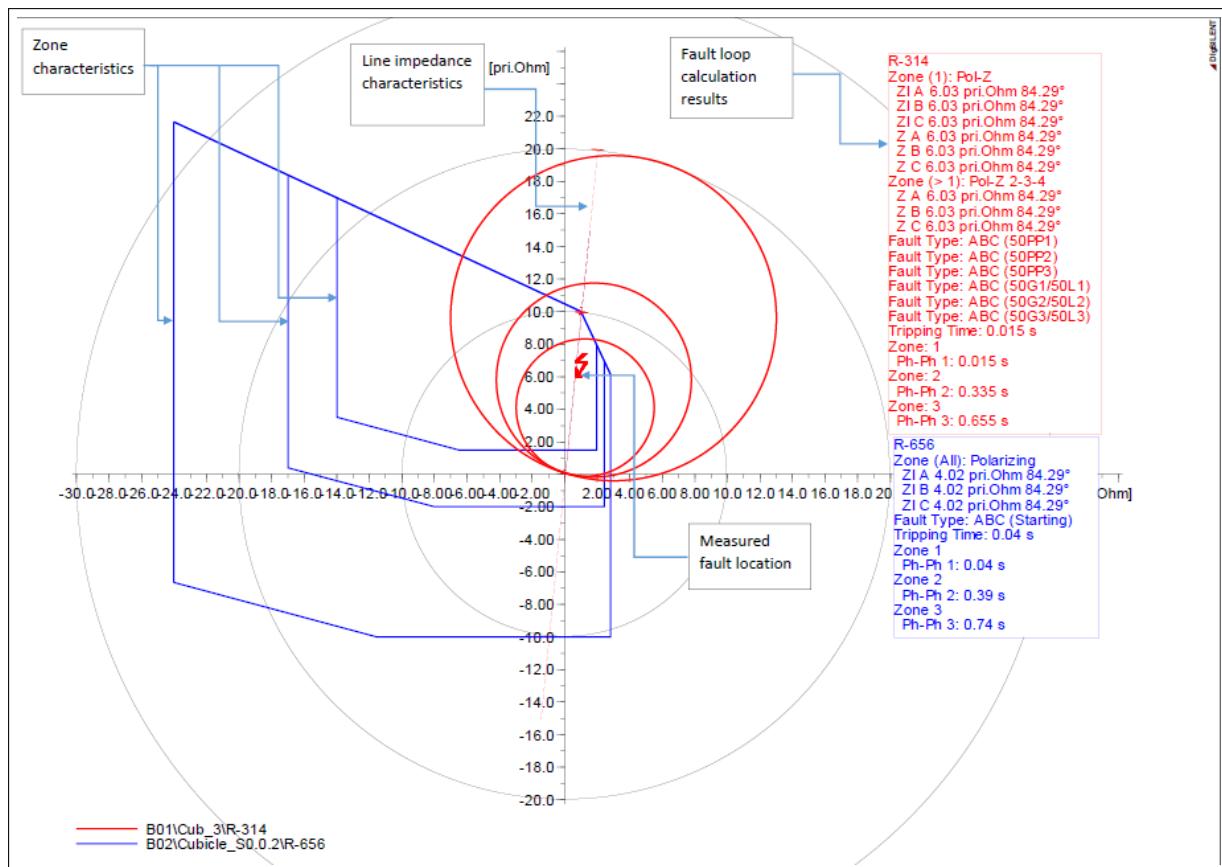


Figure 33.6.1: A R-X plot with short-circuit results and two relays

Note the information shown on the plot can be configured by altering the settings of the R-X plot. Refer to Section 33.6.3.

### 33.6.3 Configuring the R-X plot

There are several ways to alter the appearance of the R-X diagram. Many configuration parameters can be adjusted by right-clicking the plot and using the context sensitive menu. Alternatively, double-clicking

the plot avoiding the selection of any characteristics showing on the plot will show the plot dialog.

The following sections explain the various ways to alter the display of the plot.

### 33.6.3.1 Adjusting the grid lines in the R-X diagram

To change the grid settings in the R-X diagram:

1. Right-click the R-X diagram. A context sensitive menu will appear.
2. Select *Grid*. The grid options dialog will appear.
3. Select the *Layout* page.
4. To enable grid lines on the major plot divisions, check *Main*.
5. To enable grid lines on the minor plot divisions, check *Help*.

### 33.6.3.2 Changing the position of the R-X plot origin

Section 33.6.3.4 explains how the limits and size of the R-X diagram can be altered in detail. However, it is also possible to reposition the origin of the plot graphically. To do this:

1. Right-click the R-X diagram exactly where you would like the new origin (0,0) point of the plot to be. A context sensitive menu will appear.
2. Select *Set origin*. *PowerFactory* will reposition the origin of the plot to the place that you right-clicked.

### 33.6.3.3 Centring the origin of the R-X plot

To centre the origin (0,0) of the plot in the centre of the page:

1. Right-click the R-X diagram. A context sensitive menu will appear.
2. Select *Centre origin*. *PowerFactory* will reposition the origin of the plot to the centre of the page.

### 33.6.3.4 R-X plot basic data page

The tabular area at the top of the dialog shows the currently displayed relays, and the colours, line styles and line widths that are used to represent them on the plot. Each of these can be adjusted by double-clicking and selecting an alternate option. Refer to Section 19.7 for more information on configuring plots in *PowerFactory*.

The *Axis* area at the bottom of the dialog shows the settings that are currently used to scale the axes on the plot. These settings and their effect on the plot is explained further in the following section.

- **Scale.** This number affects the interval between the tick marks on the x and y axis, in the units specified in the *Unit* field. If the Distance (see below) field remains constant, then increasing this number increases the size of the diagram and effectively zooms out on the displayed characteristics.
- **Distance.** This number affects the distance in mm between each tick mark. Remember that in *PowerFactory* it is usual for plots and diagrams to be formatted in a standard page size (often A4). Consequently, this number has the opposite effect of the scale - when the scale field is constant increasing the distance effectively zooms into the displayed characteristics.

- **x-Min.** This field determines the left minimum point of the diagram. However, it also implicitly considers the scale. Consequently, the true minimum is determined by the product of the *Scale* and *x-Min*. For example, if the scale is 4 and *x-Min* is set to 2, then the minimum x axis value (resistance) displayed would be -8.
- **y-Min.** The concept for *y-Min* is the same as *x-Min* with the minimum value determined by the product of the scale and the specified minimum value.

**Note:** The user can ask *PowerFactory* to adjust the scale of the R-X diagram automatically based on the set Distance. Click **Characteristics** to adjust the scale automatically to fit all the displayed characteristics, or click **Impedances** to adjust the scale to fit all displayed network impedances.

### 33.6.3.5 R-X plot options

The R-X plot advanced settings can be accessed by right-clicking the plot and selecting *Options* from the context-sensitive menu, or by pressing the **Options** button in the edit dialog of the plot.

The options dialog has the following settings:

- **Unit.** This option affects whether the characteristics on the plot are displayed in primary or secondary (relay) Ohm. It is also possible to select % of line which will display all characteristics in terms of a % impedance of their primary protected branch. This latter option is quite useful for visualising inspecting that the zone settings are as expected.
- **Relays Units.** This option is used to display only certain types of relay characteristics. For example, it is possible to display only earth fault distance characteristics by selecting the option *Ph-E*.
- **Zones.** This setting affects what zones are displayed. For example, to only show zone 1 characteristics, "1" should be selected.
- **Starting.** This checkbox configures whether starting elements 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 the line style for the displayed part of the characteristic that is not normally visible.
- **Display.** This option is used to select how the calculated load-flow or short-circuit current/equivalent impedance will be displayed. The options are a short-circuit *Arrow*, a *Cross* or to *Hide* it completely.
- **Zone.** User has an option to select what zone relevant results should be displayed in plot and in legend.
- **Show Impedance.** Different impedance values can be selected here: all, only smallest,  $Z < 1.5 \cdot Z_{min}$  and  $Z < 5 \cdot Z_{min}$ .
- **Force Positive Sequence Representation.** By selecting this check box R-X plot will be showing all measured impedance and measuring zones in the positive sequence representation only. This make it easier to coordinate between relays of different manufacturers as well as coordinate between measured impedance and measuring zones.

- **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.
- **Default Length for Blinder Units.** This options specifies the length of blinder units on the plot in secondary Ohms.

### Branch impedances page

This page specifies how the branch impedance elements are displayed on the diagram:

- **Number of Relay Locations.** Only the branches are shown up to the specified number of relay locations. If zero, no branches are shown at all.
- **Branches, max. Depth.** Maximum number of branches shown from each relay location. If zero, no branches are shown at all.
- **Ignore Transformers.** Transformer impedances are ignored when activated.
- **Method.** 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*, effectively completes a short circuit sweep similar to that described in Section 33.8.3 except that 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.
- **Show Branch Options.** Here the line style and width can be selected.

### Legend page

This page determines the configuration of the coloured legend visible after a short circuit or load-flow calculations for each relay on the R-X diagram.

The following options are available:

- **Show Calculated Impedances.** Determines whether the impedances calculated for each fault loop by the polarising block will be displayed in the legend.
- **Detected Fault Type.** Determines whether the fault type calculated by the starting element will be displayed in the legend.
- **Tripping Time of Relay.** Determines if the overall tripping time of the relay will be displayed in the legend.
- **Tripped Zones.** Determines if the tripping time of each zone that trips will be displayed in the legend.

### 33.6.4 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.

## 33.7 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 [19.7](#).
  - (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 symbols 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.

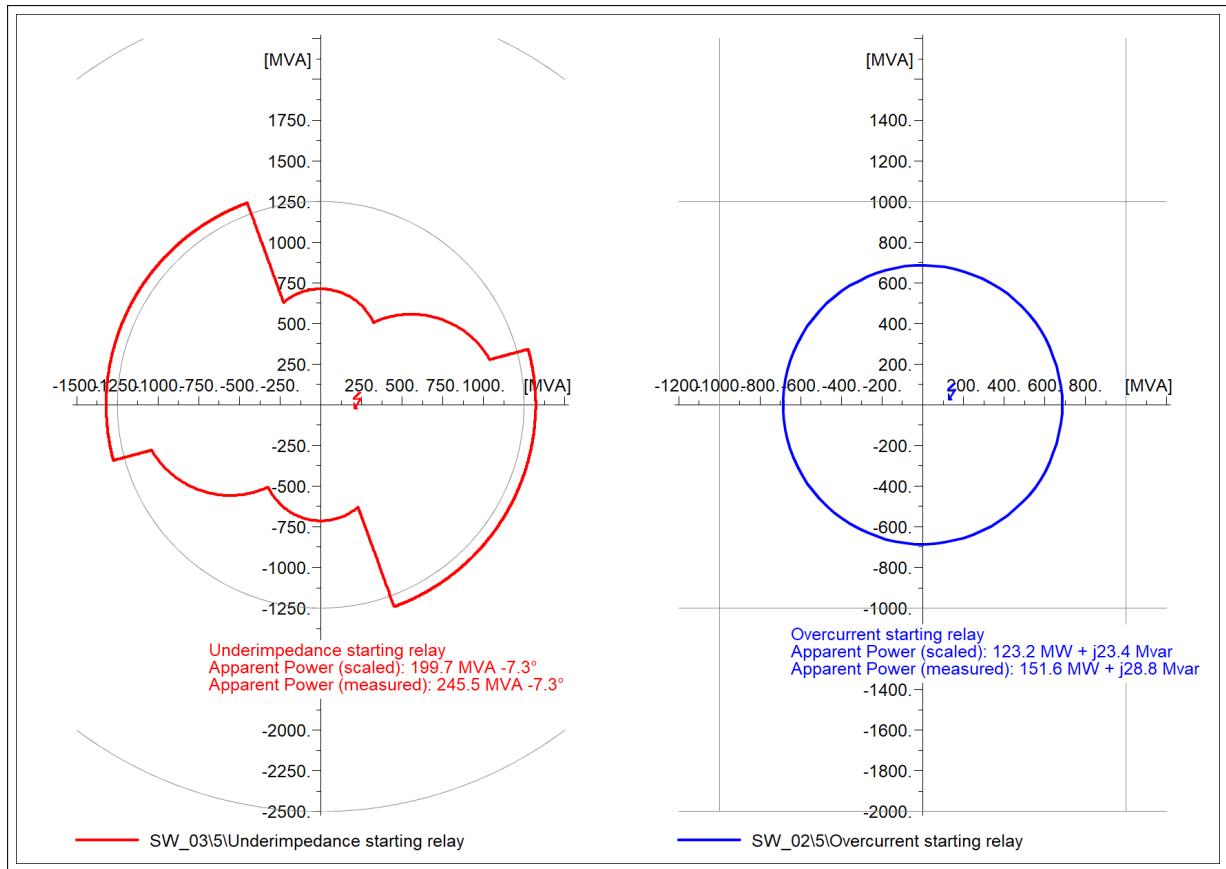


Figure 33.7.1: A P-Q diagram plot with load flow results and starting characteristics for two separate relays

Note the information shown on the plot can be configured by altering the settings of the P-Q plot. Refer to Section 33.7.3).

### 33.7.3 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 the plot and using the context sensitive menu. Alternatively, double-clicking in empty space in the plot will show the plot dialog.

The following sections explain the various ways to alter the display of the plot.

#### 33.7.3.1 Changing the position of the P-Q plot origin

Section 33.7.3.5 explains how the limits and size of the P-Q diagram can be altered in detail. However, it is also possible to reposition the origin of the plot graphically. To do this:

1. Right-click the P-Q diagram exactly where you would like the new origin (0,0) point of the plot to be. A context sensitive menu will appear.
2. Select *Set origin*. PowerFactory will reposition the origin of the plot to the place that you right-clicked.

### 33.7.3.2 Centring the origin of the P-Q plot

To centre the origin (0,0) of the plot in the centre of the page:

1. Right-click the P-Q diagram. A context sensitive menu will appear.
2. Select *Centre origin*. *PowerFactory* will reposition the origin of the plot to the centre of the page.

### 33.7.3.3 Switching from polar to Cartesian coordinates

To switch from polar to Cartesian coordinates or viceversa:

1. Right-click the P-Q diagram. A context sensitive menu will appear.
2. Select *Polar*. *PowerFactory* will select or deselect the option and coordinate system will change correspondingly.

### 33.7.3.4 P-Q plot relays page

The tabular area in the plot dialog entitled *relays* lists the displayed relays along with the colours, line styles and line widths that are used to represent their characteristics in the plot. Each of these can be adjusted by double-clicking and selecting an alternate option. Refer to Section 19.7 for more information on configuring plots in *PowerFactory*.

### 33.7.3.5 P-Q plot scale page

- **Scale.** This variable affects the interval between the tick marks on the x and y axis, in the units specified in the *Unit* field. If the Distance (see below) field remains constant, then increasing this number increases the size of the diagram and effectively zooms out on the displayed characteristics.
- **x-Min.** This field determines the left minimum point of the diagram. However, it also implicitly considers the scale. Consequently, the true minimum is determined by the product of the *Scale* and *x-Min*. For example, if the scale is 4 and *x-Min* is set to 2, then the minimum x axis value displayed would be -8.
- **y-Min.** The concept for *y-Min* is the same as *x-Min* with the minimum value determined by the product of the scale and the specified minimum value.

**Note:** The user can ask *PowerFactory* to adjust the scale of the P-Q diagram automatically. Click **Scale** to adjust the scale automatically to fit all the displayed characteristics

### 33.7.3.6 P-Q plot options

Specific option relevant to the P-Q plot settings can be accessed by right-clicking the plot and selecting *Options* from the context-sensitive menu, or by pressing the **Options** button in the edit dialog of the plot.

The options dialog has the following settings:

- **Power Calculation > Voltage factor.** This option influences the voltage factor used to scale the characteristic and scale the load flow calculation result as also described and illustrated in section 33.7.2

- **Diagram marks > Show calculated PQ values.** This option determines whether or not to display a lightning bolt in the plot indicating the scaled load flow result observed at the relaying location following a successful load flow calculation. See section [33.7.2](#) for an example.
- **Legend > Show calculated PQ values.** This option determines whether or not to display a legend quantifying the scaled and unscaled load flow result observed at the relaying location following a successful load flow calculation. See section [33.7.2](#) for an example.

### 33.7.4 Modifying the starting element settings from the R-X plot

From the R-X 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 R-X diagram.

## 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 diagram 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 diagram 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 sensitive menu select the option *Show → Time-Distance Diagram*. 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 *Show... → Time-Distance Diagram* from the context sensitive menu. As above, this will create a new object *VisPlottz*.
- 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 *Show → Time-Distance Diagram* from the context sensitive menu.

### 33.8.1 Forward and reverse plots

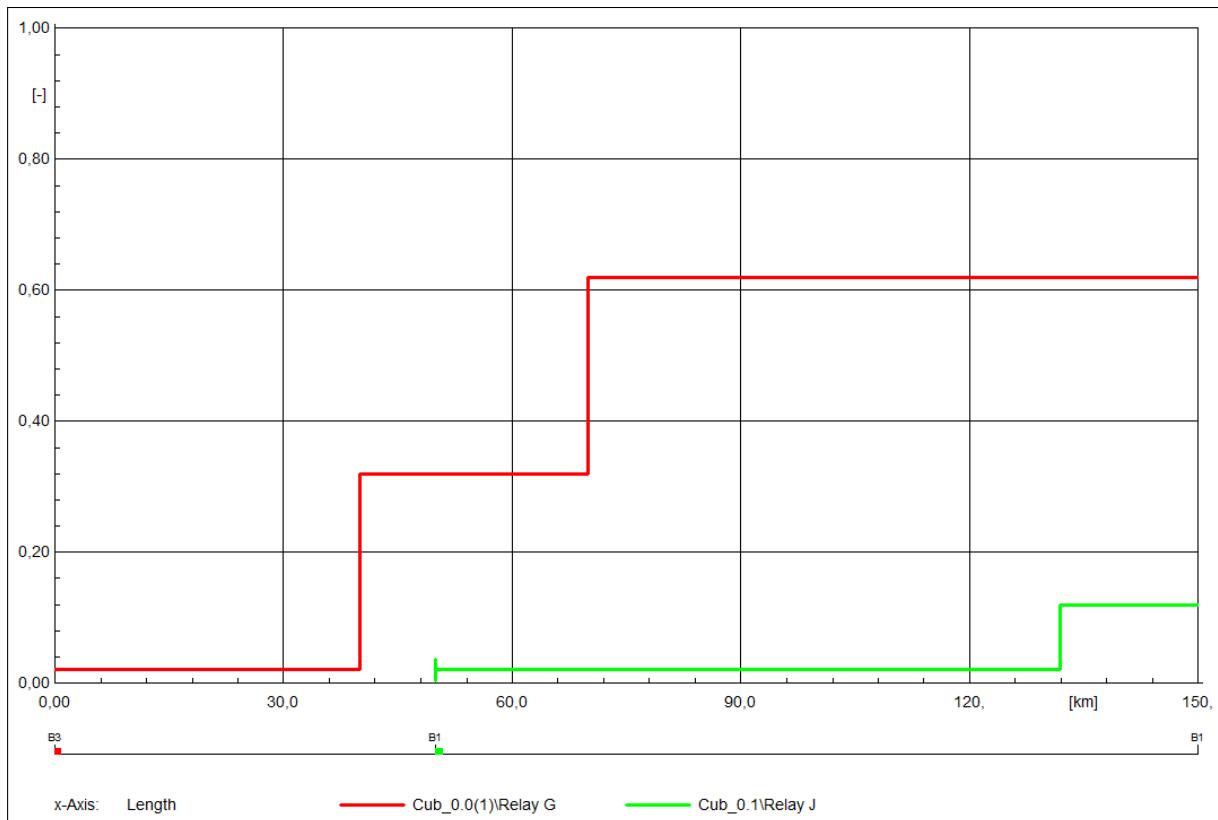


Figure 33.8.1: A forward time-distance plot

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 diagram 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

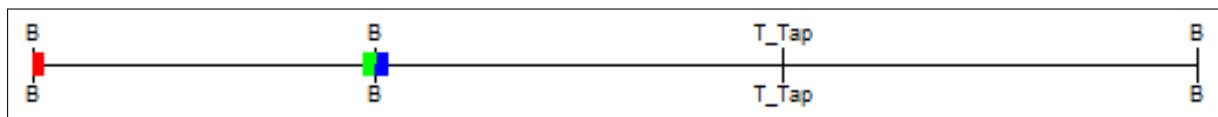


Figure 33.8.2: A path axis

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 sensitive 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 [33.13.8.2](#) 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 Iterate tripping times 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 *Continuous 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.
- **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.

### 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 Diagram 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 Relay 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 diagram. 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 sensitive menu

If the diagram is right-clicked at any position, the context sensitive menu will pop up similar to the plot menu described in Chapter 19: Reporting and Visualising Results, Section 19.7 (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 section of the [Technical References Overview Document](#).

#### 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 *VisPcompdifffplt* ( 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

- (a) Right-click a cubicle containing differential relays. The context sensitive menu will appear.
- (b) Select the option *Show → Differential Protection Plot* or *Show → In existing Protection 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

- (a) 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.
- (b) Right click the  icon. A context sensitive menu will appear.
- (c) Select *Show → Differential Protection Plot* or *Show → In existing Protection Plot*.

### 33.10.1 Magnitude biased differential diagram

This diagram (.VisMagndiffplt) 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)

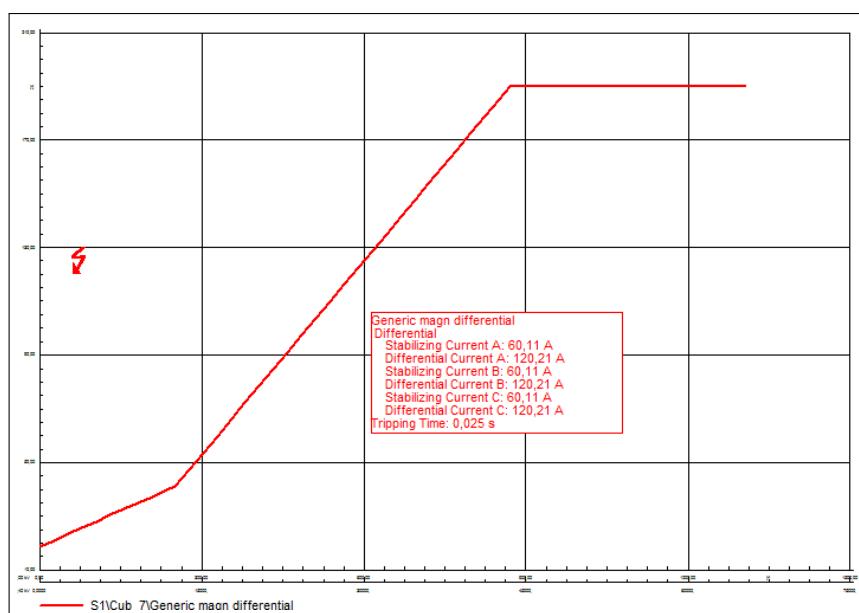


Figure 33.10.1: PowerFactory magnitude biased differential diagram for 3ph differential protection element

### 33.10.2 Phase comparison differential diagram

A phase comparison differential diagram (.VisPcompdiffplt) 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.

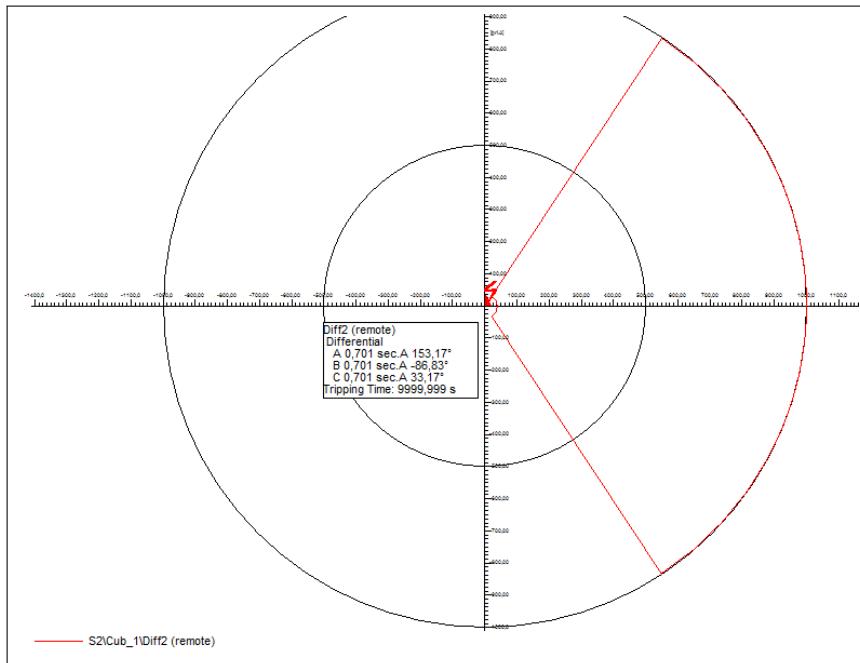


Figure 33.10.2: Example of *PowerFactory* Phase comparison differential diagram for differential protection element of the type 3ph Phase Comparison

## 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, reactance in primary ohms or reactance in 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.

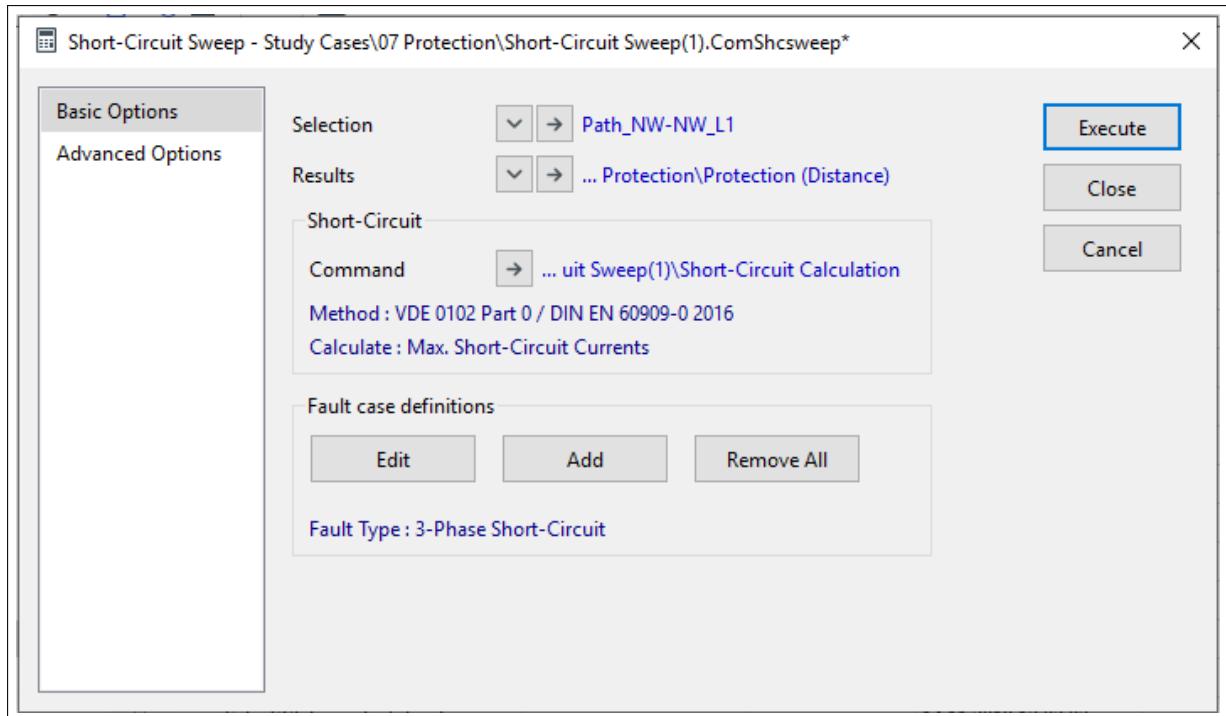


Figure 33.11.1: Short-Circuit Sweep command

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.

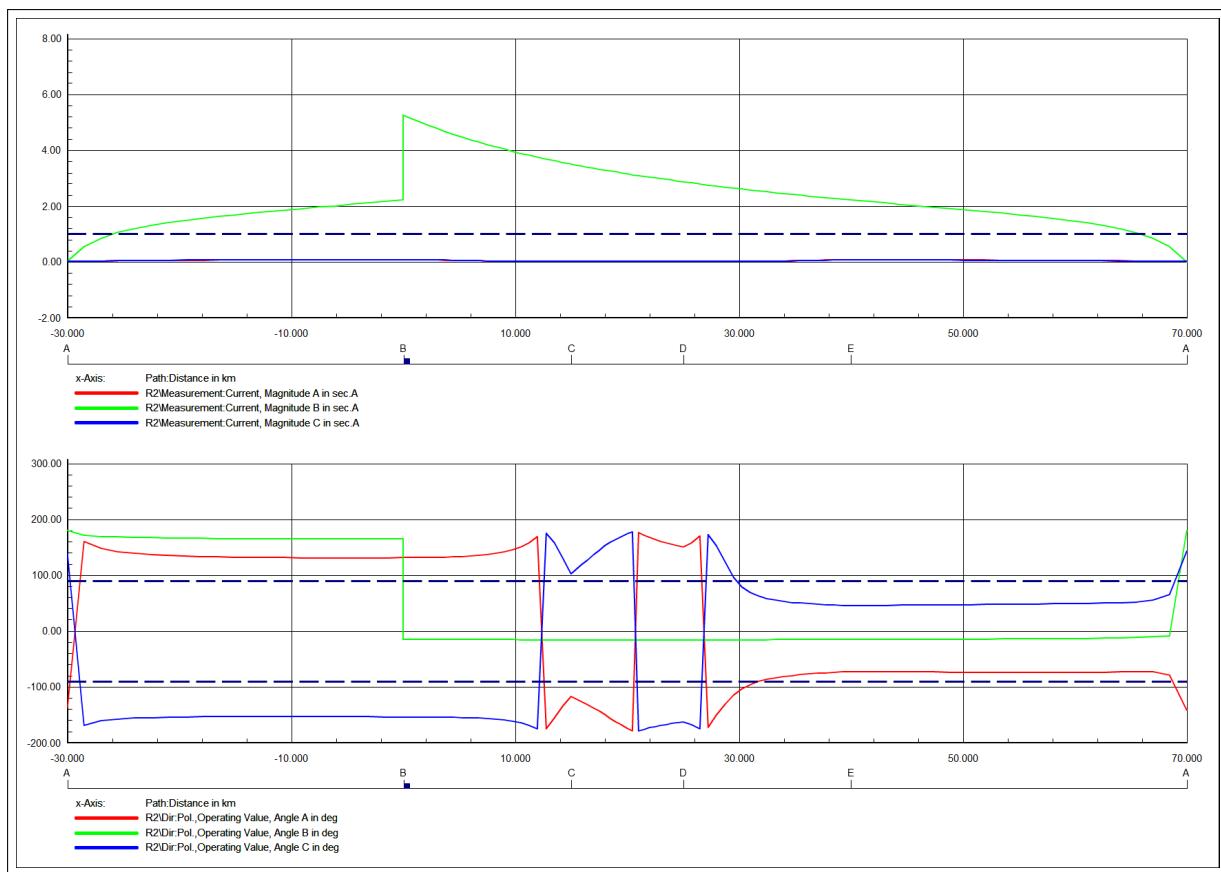


Figure 33.12.1: Typical Short-Circuit Sweep Plots

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 or reactance, 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 path object in the data manager and choosing *Show* → *Short-circuit sweep plot*, or by right clicking on an element in the single line diagram belonging to a path and choosing *Show* → *Short-circuit sweep plot*. 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.

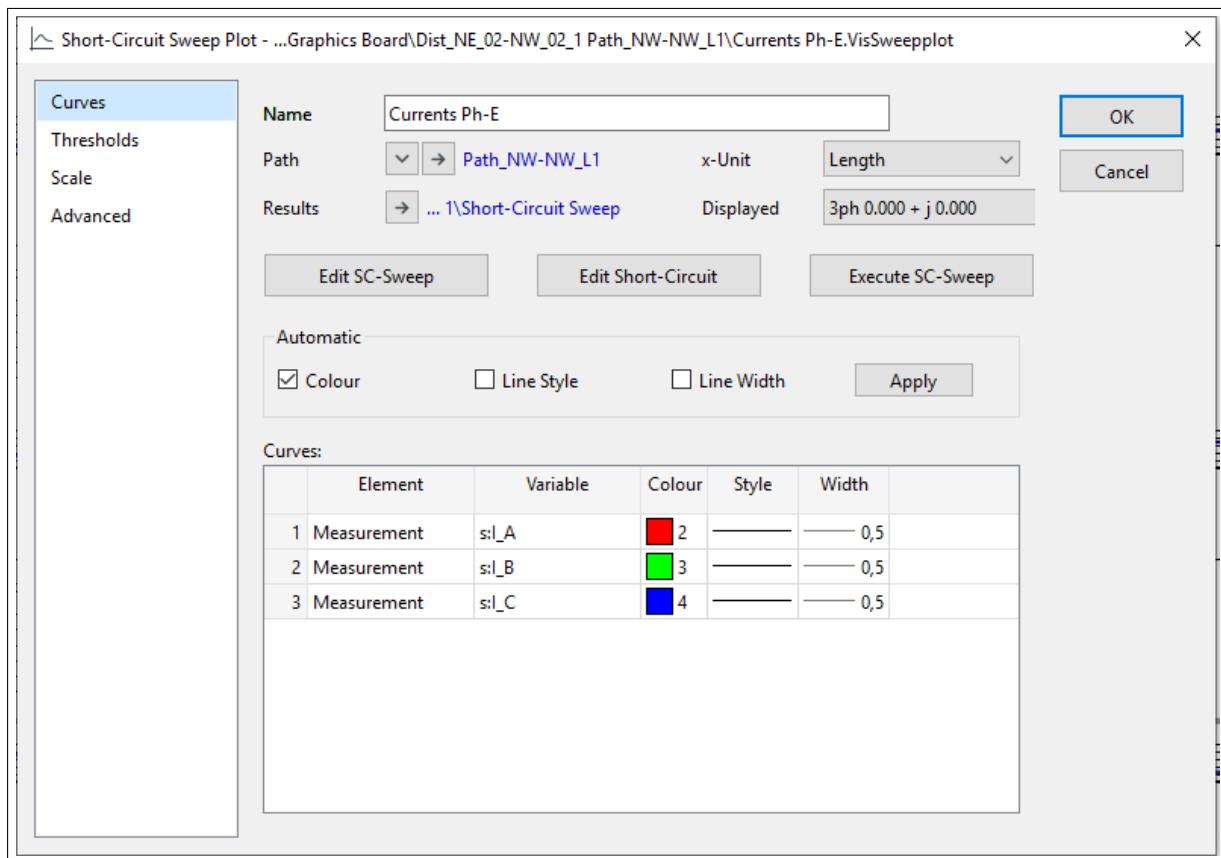


Figure 33.12.2: Short-Circuit Sweep Plot - Curves page

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 or reactance. 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.

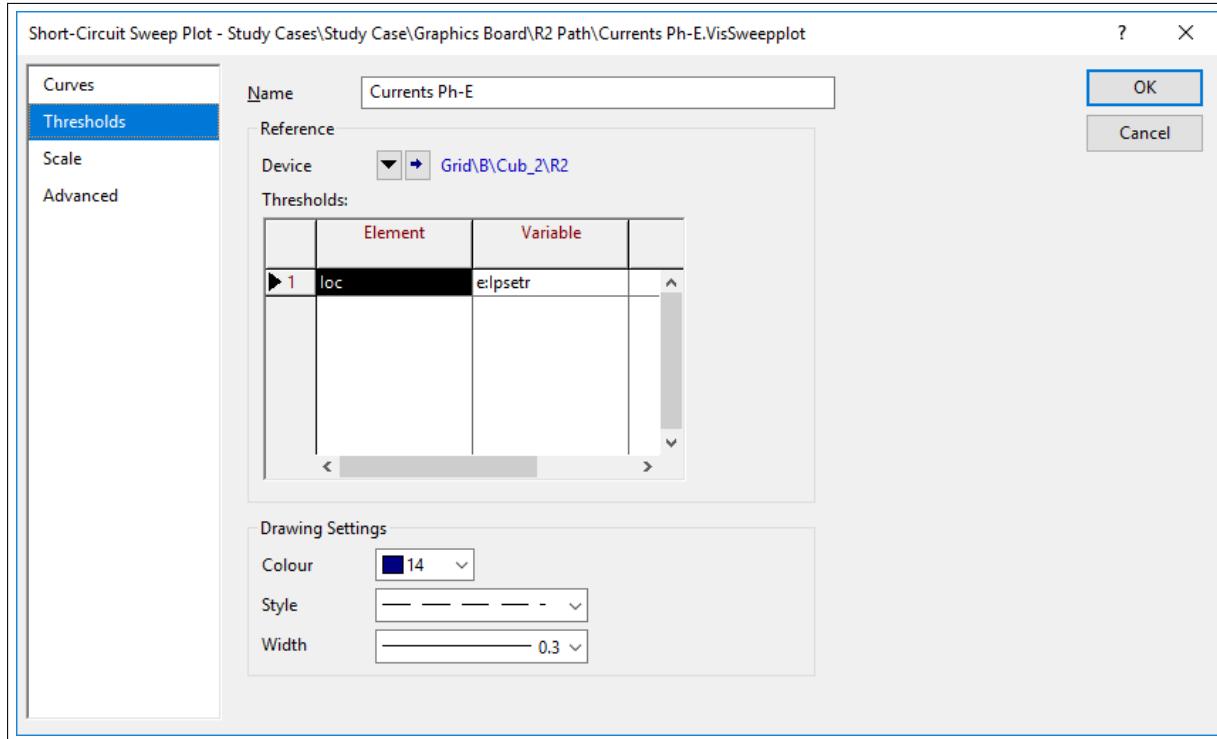


Figure 33.12.3: Short-Circuit Sweep Plot - Thresholds page

## 33.13 Protection coordination assistant

*PowerFactory* includes a protection coordination assistant that can assist with automatically determining settings for distance (impedance based) protection relays. This section explains how to use this tool.

### 33.13.1 Protection coordination assistant - technical background

This section provides a brief overview of 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 distance protection 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 so-called “polarising” voltage and “operating” current quantities are used to measure the impedance and 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.

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.

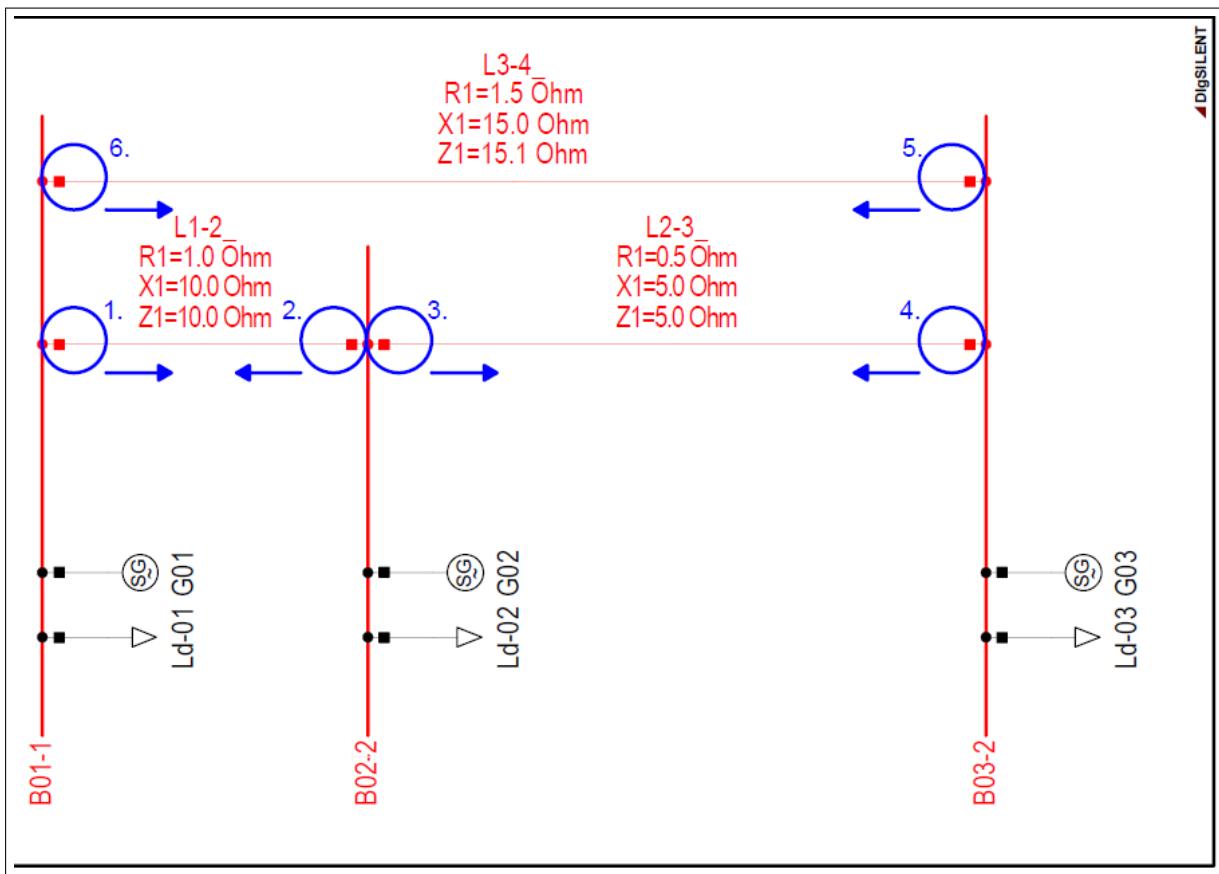


Figure 33.13.1: Example simplified transmission system for the protection coordination example

For the purposes 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 Protection coordination assistant - 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. The network impedance data.
2. The network topological connection data.
3. The relay locations.
4. The protection philosophy distance protection setting rules

The first two items are usually already available in the network model as this data is required for all standard calculations such as for example load flow calculations or short circuit calculation.

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 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 Protection Coordination Assistant Command - Basic Options

This page of options is illustrated in figure [33.13.2](#).

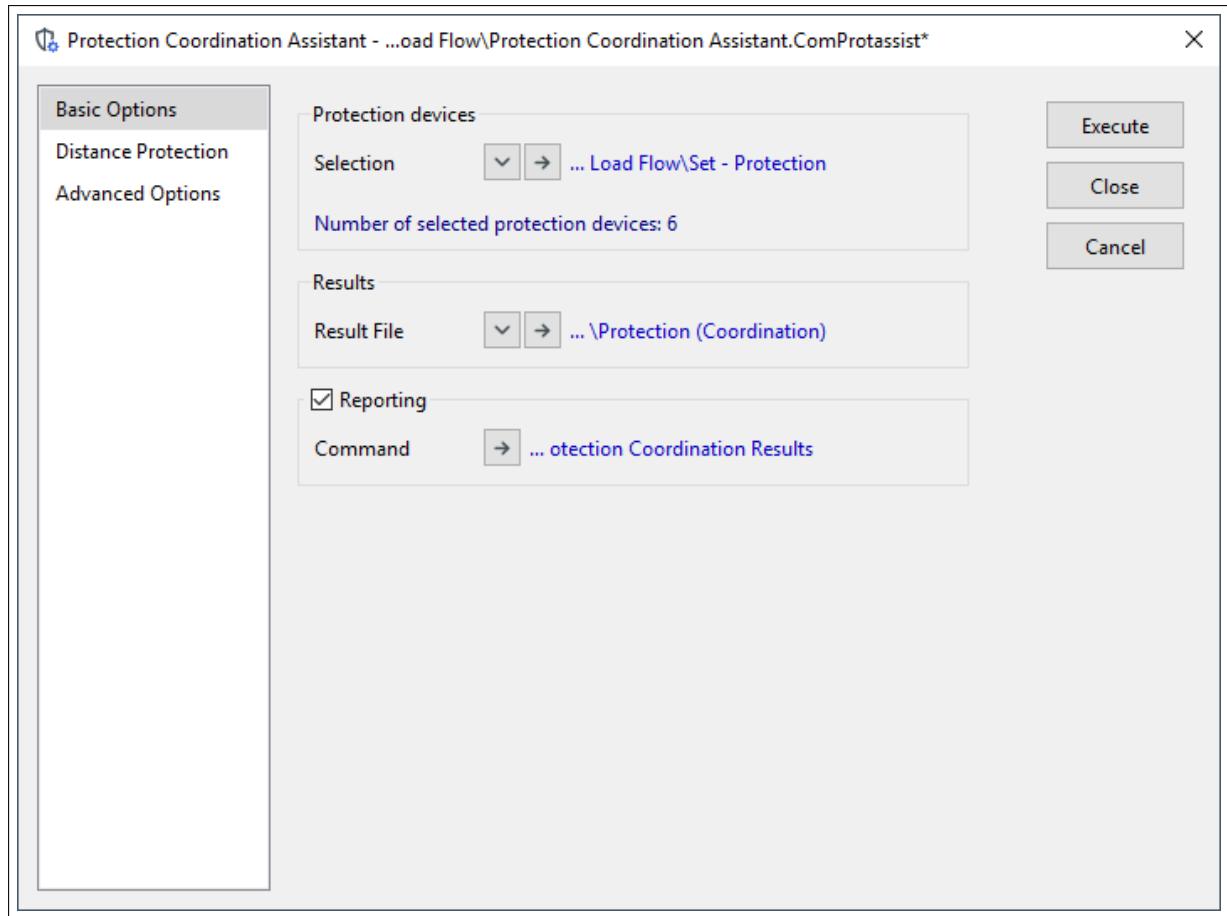


Figure 33.13.2: Protection coordination assistant command - Basic Options page

The *Selection* parameter in the *Protection devices* field is used to specify the protection relays for which the coordination will be carried out. The devices are selected by clicking on the  $\downarrow$  icon. Devices can either be selected directly or a grouping which contains devices, such as a set or a path can be selected.

The *Result File* parameter in the *Results* field is used to specify the result file in which the results of the coordination will be stored. 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  $\downarrow$  icon. The result file dialog itself can be accessed by clicking the  $\rightarrow$  icon.

The *command* parameter in the *Reporting* field is used to specify 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 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* field used for the second step. 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 Protection Coordination Assistant Command - 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 33.13.3.

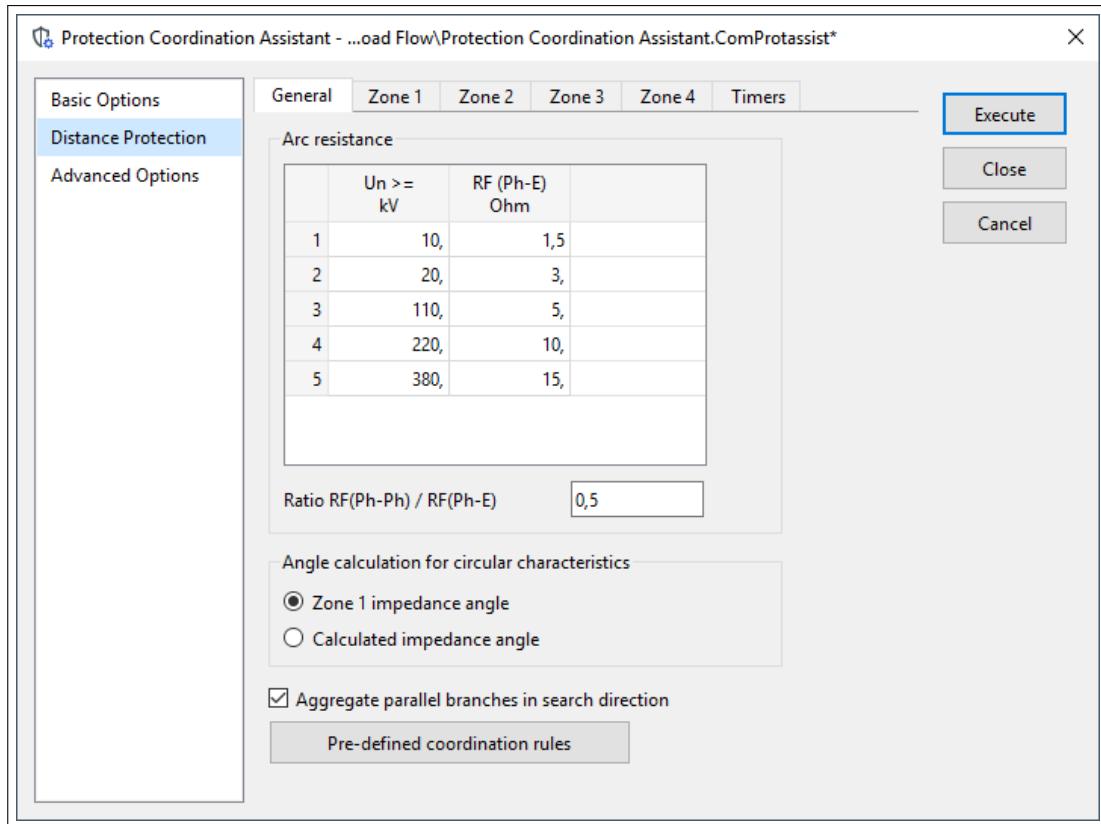


Figure 33.13.3: Protection coordination assistant command - General tab of Distance Protection page

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 33.13.4.3). 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.

### 33.13.4.1 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 33.13.4 is displayed.

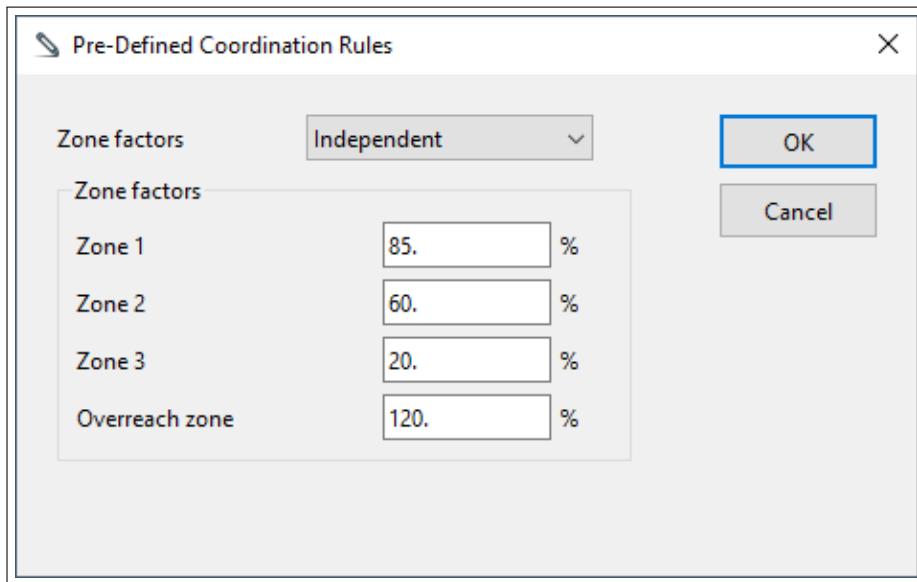


Figure 33.13.4: Pre-Defined Coordination Rules dialog

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 33.13.4. Reach equations are defined on a per zone basis with zone 1b representing the overreach zone. Section 33.13.4.3 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:

$$X_2 = ZFac_1 * X_{maxAB} + K_{par} * X_{minBC} \quad (33.68)$$

$$R_2 = ZFac_1 * R_{maxAB} + K_{par} * R_{minBC} + RF \quad (33.69)$$

$$Z_2 = ZFac_1 * Z_{maxAB} + K_{par} * Z_{minBC} \quad (33.70)$$

$$X_3 = 1.10 * (X_{maxAB} + X_{minBC}) \quad (33.71)$$

$$R_3 = 1.10 * (R_{maxAB} + R_{minBC}) + RF \quad (33.72)$$

$$Z_3 = 1.10 * (Z_{maxAB} + Z_{minBC}) \quad (33.73)$$

### 33.13.4.2 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 33.13.5 and the Zone 2 tab is illustrated in figure 33.13.6 the Zone 3 and Zone 4 tabs are almost identical to the Zone 2 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.

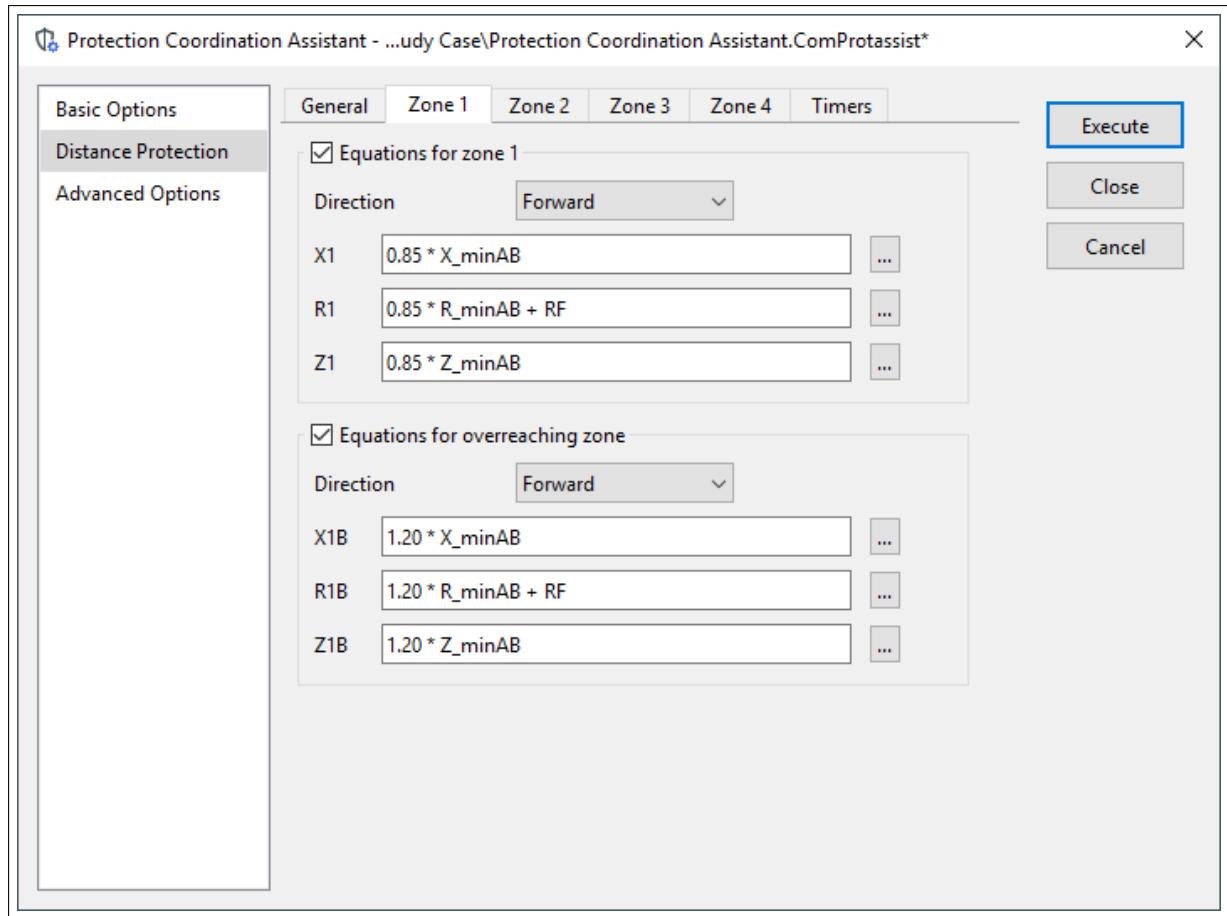


Figure 33.13.5: Zone 1 tab of the distance protection page of the protection coordination assistant command

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 (+, -, /, \*) as well as some special variables. These variables, also known as keywords are described in section [33.13.4.3](#).

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.

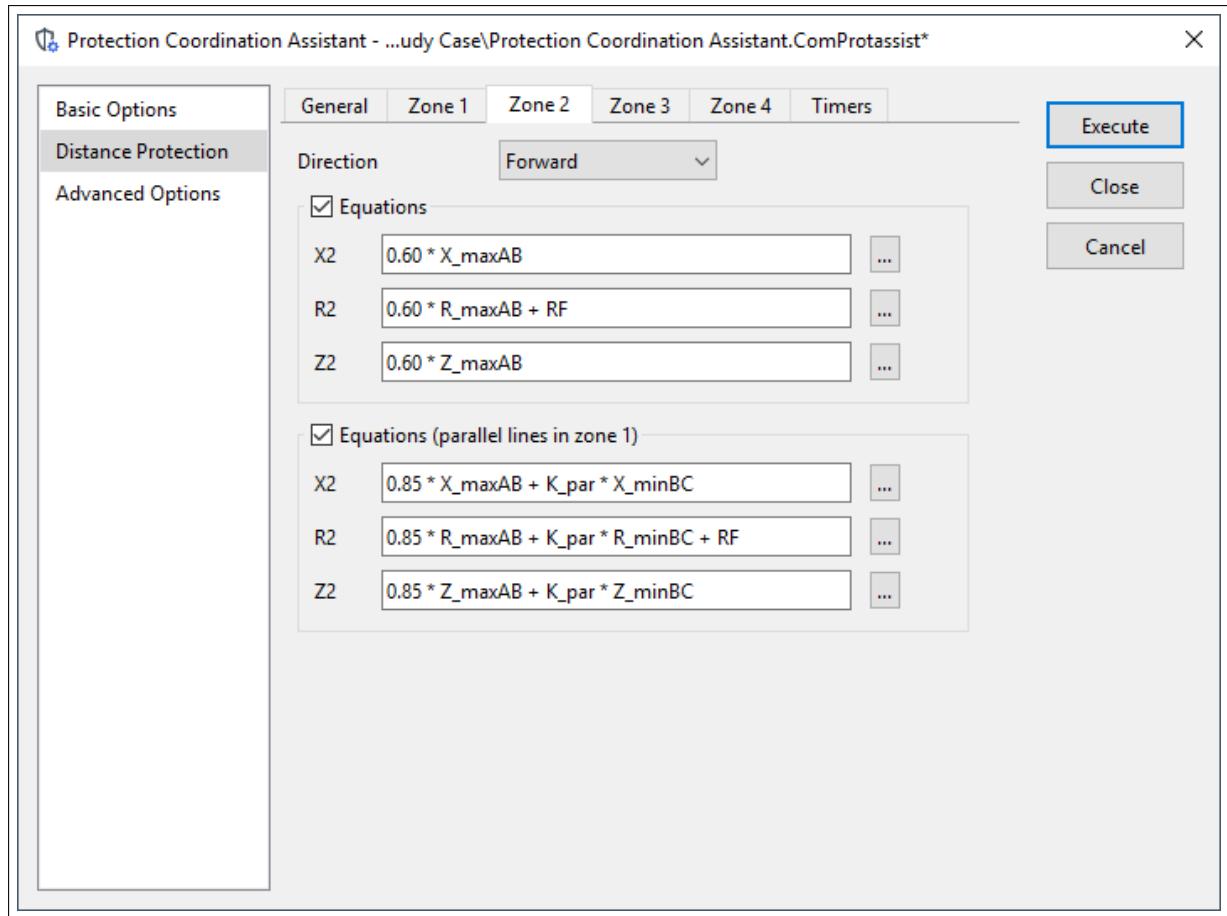


Figure 33.13.6: Zone 2 tab of the distance protection page of the protection coordination assistant command

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.

### 33.13.4.3 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. 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 zone of the relay is facing. 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.

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.

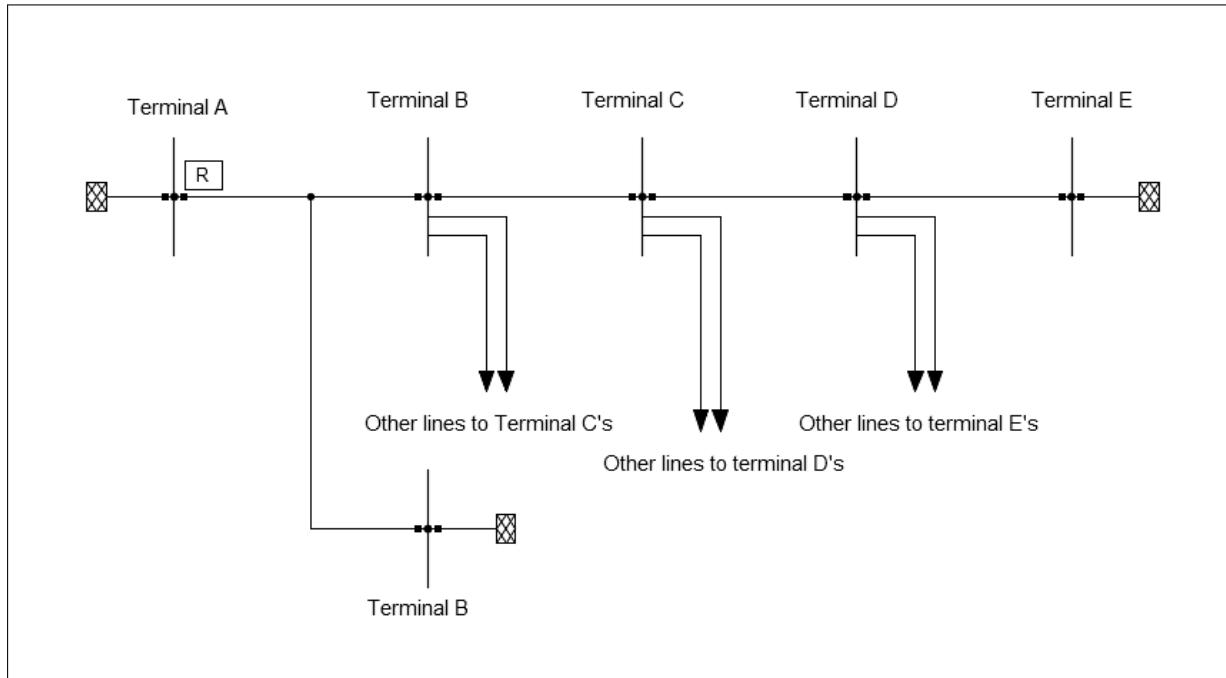


Figure 33.13.7: Classification of busbars in a portion of a network

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 busbar A and any busbar B.

X\_maxAB: The reactance of the maximum impedance line located between busbar A and any busbar B.

R\_minAB: The resistance of the minimum impedance line located between busbar A and any busbar B.

R\_maxAB: The resistance of the maximum impedance line located between busbar A and any busbar B.

Z\_minAB: The impedance of the minimum impedance line located between busbar A and any busbar B.

Z\_maxAB: The impedance of the maximum impedance line located between busbar A and any busbar 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 limiting component could be an item of primary plant such as a line or alternatively a current transformer. 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 busbar B and any adjacent busbar C, where busbar B is the same busbar that was selected for busbar B in the section AB part of the equation.

X\_maxBC: The reactance of the maximum impedance line located between busbar B and any adjacent busbar C, where busbar B is the same busbar that was selected for busbar B in the section AB part of the equation.

R\_minBC: The resistance of the minimum impedance line located between busbar B and any adjacent busbar C, where busbar B is the same busbar that was selected for busbar B in the section AB part of the equation.

R\_maxBC: The resistance of the maximum impedance line located between busbar B and any adjacent busbar C, where busbar B is the same busbar that was selected for busbar B in the section AB part of the equation.

Z\_minBC: The impedance of the minimum impedance line located between busbar B and any adjacent busbar C, where busbar B is the same busbar that was selected for busbar B in the section AB part of the equation.

Z\_maxBC: The impedance of the maximum impedance line located between busbar B and any adjacent busbar C, where busbar B is the same busbar that was selected for busbar 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 busbar C and any adjacent busbar D, where busbar C is the same busbar that was selected for busbar C in the section BC part of the equation.

X\_maxCD: The reactance of the maximum impedance line located between busbar C and any adjacent busbar D, where busbar C is the same busbar that was selected for busbar C in the section BC part of the equation.

R\_minCD: The resistance of the minimum impedance line located between busbar C and any adjacent busbar D, where busbar C is the same busbar that was selected for busbar C in the section BC part of the equation.

R\_maxCD: The resistance of the maximum impedance line located between busbar C and any adjacent busbar D, where busbar C is the same busbar that was selected for busbar C in the section BC part of the equation.

Z\_minCD: The impedance of the minimum impedance line located between busbar C and any adjacent busbar D, where busbar C is the same busbar that was selected for busbar C in the section BC part of the equation..

$Z_{\text{maxCD}}$ : The impedance of the maximum impedance line located between busbar C and any adjacent busbar D, where busbar C is the same busbar that was selected for busbar 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 busbar D and any adjacent busbar E, where busbar D is the same busbar that was selected for busbar D in the section CD part of the equation.

$X_{\text{maxDE}}$ : The reactance of the maximum impedance line located between busbar D and any adjacent busbar E, where busbar D is the same busbar that was selected for busbar D in the section CD part of the equation.

$R_{\text{minDE}}$ : The resistance of the minimum impedance line located between busbar D and any adjacent busbar E, where busbar D is the same busbar that was selected for busbar D in the section CD part of the equation.

$R_{\text{maxDE}}$ : The resistance of the maximum impedance line located between busbar D and any adjacent busbar E, where busbar D is the same busbar that was selected for busbar D in the section CD part of the equation.

$Z_{\text{minDE}}$ : The impedance of the minimum impedance line located between busbar D and any adjacent busbar E, where busbar D is the same busbar that was selected for busbar D in the section CD part of the equation.

$Z_{\text{maxDE}}$ : The impedance of the maximum impedance line located between busbar D and any adjacent busbar E, where busbar D is the same busbar that was selected for busbar 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.

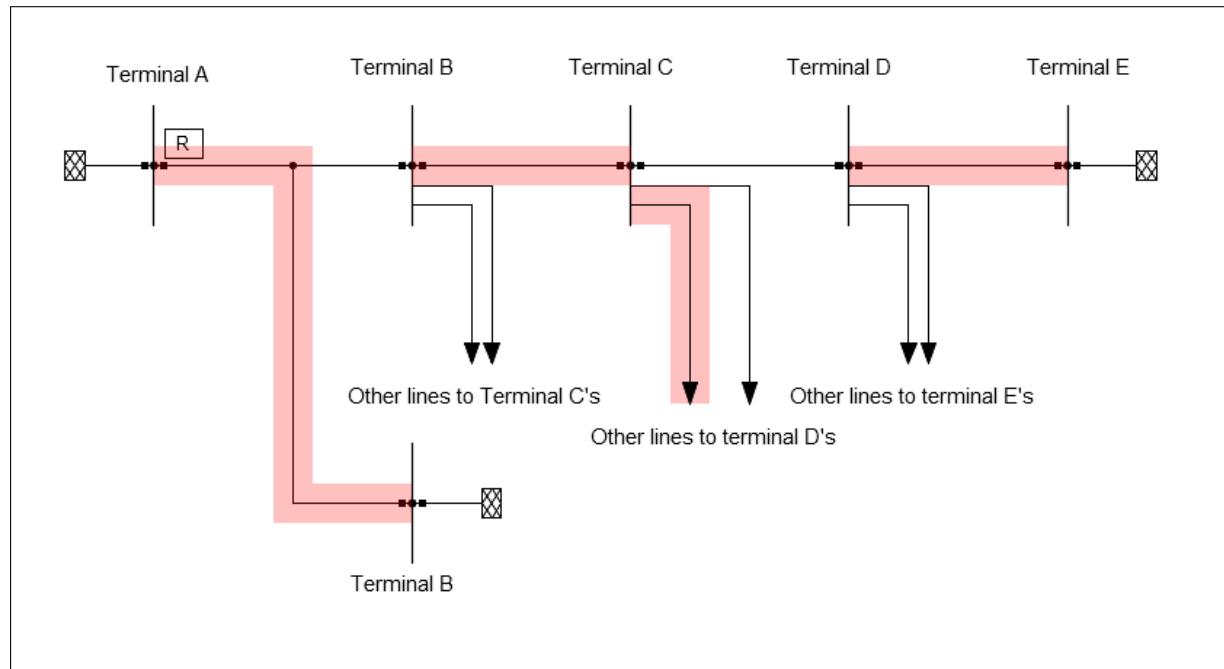


Figure 33.13.8: Example of an impossible combination of lines for a reach equation

**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  $Z_{\text{min}}BA$ .

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.

#### 33.13.4.4 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

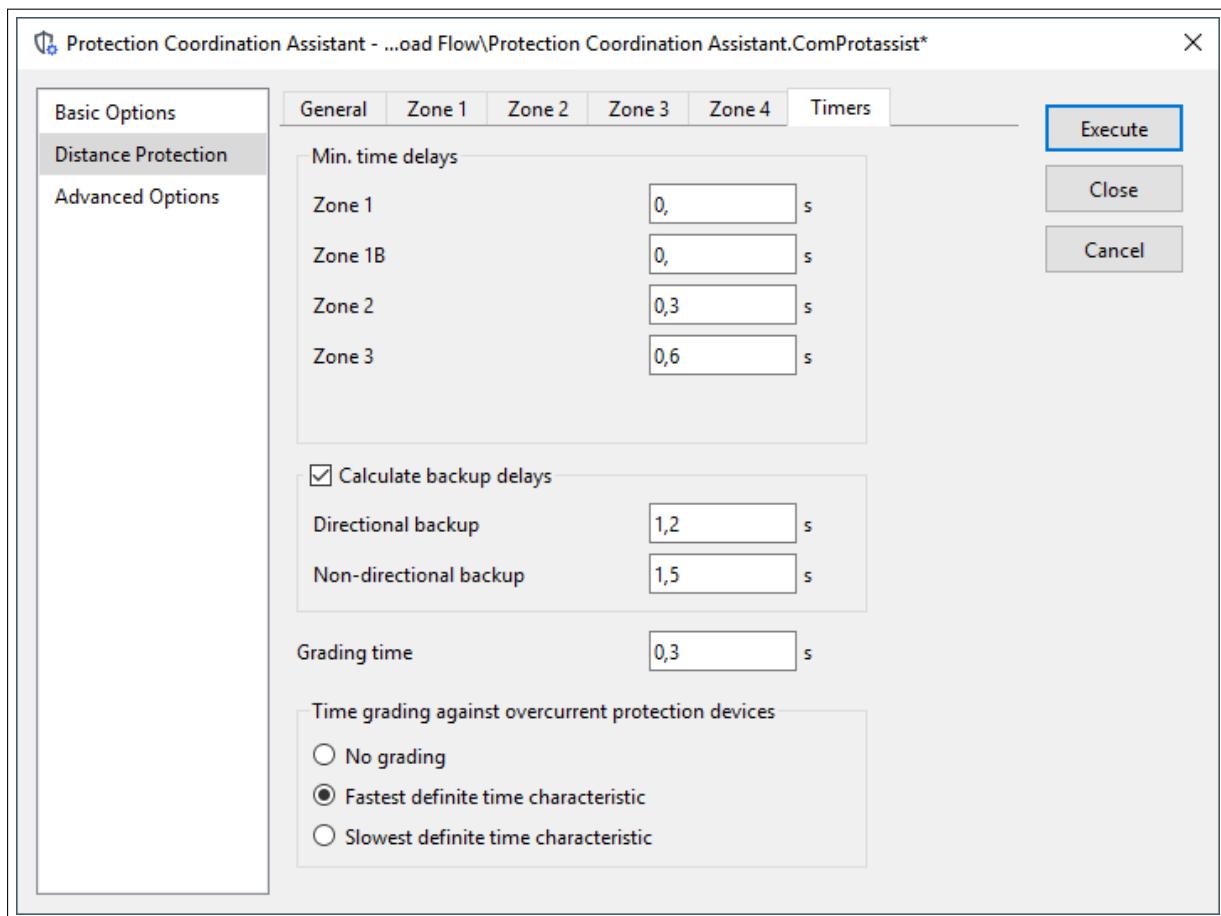


Figure 33.13.9: Timers tab of the distance protection page of the protection coordination assistant command

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.

The *grading time* parameter can be used to apply an additional constraint to the setting calculation for the case where a distance zone reaches over a position containing an overcurrent relay. In such a case, the backup zone will be coordinated with the overcurrent relay so as to achieve the specified grading time margin. The *Time grading against overcurrent protection devices* field is used to specify over which part of the overcurrent relay's time-overcurrent characteristic the margin will be sought. For characteristics with two definite time stages, either the fastest or the slowest definite time stage can be selected. This could be useful for example where a distance protected line contains a tee off to a transformer which is protected by an overcurrent relay with two definite time stages. The fast definite time stage will most likely be set to reach into the transformer, whilst the slower stage will be set to reach past the transformer so as to coordinate with and provide backup protection to, an overcurrent protection device on the lower voltage side. Depending on the relative penetration of the distance protection and the fast definite time characteristic of the overcurrent protection into the transformer, there may be benefit in selecting a grading margin against the fast definite time characteristic as opposed to the slow one, or even vice-versa.

### 33.13.5 Protection Coordination Assistant Command - Advanced Options

This page is illustrated in figure 33.13.10.

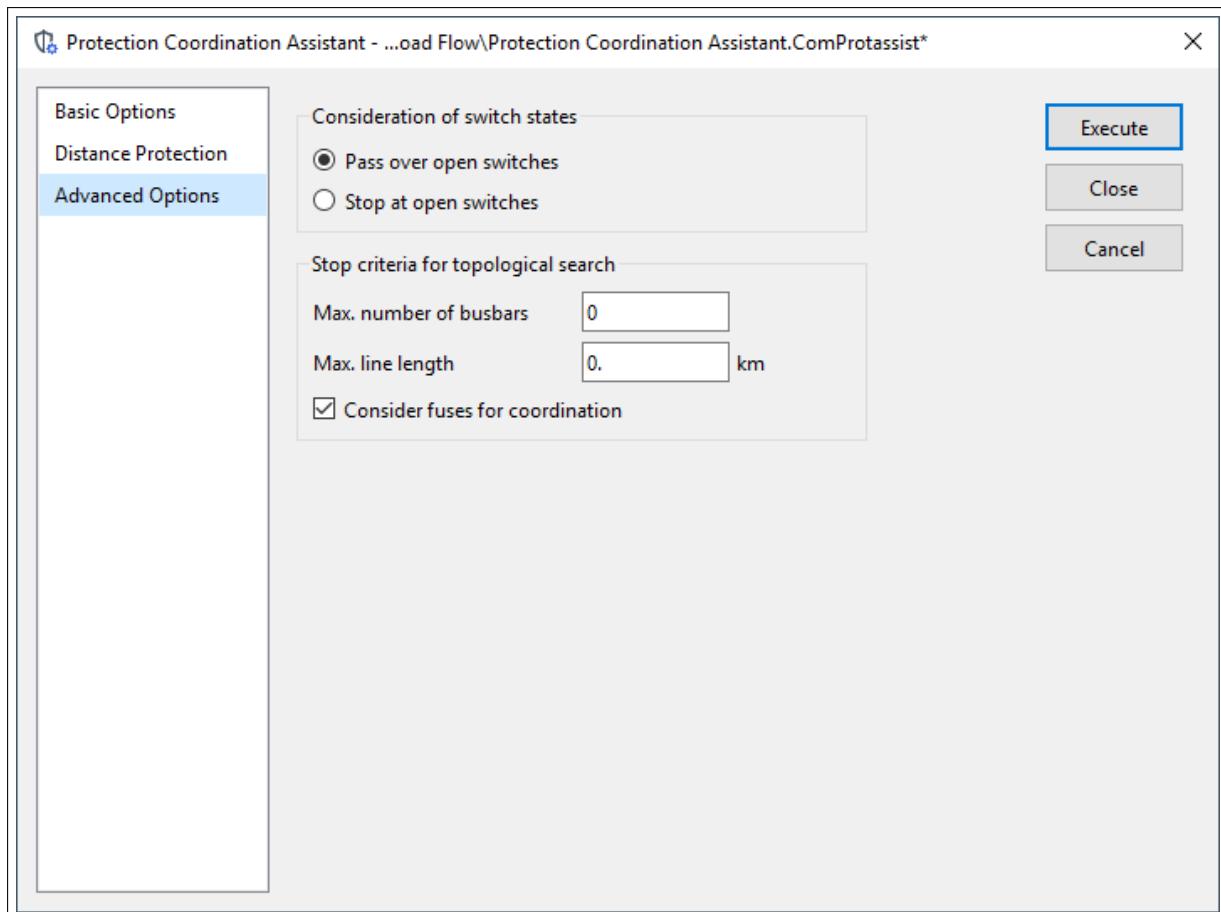


Figure 33.13.10: Protection coordination assistant command - Advanced Options

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 *Consideration of switch states* field allows the user to specify whether the search routine stops at open switches or continues past them so that impedances beyond the open switches may be used in the reach calculations.

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.

### 33.13.6 Prerequisites for using the distance protection coordination tool

Before starting the distance 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.

### 33.13.7 How to run the distance protection coordination calculation

To run the distance 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 *Distance Protection* and *Advanced Options* pages.
6. Click **OK** to run the coordination.
7. To analyse results of the coordination see Section [33.13.8](#).

### 33.13.8 How to output results from the protection coordination assistant

This section explains how the results from the distance protection coordination assistant can be analysed. The graphical method of analysis using the time-distance diagram 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 relays located within the analysed path 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:

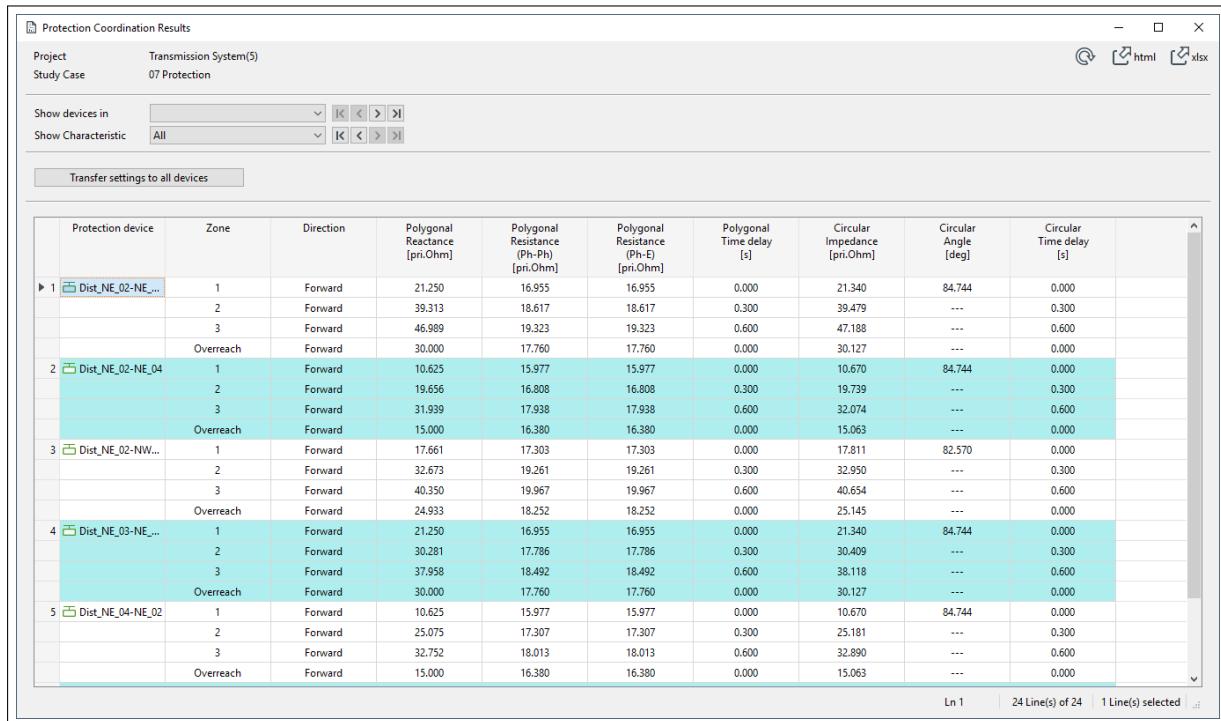
To output results from the protection coordination assistant follow these steps:

1. Execute the protection coordination tool. See Section [33.13.7](#) for instructions how to do this.

2. Click  the icon from the protection toolbar. A dialog for choosing the output options will appear.
3. Check the boxes for the reports that you would like *PowerFactory* to produce. The types of reports are:
  - **Create Report.** This option produces a tabular report of the coordination results. See Section [33.13.8.1](#) for further information on this report.
  - **Create Time-Distance-diagram.** This option presents a plot showing graphically the results of the protection coordination. More information about this diagram is presented in Section [33.8](#).
4. The other options in this dialog are:
  - **Results File.** Here the results 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.
  - **Path Selection.** This option is available only when the *Create Time-Distance Plot* check box is selected. Time-Distance plots are always linked to a path object. So in order to create time distance plots, paths must first be specified in the network model. See Section [15.9](#) for more information about paths. If paths have already been created these can be selected via this option here.

### 33.13.8.1 Tabular report of the protection coordination results

Enabling the option *Create Report* when outputting the coordination results as described in Section [33.13.8](#), automatically generates a table report showing the results from the previously executed protection coordination. An example of the coordination result report is illustrated in figure [33.13.11](#).



The screenshot shows a software window titled "Protection Coordination Results". The window has tabs for "Project" (Transmission System(5)) and "Study Case" (07 Protection). It includes filters for "Show devices in" and "Show Characteristic". A "Transfer settings to all devices" button is present. The main area is a table with the following columns: Protection device, Zone, Direction, Polygonal Reactance [pri.Ohm], Polygonal Resistance (Ph-Ph) [pri.Ohm], Polygonal Resistance (Ph-E) [pri.Ohm], Polygonal Time delay [s], Circular Impedance [pri.Ohm], Circular Angle [deg], Circular Time delay [s]. The table contains data for five protection devices (1 through 5), each with multiple zones (1, 2, 3, Overreach) and their respective settings. The data is color-coded by zone.

Protection device	Zone	Direction	Polygonal Reactance [pri.Ohm]	Polygonal Resistance (Ph-Ph) [pri.Ohm]	Polygonal Resistance (Ph-E) [pri.Ohm]	Polygonal Time delay [s]	Circular Impedance [pri.Ohm]	Circular Angle [deg]	Circular Time delay [s]
1  Dist_NE_02-NE_...	1	Forward	21.250	16.955	16.955	0.000	21.340	84.744	0.000
	2	Forward	39.313	18.617	18.617	0.300	39.479	---	0.300
	3	Forward	46.998	19.323	19.323	0.600	47.188	---	0.600
	Overreach	Forward	30.000	17.760	17.760	0.000	30.127	---	0.000
2  Dist_NE_02-NE_04	1	Forward	10.625	15.977	15.977	0.000	10.670	84.744	0.000
	2	Forward	19.656	16.808	16.808	0.300	19.739	---	0.300
	3	Forward	31.939	17.938	17.938	0.600	32.074	---	0.600
	Overreach	Forward	15.000	16.380	16.380	0.000	15.063	---	0.000
3  Dist_NE_02-NW...	1	Forward	17.661	17.303	17.303	0.000	17.811	82.570	0.000
	2	Forward	32.673	19.261	19.261	0.300	32.950	---	0.300
	3	Forward	40.350	19.967	19.967	0.600	40.654	---	0.600
	Overreach	Forward	24.933	18.252	18.252	0.000	25.145	---	0.000
4  Dist_NE_03-NE_...	1	Forward	21.250	16.955	16.955	0.000	21.340	84.744	0.000
	2	Forward	30.281	17.786	17.786	0.300	30.409	---	0.300
	3	Forward	37.958	18.492	18.492	0.600	38.118	---	0.600
	Overreach	Forward	30.000	17.760	17.760	0.000	30.127	---	0.000
5  Dist_NE_04-NE_02	1	Forward	10.625	15.977	15.977	0.000	10.670	84.744	0.000
	2	Forward	25.075	17.307	17.307	0.300	25.181	---	0.300
	3	Forward	32.752	18.013	18.013	0.600	32.890	---	0.600
	Overreach	Forward	15.000	16.380	16.380	0.000	15.063	---	0.000

Figure 33.13.11: Protection coordination results report

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.

---

### 33.13.8.2 Time distance diagrams from the protection coordination

Enabling the option *Create Report* when outputting the coordination results as described in Section 33.13.8, 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 33.13.12.

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.

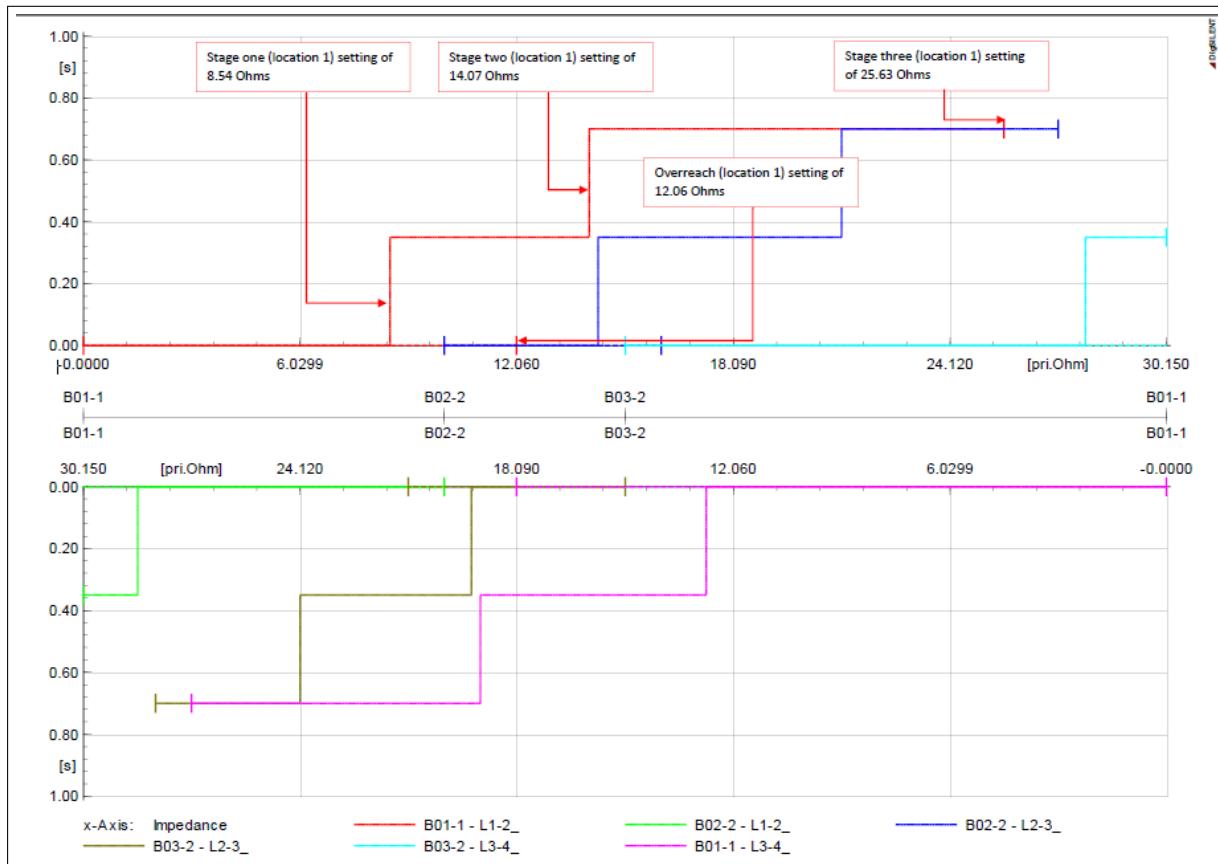


Figure 33.13.12: Time distance diagram showing the result from the protection coordination using the independent method on the network shown in Figure 33.13.1

## 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 19.7.15: 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 *Show → In existing Protection Plots* from the context menu. This option can be called from the Object Filter (see Figure 33.14.1) or directly out of the network graphic (see Figure 33.14.2).

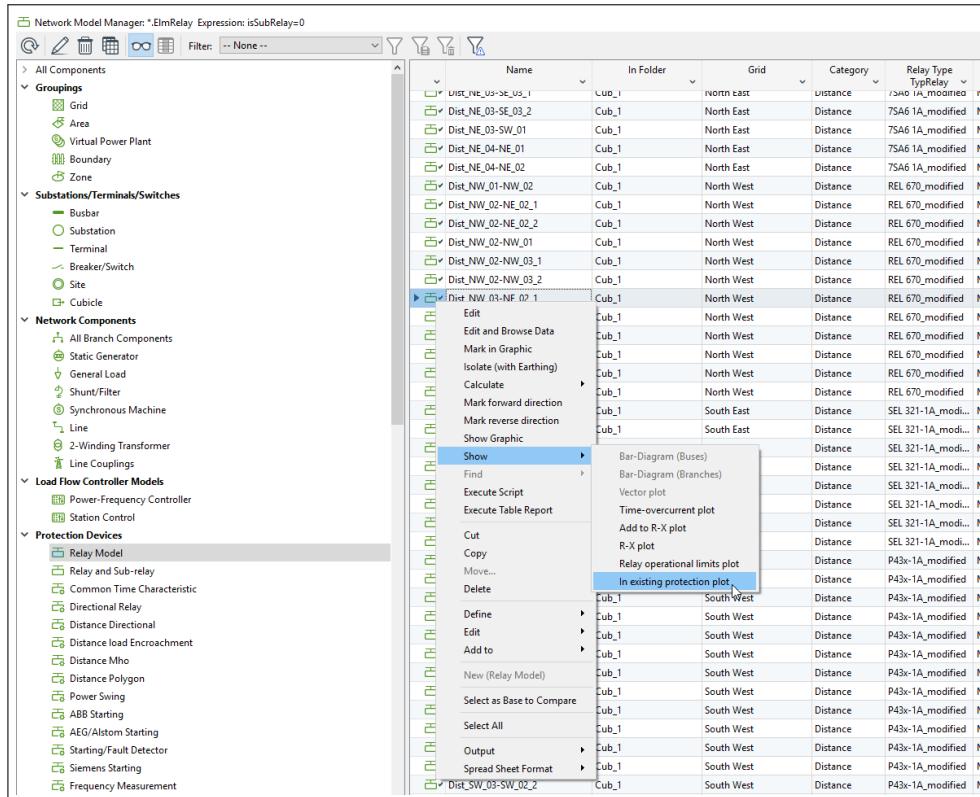


Figure 33.14.1: Access through Object Filter

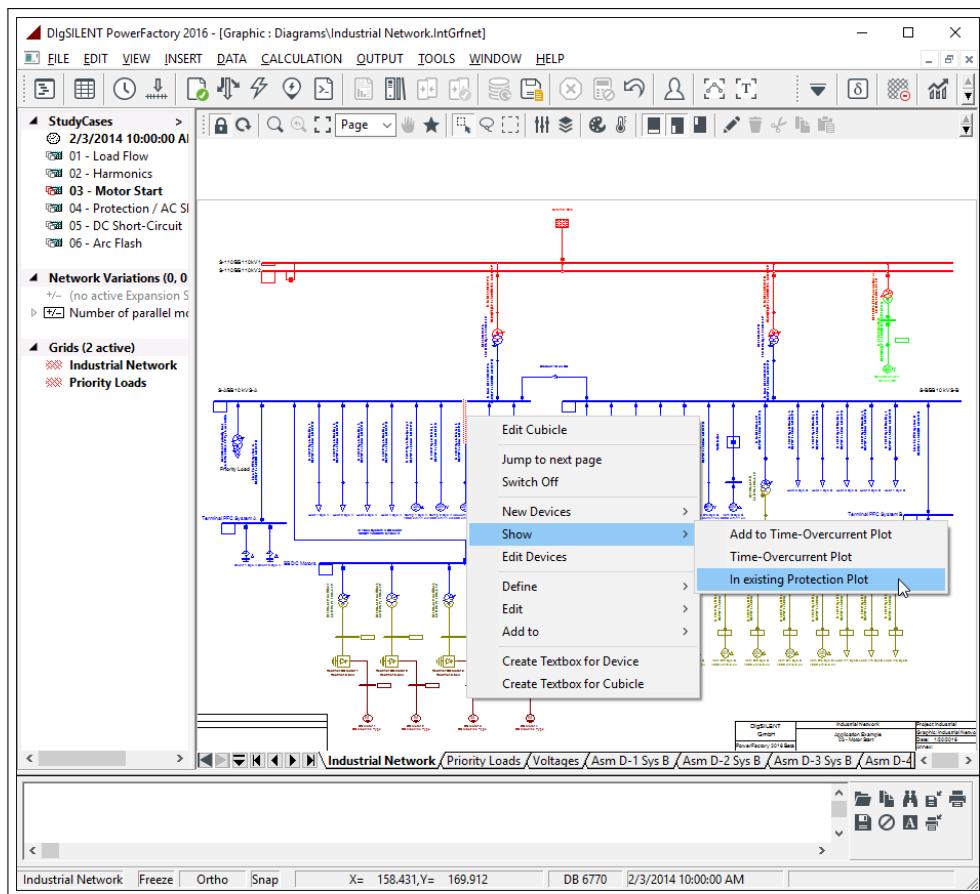


Figure 33.14.2: Access from the Grid

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).

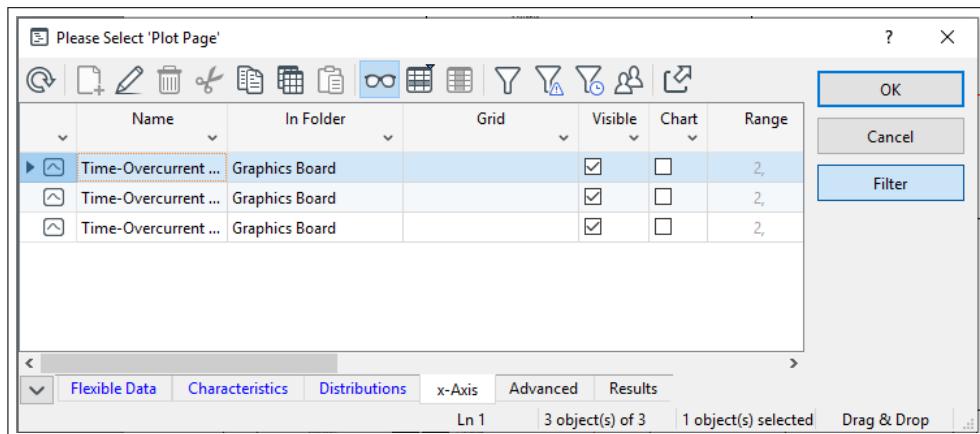


Figure 33.14.3: selection of the plot

### 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 top bar includes project information ("Project: Industrial Example(1)", "Study Case: 02\_Protection") and export options ("html", "xlsx"). The main area is a table with the following columns: Protection Devi..., Location, Branch, Manufacturer, Model, Stage (Phase), Current [pri.A], Current [sec.A], Current [p.u.], Time, Characteristic, and Directional. The table contains 7 rows of data, each representing a relay stage. Row 1 (R\_4-7) is highlighted with a blue border. The data is as follows:

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

At the bottom of the table, there are navigation buttons (left, right, first, last) and status text: "Ln 1 | 159 Line(s) of 159 | 1 Line(s) selected | .:.

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 (*ElmAra*) 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.

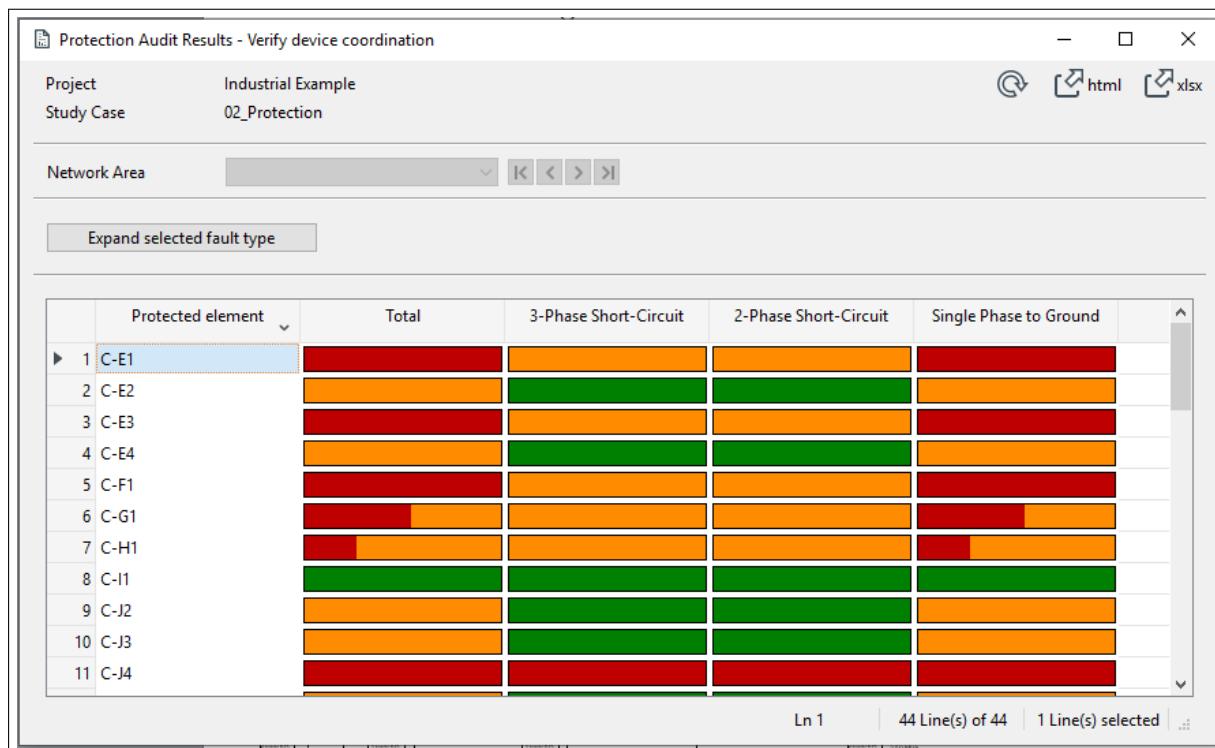


Figure 33.15.1: Network element oriented results example



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.

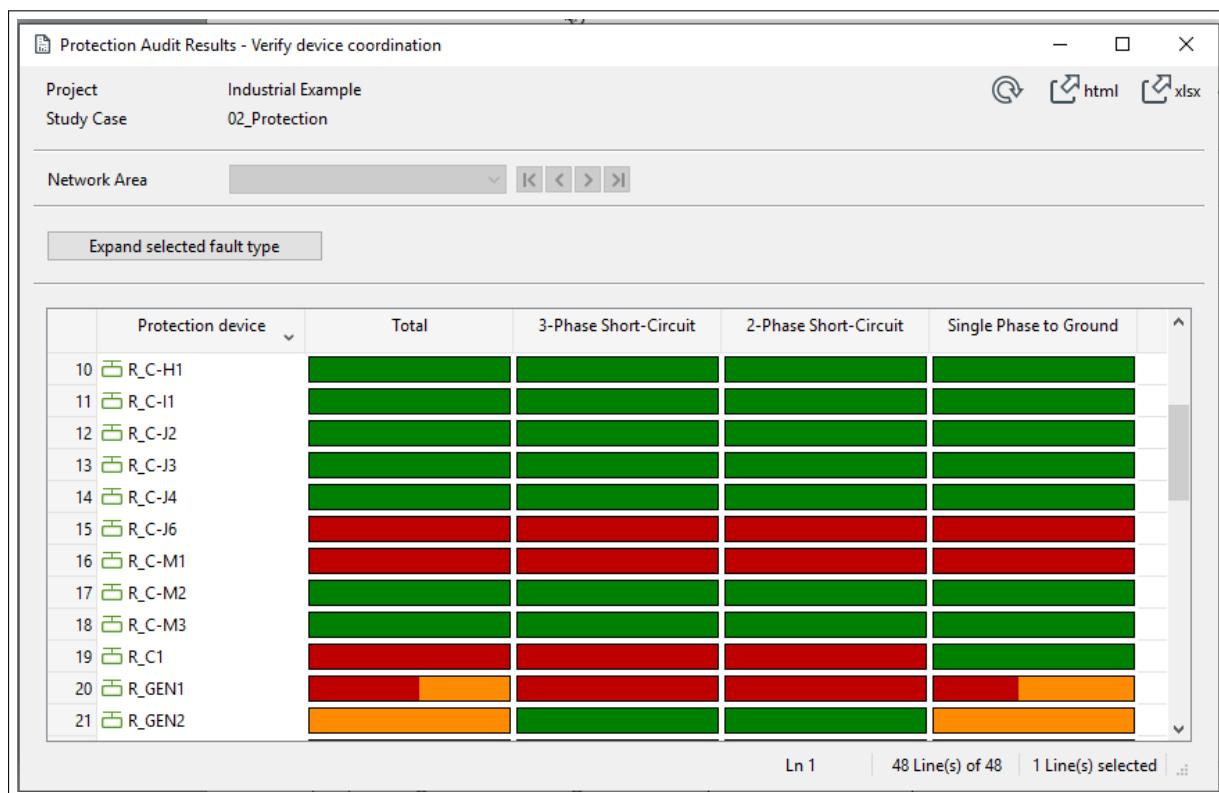


Figure 33.15.3: Protection device oriented results example

Protection device	Node	Branch	Total	1PHS_A_0,000+j,000	1PHS_A_10,000+j,000	1PHS_A_5,000+j,000	1PHS_B_0,000+j,000	1PHS_C_0,000+j,000
1 F Stage 1	T F Stage 1	F Stage 1						
2 F Stage 2	T F Stage 2	F Stage 2						
3 F Stage 3	T F Stage 3	F Stage 3						
4 F Stage 4	T F Stage 4	F Stage 4						
5 R 10 Sys A	S-A/BB 10 kV S-A	In 10kV SysA Mising 2						
6 R 10 Sys B	S-B/BB 10 kV S-B	Softstarter						
7 R 11 Sys A	S-A/BB 10 kV S-A	In 10kV SysA Mising 3						
8 R 11 Sys A	S-A/BB 10 kV S-A	In 10kV SysA Mising 4						
9 R 11 Sys B	S-B/BB 10 kV S-B	In 10kV SysB Heating 5						
10 R 12 Sys B	S-B/BB 10 kV S-B	In 10kV SysB Heating 4						
11 R 13 Sys B	S-B/BB 10 kV S-B	In 10kV SysB Heating 3						
12 R 14 Sys B	S-B/BB 10 kV S-B	In 10kV SysB Heating 2						

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:

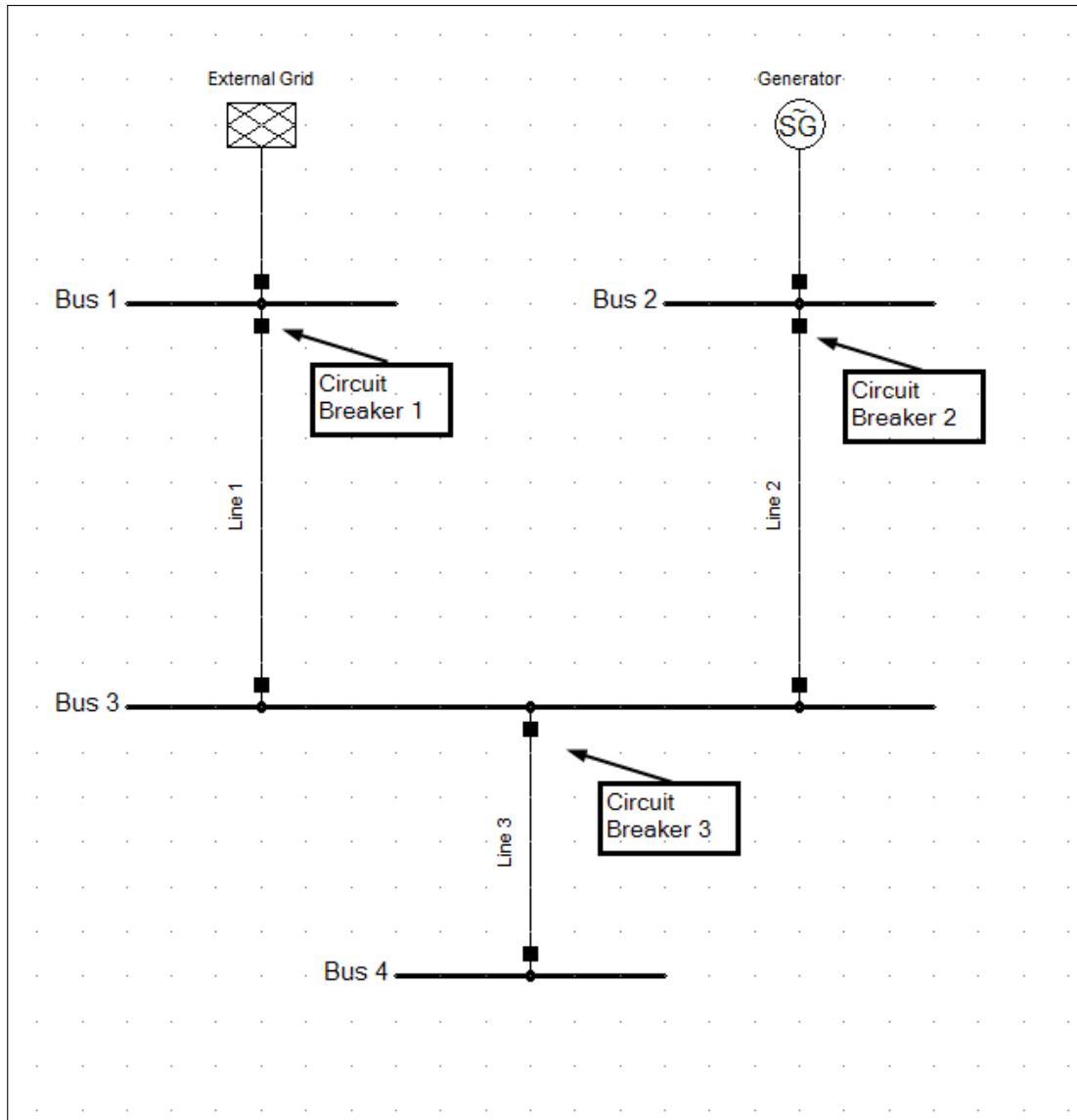


Figure 33.16.1: Short circuit trace example

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.

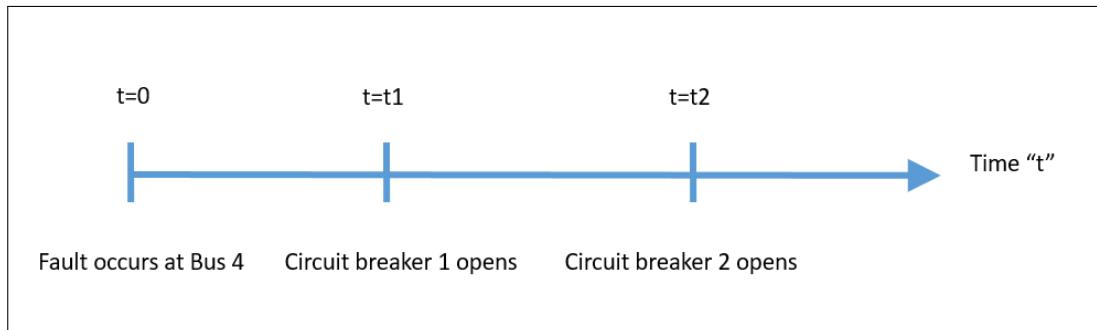


Figure 33.16.2: Short circuit trace sequence

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 ( $t = 0 + t_1$ ): 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  $t_1$ . 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 ( $t = 0 + t_2$ ): 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  $t_2$ . 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 sensitive menu by right clicking on busbar or line in the data manager or single line diagram and by choosing *Calculate → Short Circuit Trace....* When executing the command from the context sensitive 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.

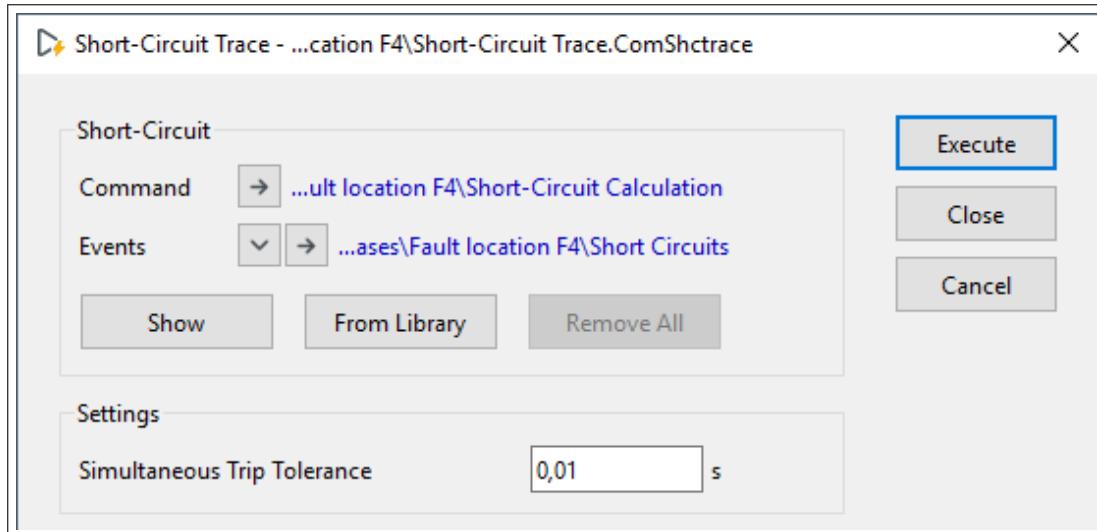


Figure 33.16.3: Short circuit trace command dialog

### 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 (*EvtShcII*)
- 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

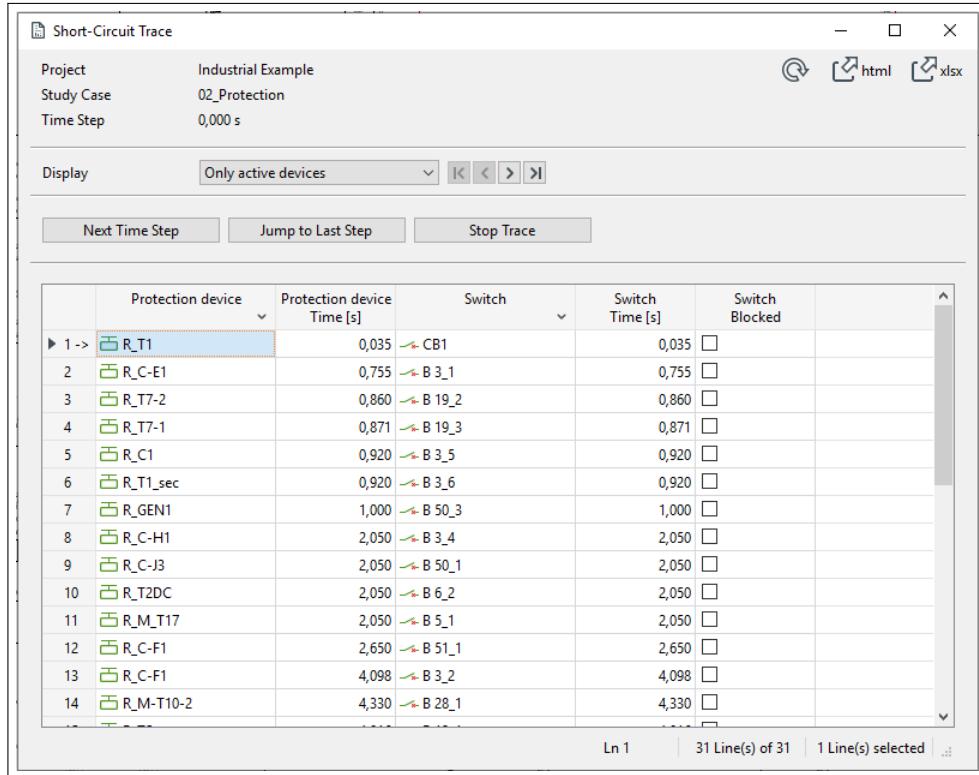
device combination. However, as the trace is progressed this value will be updated. The user is given the option to display *Only active devices* (i.e. devices which are instigated to trip in response to the fault) or *All devices*.

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 sensitive 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.



The screenshot shows the 'Short-Circuit Trace' window with the following details:

- Project:** Industrial Example
- Study Case:** 02\_Protection
- Time Step:** 0,000 s
- Display:** Only active devices
- Buttons:** Next Time Step, Jump to Last Step, Stop Trace

	Protection device	Protection device Time [s]	Switch	Switch Time [s]	Switch 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"/>

At the bottom of the table, status indicators show: 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.

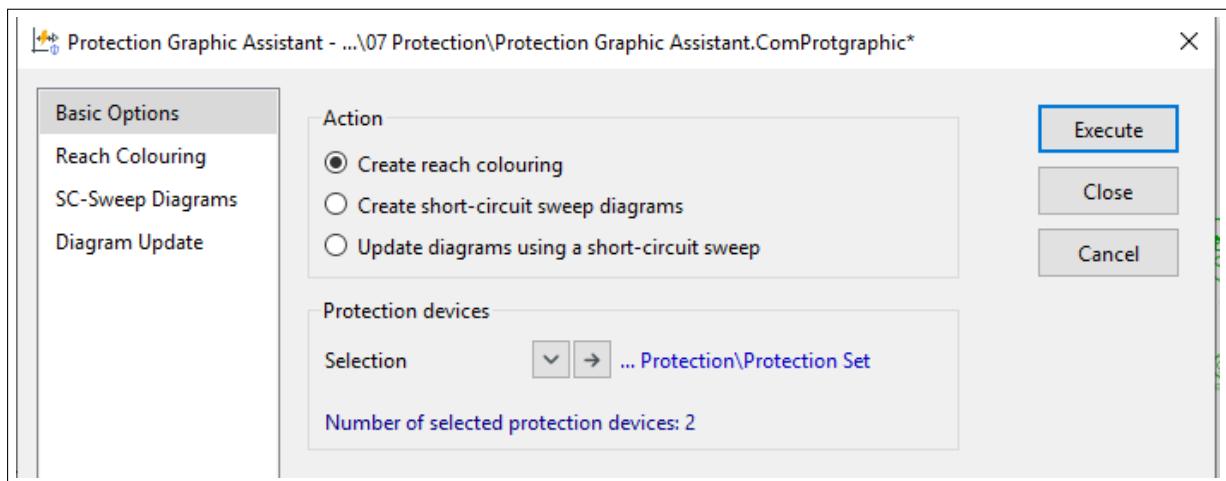


Figure 33.17.1: Protection Graphic Assistant - Basic Options

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.

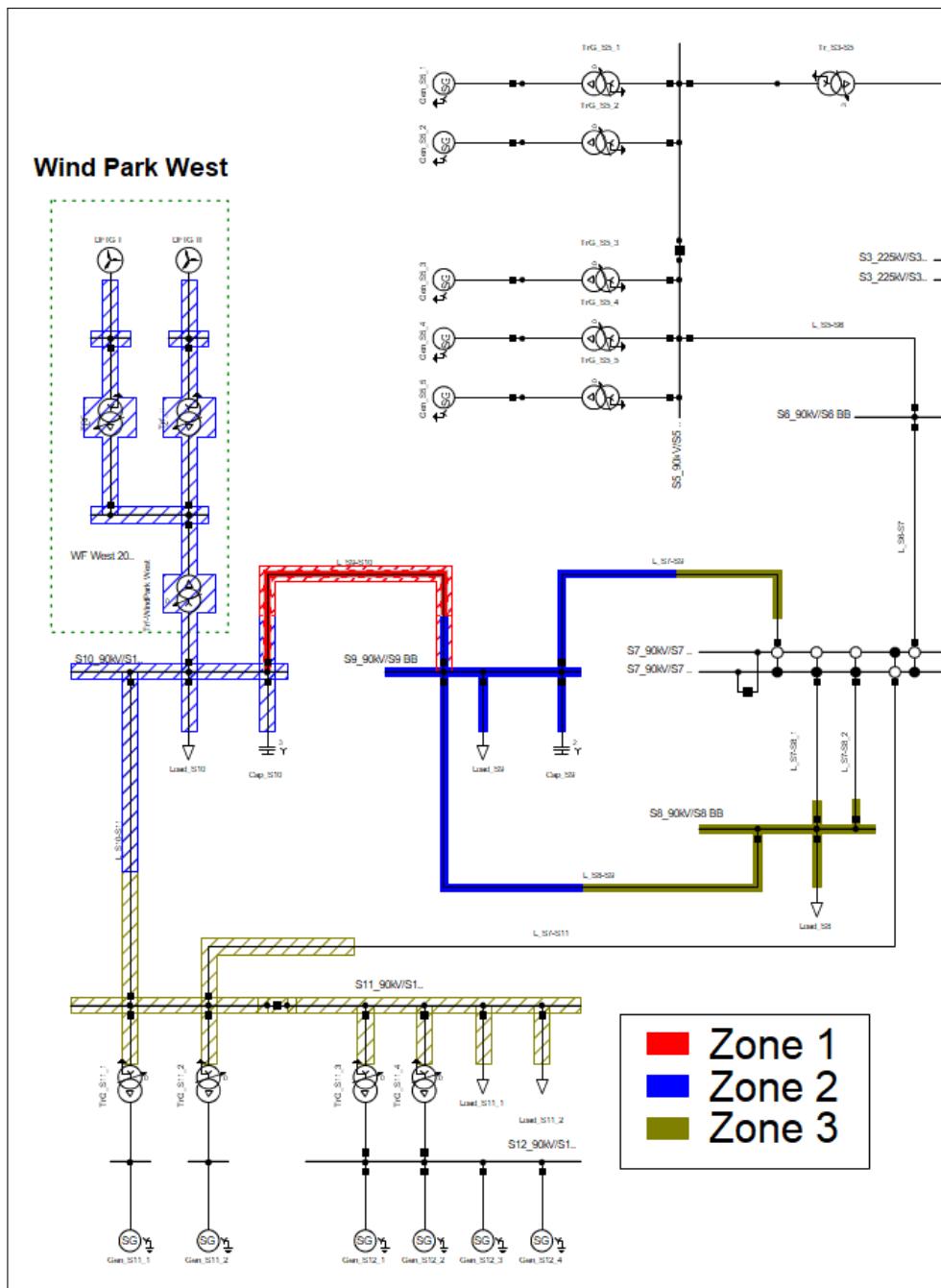


Figure 33.17.2: Distance protection reach colouring in the single line diagram

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

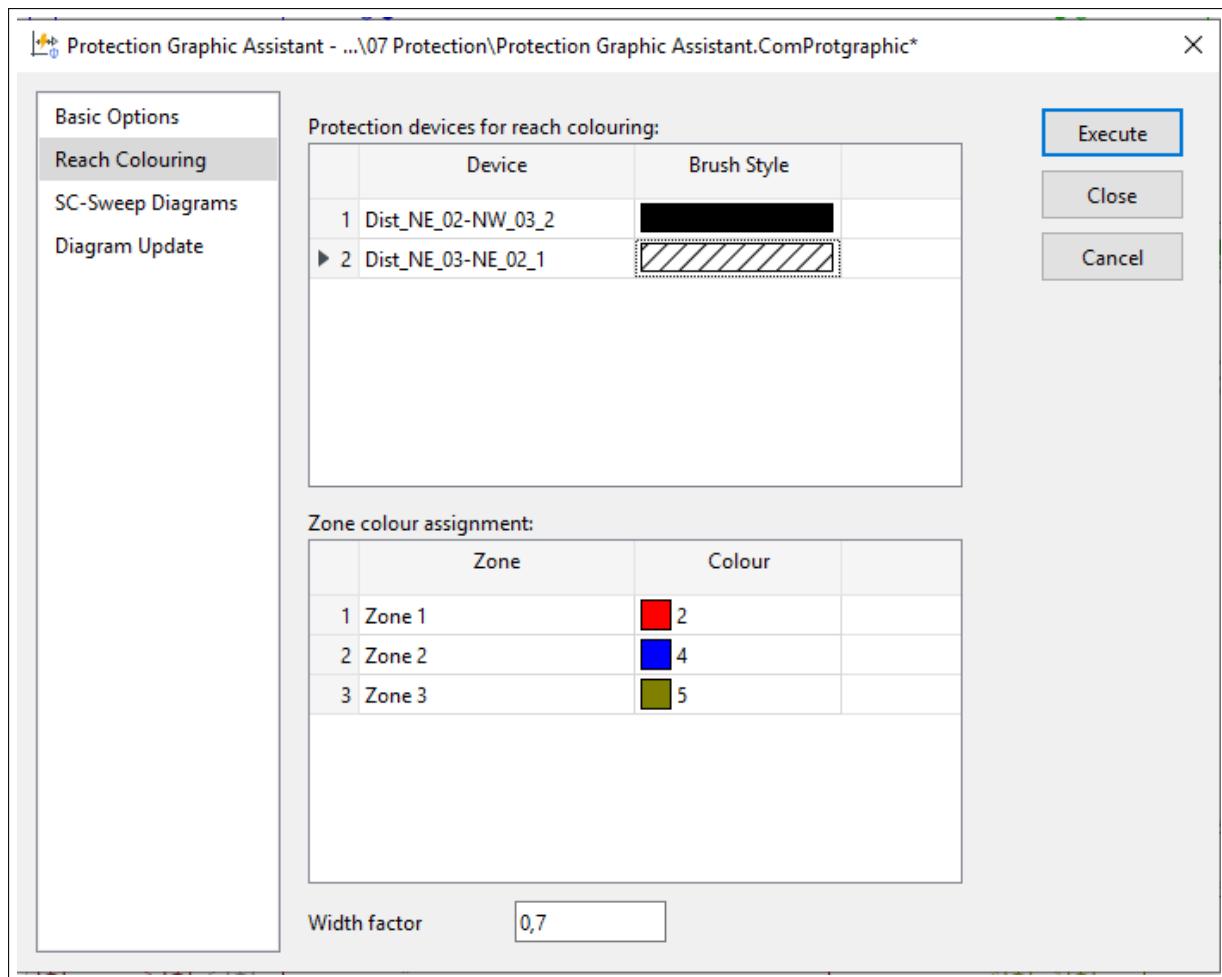


Figure 33.17.3: Protection Graphic Assistant - Reach Colouring

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).

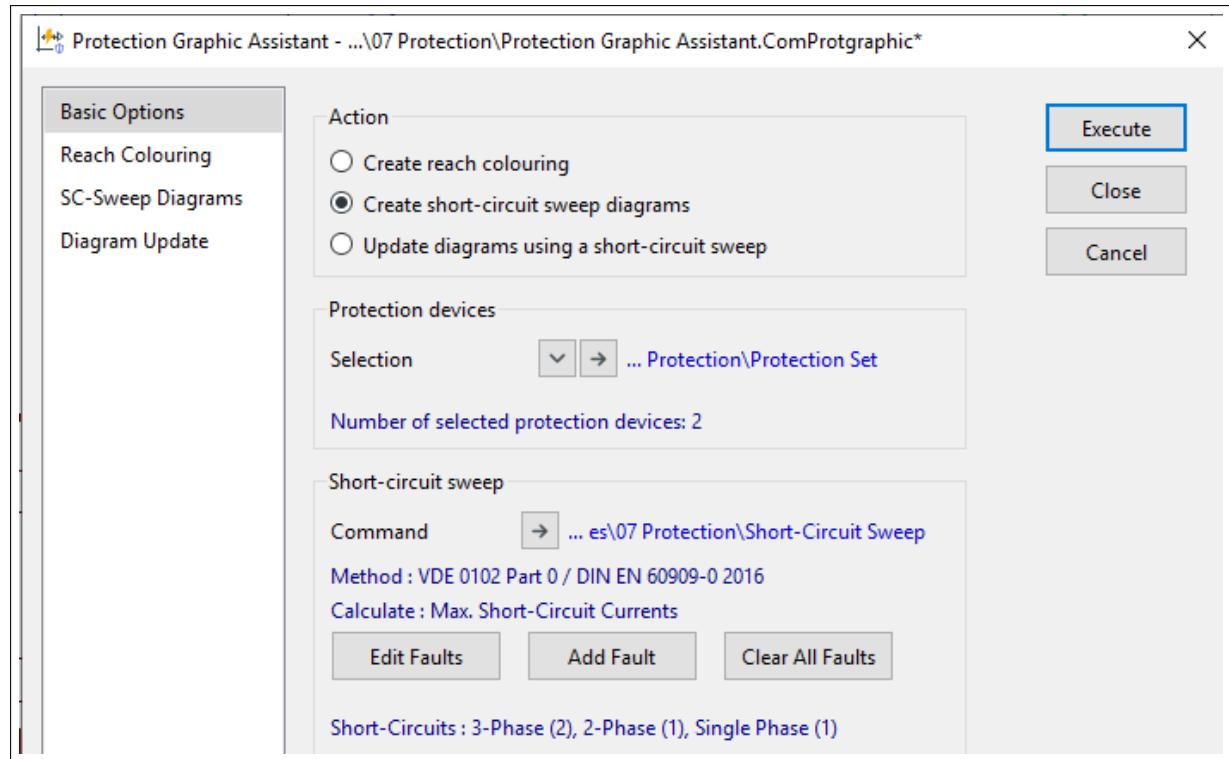


Figure 33.17.4: Protection Graphic Assistant - Basic Options for Short-Circuit Sweep Diagrams

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.

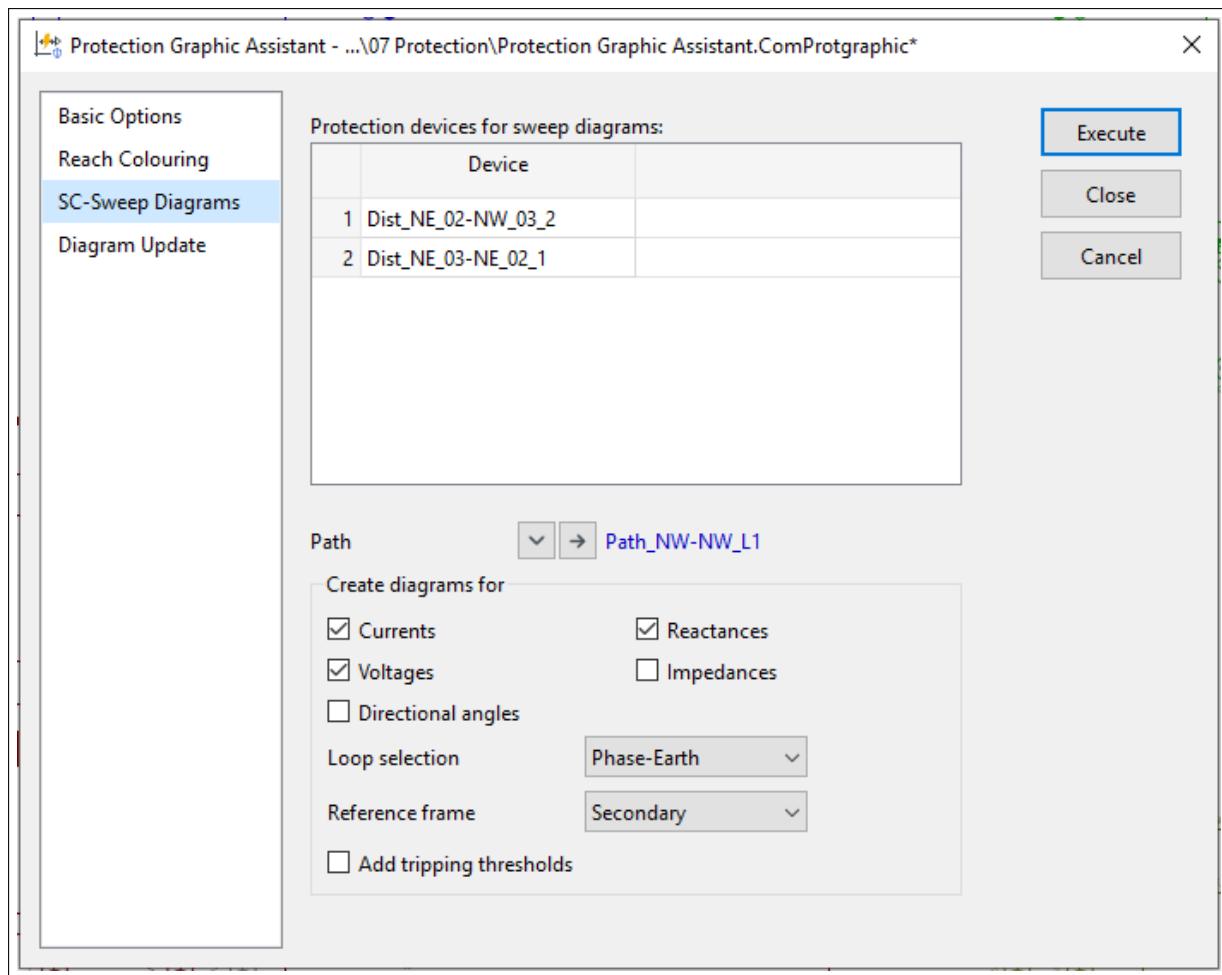


Figure 33.17.5: Protection Graphic Assistant - Short-Circuit Sweep Diagrams

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 *file* → *New* → *Block Diagram / 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.

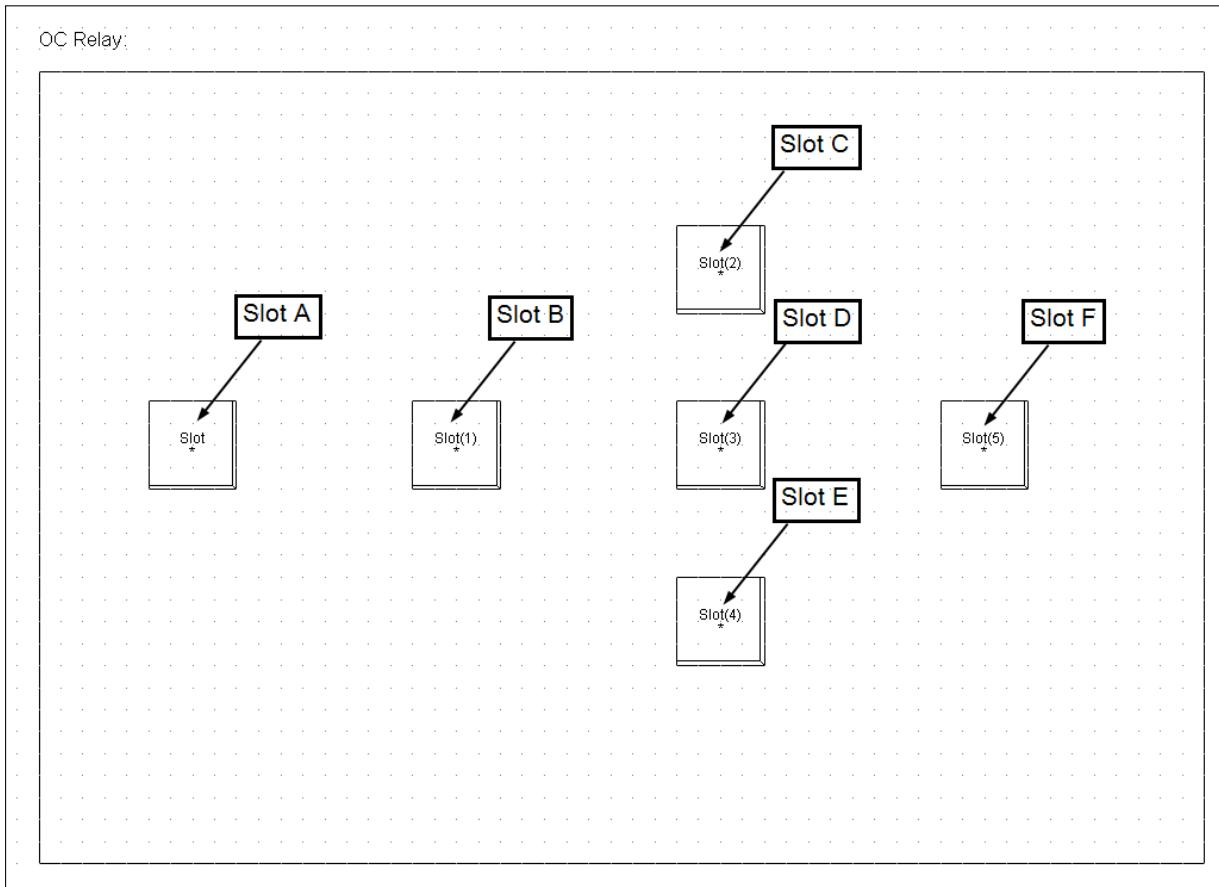


Figure 33.18.1: Arrangement of slots

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 eg. 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 eg. 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 eg. 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 eg. 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.

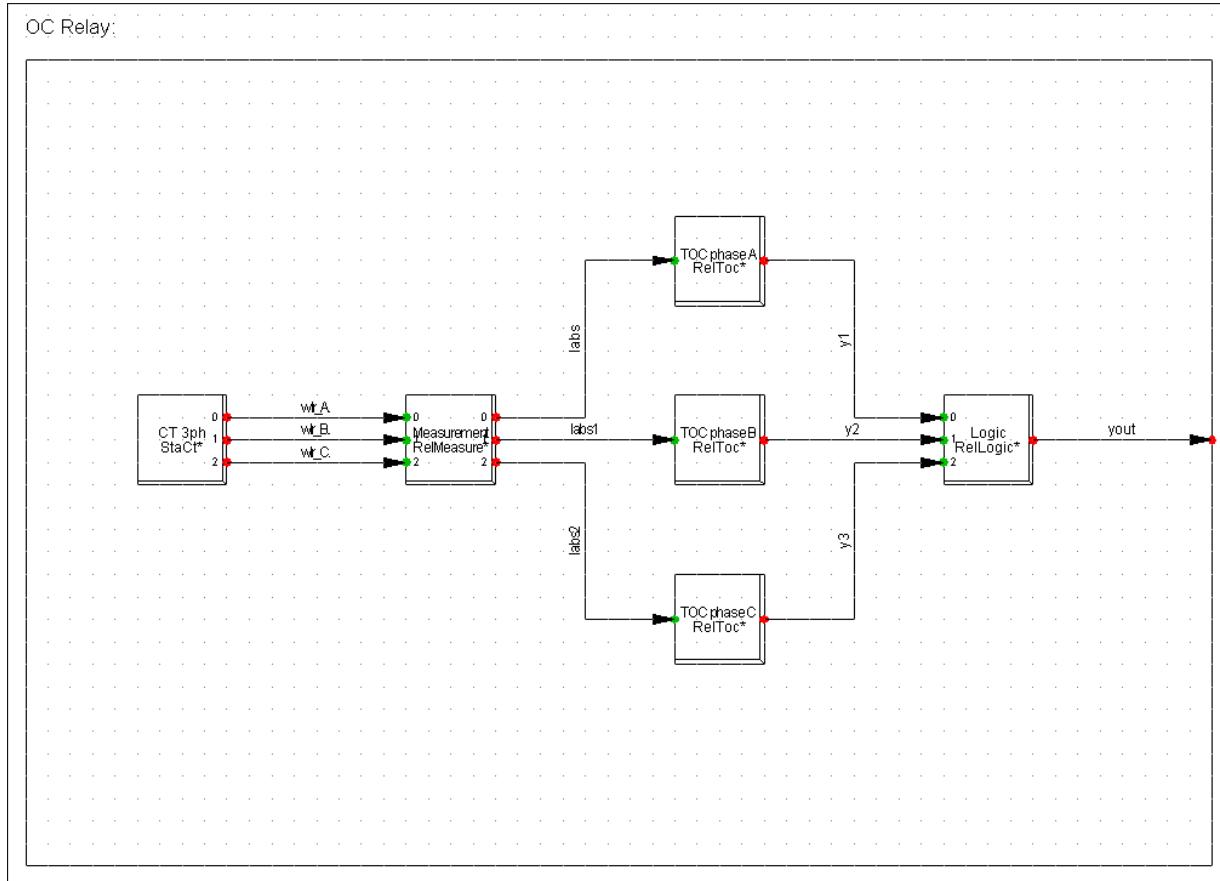


Figure 33.18.2: Signal route definition

#### 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 *Special Type → Relay Type (TypRelay)*.
- 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 section of the [Technical References Overview Document](#) for details on protection blocks and the measurement devices section for 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*, set the pointer to *Project Settings*, and on the *Input* page, and select the units to be “English-Transmission” or “English-Industry”.

---

### 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][24]. 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**[24]: Conductor Gap, Distance Factor, Working Distance, and Enclosure Type.
- **IEEE-1584 - 2018**[25]: Conductor Gap, Working Distance, and Electrode configuration. For some Electrode configurations additionally Dimensions of enclosure might be needed
- **NFPA 70E**[28]: Working Distance and Enclosure Type.
- **DGUV 203-077**[22]: Conductor Gap, Distance Factor and Working Distance.

When an Arc-Flash Hazard Analysis is conducted using the IEEE-1584 method, *PowerFactory* calculates the arcing current based on the equations presented in the standard. *PowerFactory* calculates the arc resistance required to limit the fault current to the calculated value. When the NFPA method is selected, the bolted fault current is used for the calculation. For either method, when the user chooses to use relay tripping times, a second calculation is performed at a reduced fault current (as specified by the user) and the associated (generally longer) clearing time. *PowerFactory* compares the results of these two cases and reports on the worst-case result.

### 34.2.2 Data Inputs

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][24]

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 [25]

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.

If *Accessible Location* parameter is selected on the *General* tab, the user may enter the required input parameters for arc-flash calculations on corresponding tab.

If the terminal resides within a substation, *Equipment Data* can be set to either *Local Values* or *From Substation*. When *From Substation* is selected, a pointer to the relevant substation is shown in the dialog.

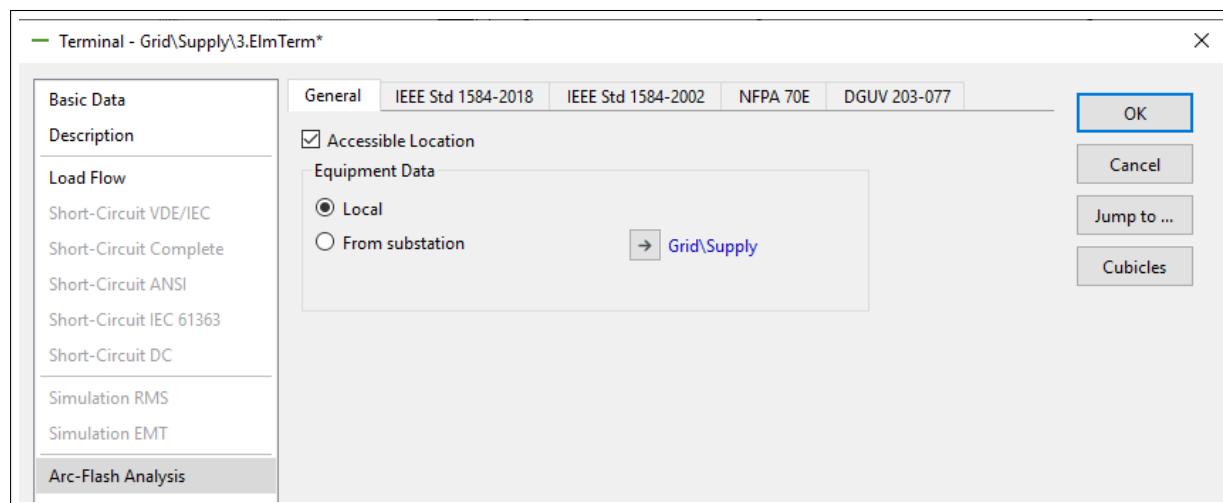


Figure 34.2.1: Arc-flash data required for terminals

Additional data required for *Fault Clearing Times* is discussed later in this chapter.

## 34.3 Arc-Flash Hazard Analysis Calculation Options

### 34.3.1 Arc-Flash Hazard Analysis Basic Options Page

#### Calculation Method

One of the following may be selected:

- according to IEEE-1584 published 2002[24]
- according to IEEE-1584 published 2018[25]
- according to NFPA 70E[28]
- according to DGUV 203-077[22]

All of the implemented methods support only AC short circuit calculation. No DC short circuit calculation has been implemented in Arc-Flash Analysis.

#### Fault Location

One of the following may be selected:

- **At User Selection:** selection of either a single location or a pre-defined set of locations.
- **At All Accessible Locations:** locations are all terminals where *Accessible Location* is selected on the *Protection* page of the element dialog.

#### Fault Clearing Times

One of the following may be selected:

- **Use Fixed Times:** in this case, detailed protection models are not required by the calculation, and the following should be defined:
  - If *Get Time from Global* is selected, then define the *Protection Tripping Time* and *Breaker Opening Time*.
  - If *Get Time from Local* is selected, then define the *Maximum Time*; i.e. the maximum fault clearing time used by the Arc-Flash command. The clearing times used by the Arc-Flash command are taken from the *Protection* page of switch elements (*ElmCoup* and *StaSwitch*), with *Switch Type* set to “Circuit Breaker” on the *Basic Data* page.
- **Use Relay Tripping:** in this case, the tripping time is based upon the relay characteristic entered in the protection model (only if the *Status* on the relay(s) *Description* tab is set to “Approved”). The Arc-Flash Hazard Analysis command performs incident energy calculations using this tripping time, and the tripping time based on a reduced fault current, as specified on the *Advanced Options* page (parameter *Arcing Current Variation*). If *Use Relay Tripping* is selected:
  - **Get Time from** can be set to either:
    - \* **Initial:** in this case the Arc-Flash command determines 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 connected to 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 connected to 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.
  - Define the **Maximum Time**, the maximum fault clearing time used by the Arc-Flash command.

### Short-Circuit Calculation

Pointer to the *Short-Circuit Calculation* command.

### Show Output

If selected, the pointer to the Output of Results can be modified. See Section [34.4](#) for details.

---

**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.

---

## 34.3.2 Arc-Flash Hazard Analysis Advanced Options Page

### Arc-Flash Calculation Options

The following parameters are used according to the IEEE-1584 and NFPA 70E standards. They are therefore only available if one of these standards has been selected on the *Basic Options* page.

- **Arcing Current Variation:** defines the percentage by which the bolted-fault current is reduced for the second calculation (see [34.3.1](#)).
- **Energy at Flash-Protection Boundary:** If IEEE 1584-2018 is selected this parameter will be applied only on the Lee equation.
- **Apply arcing current variation to nodes > 1kV:** available only for IEEE-1584 version 2002. This parameter allows a user to chose if reduction factor is going to be applied on the middle voltage nodes.
- **Iterative Energy Calculation:** uses the actual current for each step of the arc-flash calculation for the energy calculation. Therefore, on the *Basic Options* page, the *Fault Clearing Times* must be set to *Use Relay Tripping* and the time must be used from the *Iteration*.

### PPE-Ratings

One of the following options can be selected for the PPE-Ratings:

- **Acc. to NFPA 70E[28]:** in this case default values from the standard are used. These ratings can only be used for the IEEE-1584 and NFPA 70E standards.
- **Acc. to DGUV 203-077[22]:** 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 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.

### Result Boxes

To show the default set of Arc-Flash results on the single line graphic (Boundary Distance, PPE Category, and Incident Energy), right-click the terminal result box and select *Format for Short Circuit Nodes* → *Energy, Flash Boundary, PPE*. Arcing current and fault clearing time results can also be displayed.

### 34.4.2 Arc-Flash Reports Dialog

The Arc-Flash Reports (*ComArcreport*) dialog can be used to configure the output of tabular results from an Arc-Flash calculation. Additionally, if option *Create label database* is selected “Database” and “Template” files can be selected in order to facilitate the preparation of Arc-Flash Hazard warning labels. The following inputs are available in the Arc-Flash Reports dialog.

#### Create Label Database

If selected, *Database* and *Template* file names on the tab “Label Database” should be specified. A default template is selected by *PowerFactory*. Note that the *Database* Excel file should not be open when *Create Label Database* has been ticked and while the command is running.

#### Available Variables and Selected Variables

Variables to be included in the selected label database report can be selected or deselected (in which case they will be visible in the *Available Variables* pane).

#### Create Tabular Report

Select whether to *Create Tabular Report*. Once the tabular report has been created, results can be filtered according to *Min. PPE-Category* and *Min. Incident Energy*.

For each location that is represented in the tabular report, intermediate results can be shown. Intermediate results contain information on full arc and reduced arc current. The one that is responsible for the higher incident energy represents the worst case and is the one shown in overview report.

After being executed, the tabular report can be exported in HTML or XLSX format, by using the corresponding export icon located at right upper corner of tabular report.

---

**Note:** If the incident energy exceeds the incident energy in the maximum PPE category, the result is “N/A”.

---

### 34.4.3 Arc-Flash Labels

The *Create Label Database* option, handled by a DPL script, triggers an export of the selected variables to a Microsoft Excel file at the selected location. After the export of label data, a copy of the given label template will be stored at the same location as the Excel file and renamed accordingly (i.e. if the Excel file is named “ArcFlash.xls”, the copy of the template will be named “ArcFlash.doc”). If a template file with this name already exists, the user will be prompted as to whether it should be overwritten. The template copy will be opened after the export is complete. Microsoft Word’s Mail Merge feature can be used to create a series of labels based on the template and the Excel data file. To link the template copy with the database:

- Go to the “Mailings” tab. In the “Start Mail Merge” group click on “Select Recipients”.
- From the drop-down menu, select “Use Existing List...”, and then select the label database Excel file.
- Still on the “Mailings” tab, in the “Preview Results” group, click on “Preview Results” to view the label(s).

- To store or print the finished labels, still on the “Mailings” tab, in the “Finish” group, click on “Finish & Merge”.

For more information about the Mail Merge and how to create a template, refer to the MS-Word help.

Also note that data can be copied from the *Flexible Data* tab in the Data Manager in *PowerFactory* for post-processing and creation of labels.

## 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.

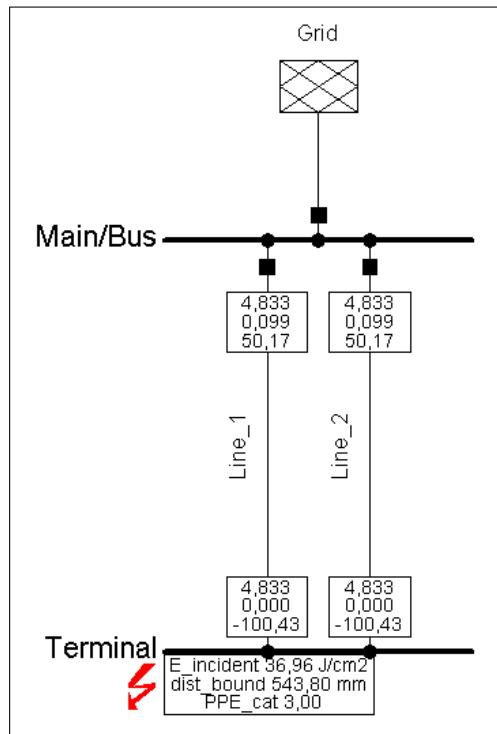


Figure 34.5.1: Example network single line graphic

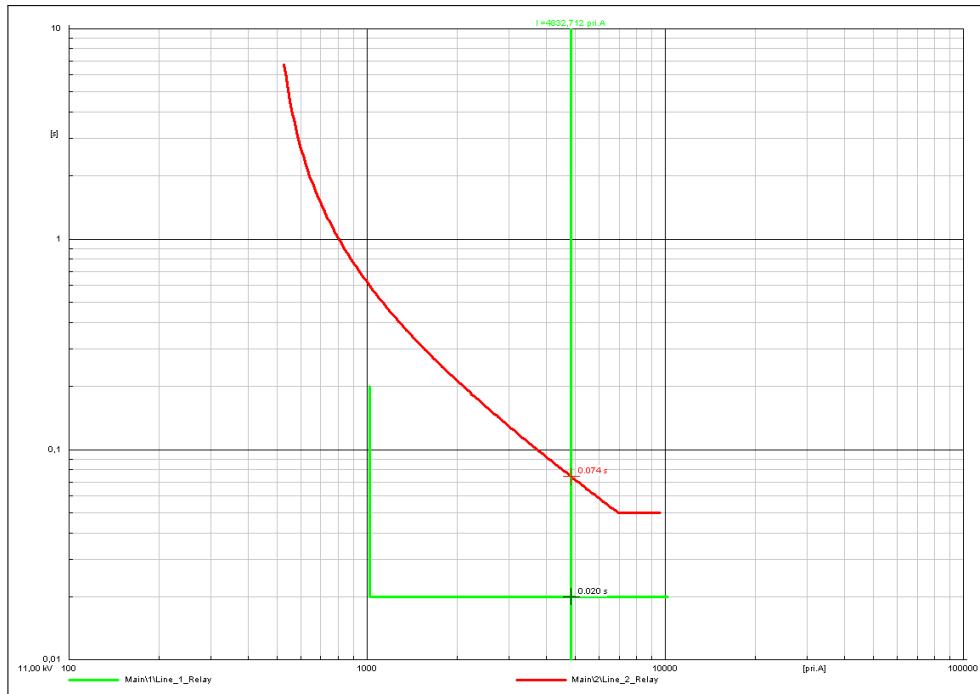


Figure 34.5.2: Protection characteristics

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: 583 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, 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:  $24 \text{ J/cm}^2$ .
  - Boundary Distance: 441 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.

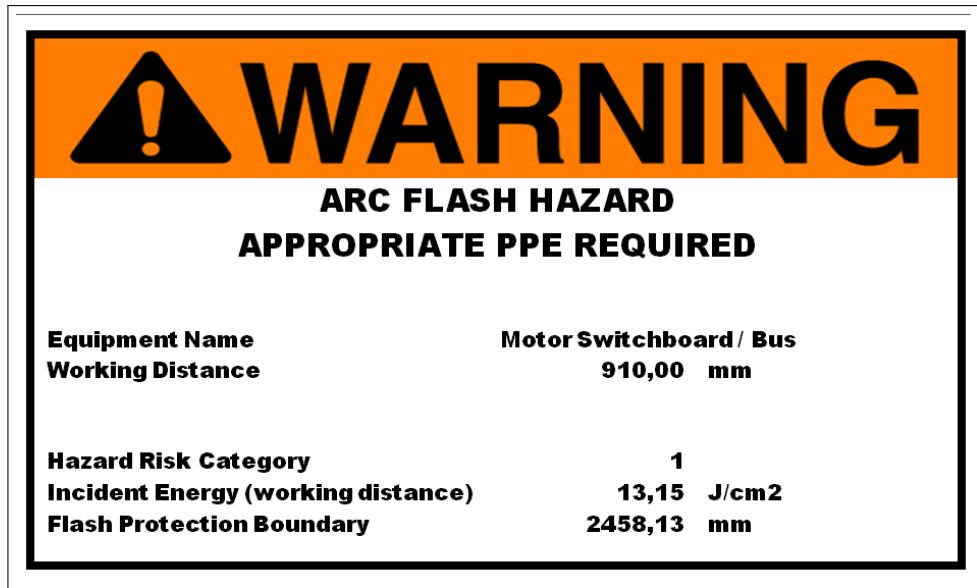


Figure 34.5.3: Example arc-flash warning label

# Chapter 35

## Cable Analysis

The cable analysis toolbox consists of two calculation tools:

- Cable Sizing command (ComCabsiz)
- Cable Ampacity calculation (ComAmpacity)

and one reporting command (ComCablereport).

Access to the cable analysis toolbox is illustrated in Figure 35.0.1.

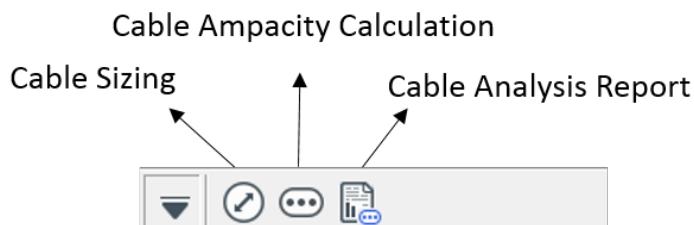


Figure 35.0.1: The cable analysis toolbox

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.

The reporting command is used to output the results of these analyses in a convenient format which can easily be exported from the software if necessary.

In addition to the tools and reporting command, 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 (*ElmLnsec*), line types (*TypLne*), single core cable types (*TypCab*) and multi core/pipe-type cable types (*TypCabmult*). For all these model classes the data required to parameterise the cable analysis features can be found on the *Cable Analysis* 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 can be used for *verification* or *recommendation* of cable types:

- *Verification* - The existing cable types which have been selected for a given network model are checked against a given set of performance constraints.
- *Recommendation* - Appropriately rated cable types are selected from a specified library, such that they meet the specified performance constraints applicable to the given network model. The user is given the option to automatically apply the recommended cable types to the network model. Recommendations may be provided on a network model which does not have any cable types defined (the elements including cable lengths should be defined).

In both cases static analysis (load flow and in some cases short circuit as well) is used to determine the performance of the network components against the applicable constraints. The constraints to be applied are determined depending upon which of the following methods are selected:

- *International Standards Method*. The suitability of the assigned cable types are verified or recommended, taking account of deratings in accordance with the selected *International Standard* (IEC 60364-5-52, NF C15-100, BS 7671, NF C13-200, VDE 0100-520). The deratings are determined according to data entered on the Cable Analysis page of the relevant cable objects.
- *Cable Reinforcement Method*. The suitability of the assigned cable Types are verified or recommended, according to user-defined voltage, thermal, and short-circuit constraints specified on the constraints page of the command dialog. The rated currents of the relevant cables without derating is used in the evaluation.

To access the Cable Sizing command (*ComCabsize*), select the *Change Toolbox* icon (▼), *Cable Analysis*, and then select the *Cable Sizing* icon (⌚).

## 35.2 Calculation Options

### 35.2.1 Basic Options Page

#### 35.2.1.1 Method

Select to execute the *Cable Sizing* command based on either:

- *International Standards* applicable to low-voltage networks up to 1kV, IEC 60364-5-52, VDE 0100-520, NF C15-100 and BS 7671, or applicable to medium-voltage networks 1kV to 33kV, NF C13-200. Refer to the standards for further details.
- *Cable Reinforcement* with user-defined types and constraints.

**Note:** Standards tables for cable ampacity, cross-section, derating factors, and impedances are stored in the *Database* → *System* → *Modules* → *Cable Sizing* folder. 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.

---

### 35.2.1.2 Lines/Feeders

- If *Method* is set to *International Standards*, specify the Line/s for the Cable Sizing analysis.
- If *Method* is set to *Cable Reinforcement*, specify the Feeder/s for the Cable Reinforcement analysis.

### 35.2.1.3 Mode

- If *Verification* is selected, then the command will assess the suitability of the existing cable types:
  - For the *International Standards* Method, the command will verify the suitability of the cable in accordance with the selected standard.
  - For the *Cable Reinforcement* Method, the command will verify the suitability of the cable in accordance with the selected constraints and / or network consistency criteria. At least one of *Thermal Loading Limits*, *Consider Voltage Drop Per Terminals*, *Consider Voltage Drop Along Feeder*, *Short Circuit Loading Limits*, and *Network Consistency* must be selected.
- If *Recommendation* is selected:
  - For the *International Standards* Method, the command will create new cable types for the low voltage and medium voltage grids according to the selected international standard. The cable derating factor will be set based on the installation method, specified on the cable element's Cable Analysis page. Types will be created in the target folder, or if no folder is selected, inside the Equipment Type Library.
  - For the *Cable Reinforcement* Method, the command will recommend cable types for those cables without Types yet defined, and those that cause violations of the specified constraints. Reference to a folder that contains the overhead / cable types to be considered should be provided. This may be a global library, however it is recommended that the available types be stored in a local project library. *PowerFactory* will automatically select the cables with a voltage rating suitable for the cable element.

---

**Note:** Line cost data in \$/km is entered on the Cable Sizing page of the cable type.

---

### 35.2.1.4 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 command and short-circuit command (if applicable) used by the optimisation algorithm. For a *Cable Reinforcement* calculation in *Verification Mode*, the user can optionally consider Short Circuit Loading Limits. The Short Circuit Calculation command will also be automatically adjusted based on the calculation method selected. However, if switching between Balanced and Unbalanced representation, the user should ensure that the short-circuit calculation is set to the required fault type.

## 35.2.2 Constraints Page

Constraints options are only applicable if *Cable Reinforcement* is selected on the Basic Options page.

### 35.2.2.1 Thermal Loading Limits

Optionally select to consider *Thermal Loading Limits*. There are two options for thermal constraints:

- *Global Constraints For All Lines*. This is the default option, where individual component thermal limits are ignored. If enabled, a maximum thermal loading percentage must be entered in the *Maximum Thermal Loading* field.
- *Individual Constraint Per Line*. Select this option to automatically consider 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 cable.

### 35.2.2.2 Consider Voltage Drop Per Terminals

Optionally select to *Consider Voltage Drop Per Terminals*. There are two options for terminal voltage drop constraints:

- *Global Constraints For All Terminals* (absolute value). If selected, a lower voltage limit must be entered in the *Lower Limit of Terminal Voltage* field.
- *Individual Constraint Per Terminal*. Note, the voltage limit is specified in the Load Flow tab of each terminal.

### 35.2.2.3 Consider Voltage Drop Along Feeder

For balanced calculations, optionally select to *Consider Voltage Drop Along Feeder*. The voltage drop is calculated as the absolute voltage difference between the source terminal of the feeder and the final terminal of the feeder. There are two options for feeder voltage drop constraints:

- *Global Constraints For All Feeders*. If this option is selected, then the maximum voltage drop must be entered in the *Maximum Voltage Drop* field.
- *Individual Constraint Per Feeder*. Note, the maximum voltage drop is specified in the Load Flow tab of each feeder.

### 35.2.2.4 Short Circuit Loading Limits

When the *Mode* is set to *Verification*, optionally select to consider *Short Circuit Loading Limits*. Constraints can be entered in the *Maximum Loading* field as a percentage of the rated short-circuit current in the Type data for cables and terminals, etc.

---

**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 41.6: Optimal Capacitor Placement.

---

## 35.2.3 Output Page

### Output

Various output options for the optimisation results are possible.

- *Report Only*: Any new cable types are listed in a pre-defined report displayed in the output window.

- *Modification of Cables Type in the Existing Network*: If this option is selected, the Report will be generated and the optimisation routine will update the network model with the proposed types. Note that this option is only available when the *Mode* is set to *Recommendation* on the *Basic Options* tab.
- *Create a New Variation with Recommended Cables*: If this option is selected, the Report will be generated and the optimisation routine will create a Variation with the proposed modifications. Note that this option is only available when the *Mode* is set to *Recommendation* on the *Basic Options* tab.

### Report

This is a reference (pointer) to the result report output, which details calculation settings, and results of the verification or recommendation. For more information about the result language format see Chapter 19: Reporting and Visualising Results, Section 19.4.2.

### Results

This is a reference (pointer) to the results output. It is possible to select an alternative results file. Results are indexed as follows for the Cable Reinforcement method:

0. Initial value - Initial calculation of all parameters of feeder, *ComCabsiz*, cables and terminals.
1. Thermal cable verification - Only those cables' variables are written which violate the thermal constraint.
2. 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.
3. Thermal cables cannot be solved - Only those cables' variables are written, that are unsolvable and still violate thermal constraints after thermal recommendation process.
4. Voltage verification - Only those terminals' variables are written which violate voltage constraints.
5. 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.
6. Terminals cannot be solved - Only those terminals' variables are written which are unsolvable and still violate voltage constraints after the voltage recommendation process.
7. Consistency verification - Only those terminal's variables are written which violate network consistency.
8. 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.
9. Consistency violation - Only those terminals' variables are written that are unsolvable and still violate network consistency after the recommendation process.
10. Changed cables - Only those cables' variables are written, for which a new cable type is recommended after the complete load flow optimisation process.
11. Short-circuit verification - Only those cables' variables are written which violate short-circuit constraints.

Results are indexed as follows for the International Standards method:

100. Pre-verification results.
101. Post-verification results.
102. Pre-recommendation results.
103. Post-recommendation results.

## 35.2.4 Advanced Options Page

### 35.2.4.1 International Standards Method

If *International Standards* and *Recommendation* is selected on the *Basic Options* page, then configure the *Advanced Options* as follows.

#### Cable Sizing

- Define the *Safety margin for the cable current capacity* in percent. If a non-zero safety margin is entered, a cable with higher capacity is selected.
- Optionally select to *Set cable electrical parameters according to the IEC 60909* to set cable resistance and reactance parameters from conductor cross-section and material according to the IEC 60909 calculation.
- 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.

### 35.2.4.2 Cable Reinforcement Method

If *Cable Reinforcement* is selected on the Basic Options page, then configure the Advanced Options as follows.

#### Network Consistency

This option, if enabled, forces the optimisation routine to complete a final “consistency” check of the Line Type rated nominal 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 35.2.1. 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”.

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 nominal current) of the feeding cables are summated and compared with the sum of the cross sectional area (or nominal 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.

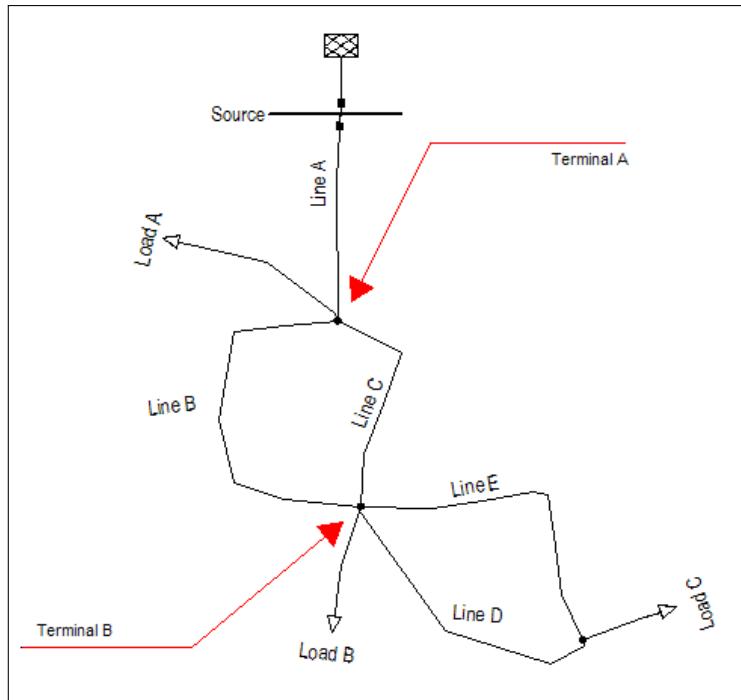


Figure 35.2.1: Example feeder network

### Recommended Options

Available when Mode is set to *Recommendation* on the *Basic Options* tab.

- Specify the *Max. Voltage Deviation in Type Selection* in percent. If “0%” is entered, the rated voltage on the cable type should match the rated 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.
- Optionally select to *Assign Missing Line Types*. Note that for low voltage networks (less than 1 kV) the Cable type rated voltage should be equal to 1 kV.

## 35.3 Cable Ampacity

*PowerFactory*'s Cable Ampacity calculation supports two methods described in literature to determine the current rating of cables:

- *IEC 60287*
- *Neher-McGrath*

At time of writing The latest editions of the norm 60287-1-1 Edition 2 (2014) and 60287-3-3 Edition 1 (2007) are supported.

The Neher-McGrath method is derived from the paper J. H. Neher and M. H. McGrath, “The Calculation of the Temperature Rise and Load Capability of Cable Systems”, AIEE Transactions, Part III, Volume 76, pp 752-772, October, 1957.

To access the Cable Ampacity command (*ComAmpacity*) select the *Cable Ampacity* icon (●●●) from the Cable Analysis toolbar, as illustrated in Figure 35.0.1.

### 35.3.1 Cable Ampacity calculation options

ComAmpacity command window contains following input settings:

- *Method*:
  - IEC 60287 (selected by default)
  - Neher-McGrath
- *Selection* - Selection of either ungrouped or grouped cables. If the option *Cables* is selected, cable deratings will be calculated considering each selected cable or cable system independently. While if the option *Cable Layout* is selected, Cable Layout objects should be chosen and deratings will be calculated taking account of the various special derating effects facilitated by this object see section [35.5.4](#) for more information. For the *Cables* option, when selecting the lines to be analysed, line objects will only be presented by the filter if a Cable System *TypCabsys* is defined for the line.
- *Output*
  - Report only - Once the calculation is completed a tabular report is generated. The result data provides information on the maximum allowed current (i.e. the ampacity) for the cables along with additional information regarding the losses and the resulting temperature rises at the maximum allowed current.
  - Modify derating factor of lines in the existing network - Also generates the report, but in addition it writes a derating factor to the line element which modifies its rated current to match the calculated maximum allowed current. The derating factor is considered to be the ratio between the max. ampacity and the nominal cable current. **Note:This option modifies the network data!**
  - As the previous option but in this case a new variation is created and the modifications to the network model are stored in the new variation.
  - Reports (pointer to the report command)
  - Results (pointer, selection of the results file)
- *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.

## 35.4 Reporting command (ComCablereport)

The cable analysis report command is used to print cable ampacity, cable sizing to international standards and cable reinforcement reports. This command is independent of the calculation, the only requirement is the valid cable analysis 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 cable analysis results file (the label above shall display the main results type, if valid)
- **Report output condition**-Contains options whether to show additional data for the ampacity calculation.

- **Output format**-Either ASCII or Tabular. International standards support both options and cable reinforcement only ASCII report.
- **Title**-User-defined title for the report header.
- **Used Format**-Form to be used for the selected results file and report.

## 35.5 Model Parameters

Before carrying out either of the cable analysis calculations it is necessary to provide a certain amount of input data in the network models specific to the cable analysis functions. This data is entered in the objects representing the cables themselves. The data is entered on the *Cable Analysis* page of their respective type and element object dialogs.

### 35.5.1 Line Type Parameters

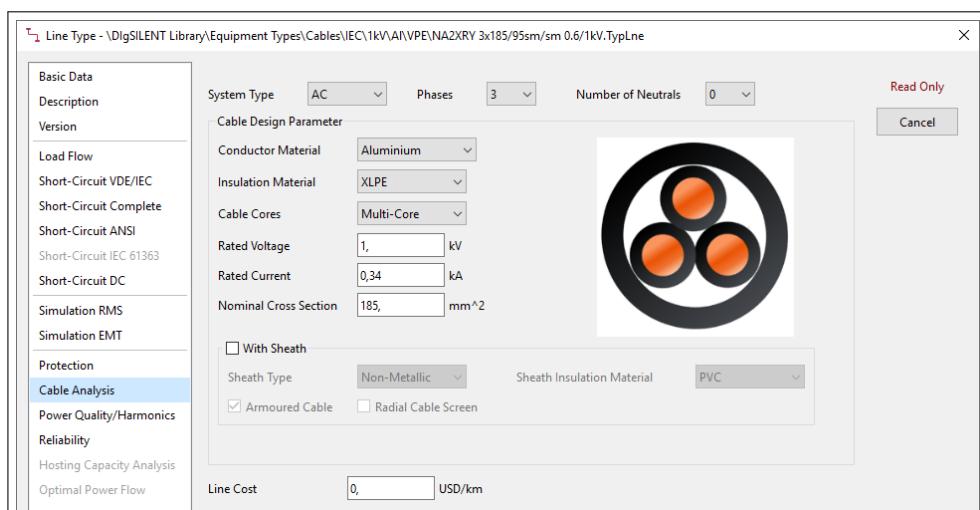


Figure 35.5.1: Cable Analysis Line Type parameters

The parameters defined on the *Cable Analysis* page of the Line Type *TypLne*, are relevant to the cable sizing calculation only. (The cable ampacity calculation is associated with Cable System models, which do not reference *TypLne* objects). The *TypLne* object includes a simplified image of the cable as can be seen in figure 35.5.1 along with various other parameters including some which define the construction of the cable. This type data is used in combination with the associated element data to calculate an appropriate derating for the cable in accordance with an applicable standard. The type data itself is independent of the applicable standard. If the reinforcement method as opposed to the International Standards method is selected for the cable sizing calculation then the data entered on the *Cable Analysis* page of the Cable Type has no impact on the calculation result and need not be specified.

The parameters to be defined are as follows:

- *Conductor Material*. Select either Copper, Aluminium, Aldrey (AlMgSi), Aluminium-Steel or Aldrey-Steel.
- *Insulation Material*. Select either PVC, XLPE, Mineral, Paper, or EPR. Note that 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.

### 35.5.2 Line Element Parameters

Line Element parameters relevant to the Cable Analysis commands are defined on the Cable Analysis page of Line Element objects (*ElmLne*, *ElmLnesec*). If the reinforcement method as opposed to the International Standards method is selected for the cable sizing calculation then the data entered on the *Cable Analysis* page of the Cable Element has no impact on the calculation result and need not be specified. The first five tabs relate to international standards applicable for the Cable Sizing calculation (see section 35.1 for more details) and the final two applicable for the Cable Ampacity calculation (see section 35.3 for more details).

For cable sizing calculations in accordance with an international standard the tab corresponding with the relevant standard should be populated with the applicable installation and environmental conditions.

For cable ampacity calculations the tab corresponding with the relevant calculation method should be populated with the relevant environmental data.

For both types of calculation the data entered on these tabs corresponds with parameters which are defined in the associated standard/method. For more detailed information regarding these parameters the user is advised to consult the source documents.

### 35.5.3 Single Core and multicore/pipe cables

Data may be entered on the cable analysis page of Single Core cable (TypCab) objects and Multi-core/Pipe cable objects. The data entered is valid for the Cable Ampacity calculation only and has no relevance to the cable sizing calculation. The data should be entered in accordance with the relevant cable ampacity calculation method.

### 35.5.4 Cable Layout object

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.

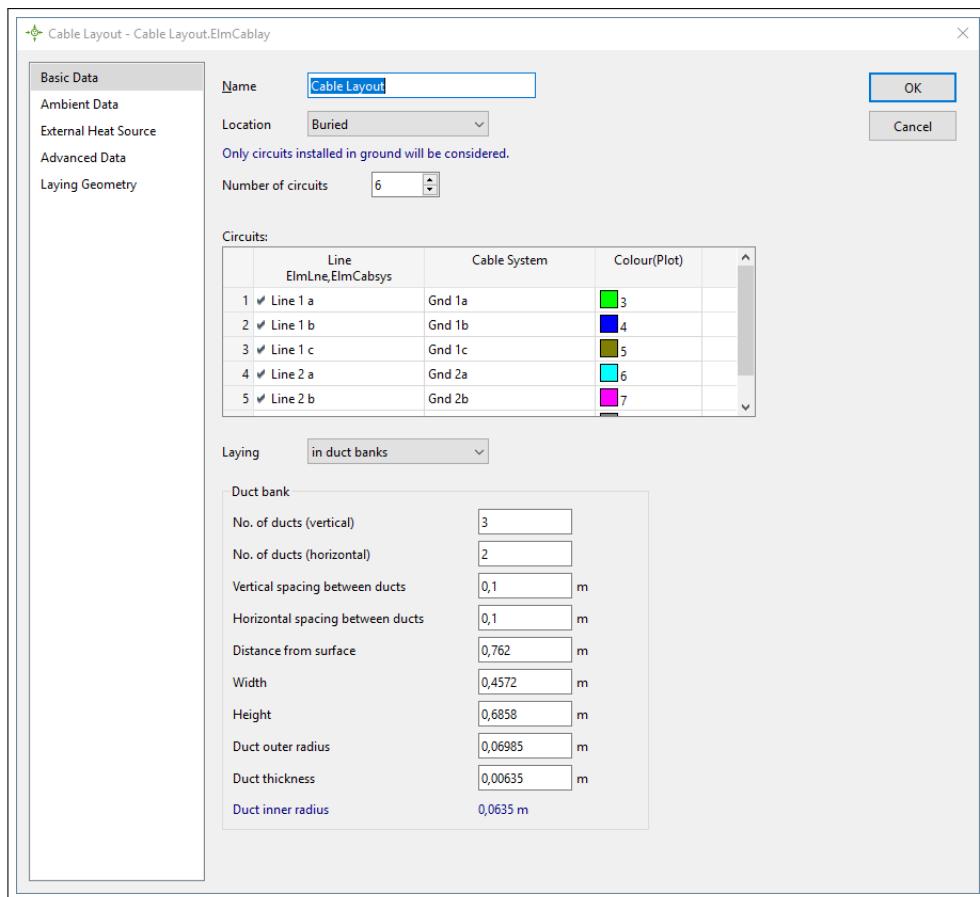


Figure 35.5.2: Cable Sizing Cable Layout parameters - Basic Data

**Note:** The distinction between a Cable System Element 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 is associated with the calculation of cable impedances, whilst the Cable Layout Object 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 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 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 *Define* → *Cable Layout* from the context sensitive menu. The new object is stored in the *Cable Layouts* folder stored in the *Network Data* folder accessible via the *Data Manager*.

- *Basic Data - See figure 35.5.2*
  - 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.
- Duct bank - Used to specify the construction and geometry of the duct bank. Please also see *Laying Geometry* below.

- *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. For further information regarding the usage of the setting parameters, please refer to the source documentation for the relevant method.

- *External Heat Source*

- 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. For further information regarding the usage of the setting parameters, please refer to the source documentation for the relevant method.

- *Laying Geometry* - see figure 35.5.3

- 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 bottom to top.

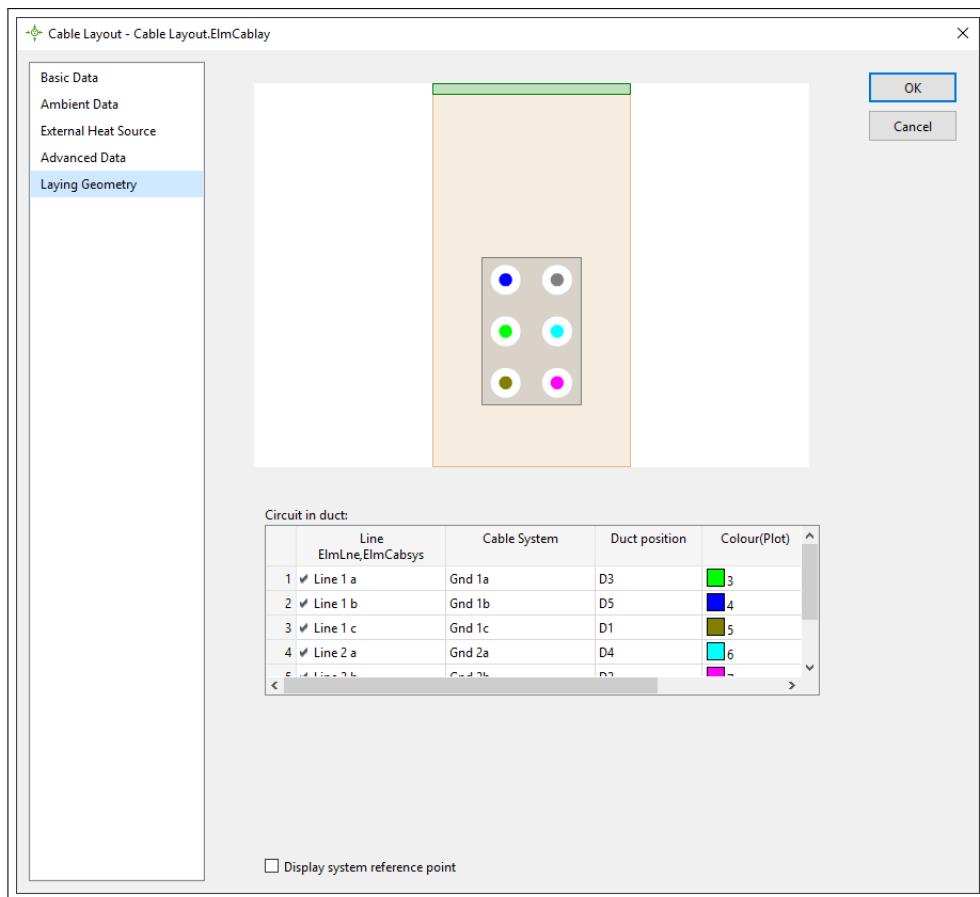


Figure 35.5.3: Cable Sizing Cable Layout parameters - Laying Geometry

# 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 [7] and flicker analysis according to IEC 61400-21 [14])
- [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 [14], 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.

*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/Harmonics → Harmonic Load Flow...*
- *Calculation → Power Quality/Harmonics → Impedance Frequency Characteristic...*
- *Calculation → Power Quality/Harmonics → Flickermeter...*
- *Calculation → Connection Request Assessment...*

Additionally, following the calculation of a harmonic load flow, the icon *Filter Analysis* on this toolbar is activated. This icon is used to open the Filter Analysis (*ComSh*) command dialog. This command analyses results from the most recent harmonic load flow calculation and opens tabular reports listing relevant results for filter installations. The Harmonic Load Flow, Frequency Sweep and Connection Request functions and their usage are described in this chapter. The Flickermeter function is described in Chapter 37 (Flickermeter).

## 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 36.5 (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

#### 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 19.7 (Plots)). The results of the analysis at the given *Output Frequency* are shown in the single line diagram.

### 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 Calculate Flicker

**Calculate Flicker.** When selected, the long- and short-term flicker disturbance factors are calculated according to IEC standard 61400-21. See Section 36.6 (Flicker Analysis (IEC 61400-21)) for more detailed information.

### 36.2.1.5 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 36.7.2.1 (Calculation of Impedance Ratios)) at 3-phase buses are also calculated. See Section 36.7 (Short-Circuit Power) for more detailed information.

### 36.2.1.6 Result Variables and Load Flow

**Result Variables.** This option is available if *Calculate Harmonic Load Flow* option *All Frequencies* has been selected, and is used to select the target results object for storing the results of the harmonic load flow. See Section 36.9 (Definition of Result Variables) for more information regarding specifying and defining result variables.

**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 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 [7]) if at least one harmonic source in the network is defined as IEC 61000 (see Section 36.5.1 ([Definition of Harmonic Injections](#))). 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.

## 36.2.3 Advanced Options

### 36.2.3.1 Calculate HD and THD

**Based on fundamental frequency values.** All values are based on fundamental frequency values, as defined by various standards. See Section 36.2.4 for mathematical descriptions of the calculated indices.

**Based on rated voltage/current.** All values are based on the rated voltage/current of the buses and branches in the network, respectively. See Section 36.2.4 for mathematical descriptions of the calculated indices.

### 36.2.3.2 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.

### 36.2.3.3 Calculate Harmonic Factor (HF)

The calculation of the harmonic factor is optional as it requires extra processing time. Leave unticked if performance is critical. See Section 36.2.4 for mathematical descriptions of the calculated indices.

### 36.2.3.4 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.

### 36.2.3.5 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.

### 36.2.4 Harmonics Indices

The Harmonic Load Flow calculation in *PowerFactory* yields a vast number of results for network elements. In this section, some of the more prominent harmonic indices are described mathematically. A full list of available result variables and their definition is given in the dedicated documentation about [Result Variables for Harmonics Analysis](#). The result variables listed below are described in terms of current, but may be identically described for voltage by substituting  $V$  for  $I$  in all cases. It should be noted that the reference value used in the equations depends upon the user selection in the Harmonic Load Flow command dialog (as described in Section 36.2.3). The two possible options are:

- (i) based on fundamental frequency values:

$$I_{ref} = |I_1| \quad (36.1)$$

or (ii) based on rated voltage/current:

$$I_{ref} = |I_{rated}| \quad (36.2)$$

The harmonic distortion (HD) of a waveform describes the individual harmonic contribution and is described in terms of current by:

$$HD_I = \frac{I_{rms_h}}{I_{ref}} \quad (36.3)$$

where  $I_{rms_h}$  is the RMS value of the waveform at harmonic order  $h$ .

The harmonic factor (HF) is a measure of the individual harmonic contribution and is calculated according to:

$$HF_I = \frac{I_{rms_h}}{\sqrt{\sum_{h=1}^n (I_{rms_h})^2}} \quad (36.4)$$

where  $n$  is the highest order for which harmonic injections have been defined.

The total harmonic distortion (THD) is the most common harmonic index, and describes the total harmonic distortion of a current waveform as follows:

$$THD_I = \frac{\sqrt{\sum_{h=2}^{h_{max}} (I_{rms_h})^2}}{I_{ref}} \quad (36.5)$$

where  $h_{max}$  is the user-specified maximum harmonic order up to which should be included the calculation of THD. This is the parameter **Max. harmonic order for calculation of THD and THF** available in the harmonic load flow command dialog, and is explained further in Section 36.2.3.

The total harmonic factor (THF) is based on RMS values and is calculated as:

$$THF_I = \sqrt{\frac{\sum_{h=2}^{h_{max}} (I_{rms_h})^2}{\sum_{h=1}^{h_{max}} (I_{rms_h})^2}} \quad (36.6)$$

The total arithmetic distortion (TAD) is calculated as:

$$TAD_I = \frac{1}{|I_1|} \cdot [I_{\Sigma A} - |I_1|] \quad (36.7)$$

where  $|I_1|$  is the magnitude of the current at the fundamental frequency, and the arithmetic sum value is given by:

$$I_{\Sigma A} = \sum_{h=1}^n |I_h| \quad (36.8)$$

where  $h$  is the harmonic order and  $n$  is the highest order for which harmonic injections have been defined.

The total power (TP) describes the power absorbed over all frequency components, as shown in:

$$TP = \sum_{h=1}^n P_h \quad (36.9)$$

where  $h$  is the harmonic order and  $n$  is the highest order for which harmonic injections have been defined.

In addition, the following THD variables are provided (note that for the corresponding TAD variables, similar definitions apply):

THDbal, excluding zero sequence:

$$THDbal = \sqrt{THD1^2 + THD2^2} \quad (36.10)$$

THDtot, including zero sequence:

$$THDtot = \sqrt{THD1^2 + THD2^2 + THD0^2} \quad (36.11)$$

THDmx, maximum of phase values:

$$THDmx = \max[THD:A, THD:B, THD:C, THD:N] \quad (36.12)$$

THDint, only considers integer harmonic orders; maximum of phase values:

$$THDint = \max[THDint:A, THDint:B, THDint:C, THDint:N] \quad (36.13)$$

THDnint, only considers non-integer harmonic orders; maximum of phase values:

$$THDnint = \max[THDnint:A, THDnint:B, THDnint:C, THDnint:N] \quad (36.14)$$

THD\_2, only considers even harmonic orders; maximum of phase values:

$$THD\_2 = \max[THDint:A, THDint:B, THDint:C, THDint:N] \quad (36.15)$$

THD\_3, only considers tripled harmonic orders; maximum of phase values:

$$THD\_3 = \max[THDint:A, THDint:B, THDint:C, THDint:N] \quad (36.16)$$

It should be noted that for networks containing IEC 61000 harmonic current sources, result variables for the voltage angle and current angle are not applicable (as the angles cannot be known). Additionally, the following result variables are available:

- $ku$ ,  $ki$ : Voltage and current diversity factors, respectively (always '1' for networks containing only phase correct sources). The voltage diversity factor is shown in (36.17):

$$ku = \frac{U_h^2}{(\sum|U|)^2} \quad (36.17)$$

where  $U_h$  is the IEC 61000 harmonic voltage magnitude as defined in (36.18) and  $|U|$  is the voltage magnitude.

- $HD$ ,  $THD$  and  $TAD$  for non-integer 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.

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.

The following sections describe the options available in the harmonic frequency sweep command.

### 36.3.1 Basic Options

#### 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 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 36.9 (Definition of Result Variables) for more information regarding specifying result variables.

**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

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.

## 36.4 Filter Analysis

The *Filter Analysis* command is a special form of the *Output of Results* command (*ComSh*), whose function is to generate a report. The Filter Results and Filter Layout command generate a table report. It outputs a summary of the harmonics for the terminals/busbars and branch elements at the frequency specified in the *Output Frequency* field of the harmonic load flow command. It also reports the parameters and different variables for the installed filters.

The filter analysis command can be activated using the *Filter Analysis* button  or by using the *Output Calculation Analysis* button  on the main icon bar (see also Section 19.4.2: Output of Results). This will open the same dialog as that used for the reporting of harmonic results, as displayed in Figure 36.4.1. This command can alternatively be launched from the single line graphic, after selecting one or more elements, and right-clicking and selecting *Output Data... → Results*. Results will then be output for the selected elements. It should be noted that elements should be selected according to the type of report being generated. This means that for *Busbars and Branches* only terminals and branches should be selected, for *Busbars/Terminals* only terminals should be selected; and for *Filter Layout* and

*Filter Results* only shunts and harmonic filters should be selected. In the dialog, the *Output Frequency* specified in the harmonic load flow command is displayed in red text (see top of dialog in Figure 36.4.1).

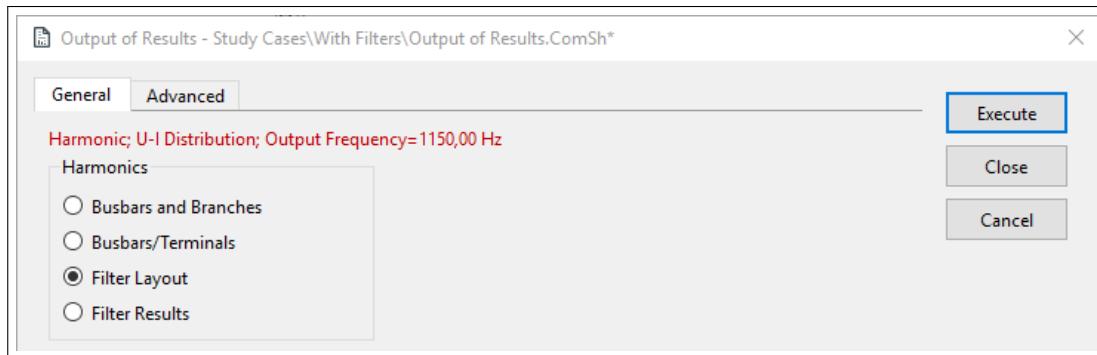


Figure 36.4.1: Filter Analysis Report Command (*ComSh*)

There are four different reports available for selection:

**Busbars and Branches.** This displays the results of the harmonic load flow for all node and branch elements in the network. The distortion for various electrical variables is printed and summarised.

**Busbars/Terminals.** For the electrical nodes, the rated voltage, the voltage at the calculation frequency, as well as RMS values and distortion at the nodes are displayed.

**Filter Layout.** The filter layout of all active filters in the network is reported for the given frequency. The rated values and impedances of the filter as well as the type and vector group are available as optional columns in the tabular report. Additionally, the currents through the different components are shown, as are the losses.

**Filter Results.** The filter results show the main layout of all filters in the network for the nominal frequency and relevant results for all other frequencies evaluated from the harmonic load flow, e.g. the voltage across the capacitance or the currents through the internal components of the filters.

**Use Selection.** Results will only be reported for elements defined in a selection. A selection of elements can be defined by selecting them either in the single line graphic or in the Data Manager, right-clicking and choosing *Define... → General Set*. This *General Set* then exists in the Study Case and can be selected when the *Use Selection* option is activated.

The default format used for the report in the output window is defined in the *Used Format* section on the Advanced tab of the dialog and can be set or changed by clicking on the *Filter Layout* arrow button → .

## 36.5 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 36.5.1 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, 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, ElmVacbi</i>	U		Spectrum defined in element
External Grid	<i>ElmXnet</i>	U		Spectrum defined in element

Table 36.5.1: List of models which can be configured as harmonic sources

See Section 36.5.1 ([Definition of Harmonic Injections](#)) for information on how to define harmonic injections 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](#).

### 36.5.1 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 36.5.3 for a complete summary.

### IEC 61000 Harmonic Sources

The IEC 61000-3-6 standard [7] describes a “second summation law”, applicable to both voltage and current, which is described mathematically as:

$$U_h = \sqrt[\alpha]{\sum_{m=0}^N U_{h,m}^\alpha} \quad (36.18)$$

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 36.5.2 [7].

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 36.5.3. 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 36.5.4 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 36.5.3 ([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 36.5.4 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*

Alpha Exponent Value	Harmonic Order
1	$h < 5$
1.4	$5 \leq h \leq 10$
2	$h > 10$

Table 36.5.2: IEC 61000-3-6 Summation Exponents According to Harmonic Order

object is assigned. The consideration by the calculation of the sequence components of harmonic injections is given in Table 36.5.3.

### 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:

$$I_h = k_h \cdot I_r \quad (36.19)$$

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:

$$I_h = k_h \cdot e^{\Delta\varphi_h} \cdot I_1 \cdot e^{h \cdot \varphi_1} \quad (36.20)$$

where

$$k_h = \begin{cases} I_h/I_1 & \text{if balanced} \\ I_{ah}/I_{a1} & \text{if unbalanced phase a} \\ I_{bh}/I_{b1} & \text{if unbalanced phase b} \\ I_{ch}/I_{c1} & \text{if unbalanced phase c} \end{cases} \quad (36.21)$$

$$\Delta\varphi_h = \begin{cases} \varphi_h - h \cdot \varphi_1 & \text{if balanced} \\ \varphi_{ah} - h \cdot \varphi_{a1} & \text{if unbalanced phase a} \\ \varphi_{bh} - h \cdot \varphi_{b1} & \text{if unbalanced phase b} \\ \varphi_{ch} - h \cdot \varphi_{c1} & \text{if unbalanced phase c} \end{cases} \quad (36.22)$$

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 36.20 and 36.21) 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 36.5.3: 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 36.5.4: Consideration of Sequence Components of Harmonic Injections for AC Voltage Source, External Grid and Synchronous Machine

### 36.5.2 Assignment of Harmonic Injections

The assignment of harmonic injections to the elements listed in table 36.5.1 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.

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 36.5.1 (Definition of Harmonic Injections) for further information.

### 36.5.3 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 36.5.1, 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 36.5.2 for a line type.

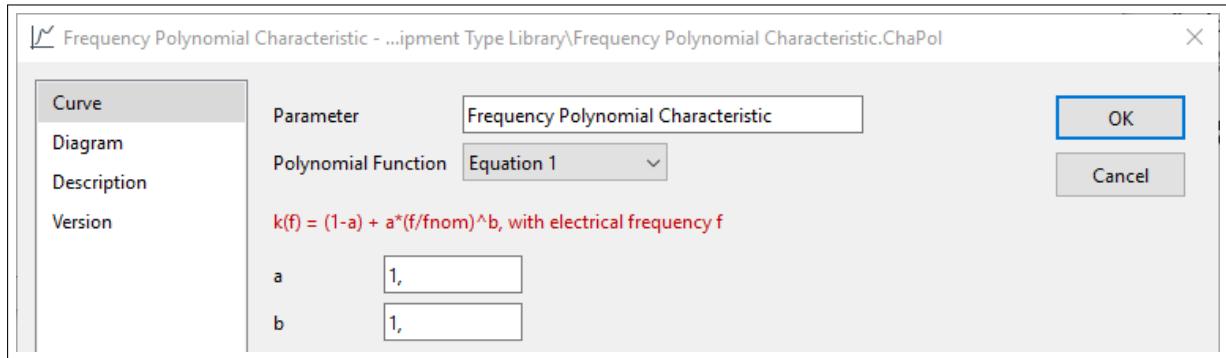


Figure 36.5.1: The Frequency Polynomial Characteristic (*ChaPol*)

For the frequency polynomial characteristic object illustrated in Figure 36.5.1, the formula given by (36.23) is used:

$$y(f_h) = (1 - a) + a \cdot \left(\frac{f_h}{f_1}\right)^b \quad (36.23)$$

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 (36.24):

$$R(f_h) = R \cdot y(f_h) \quad (36.24)$$

An example of the use of the polynomial characteristic for a line type is shown in Figure 36.5.2.

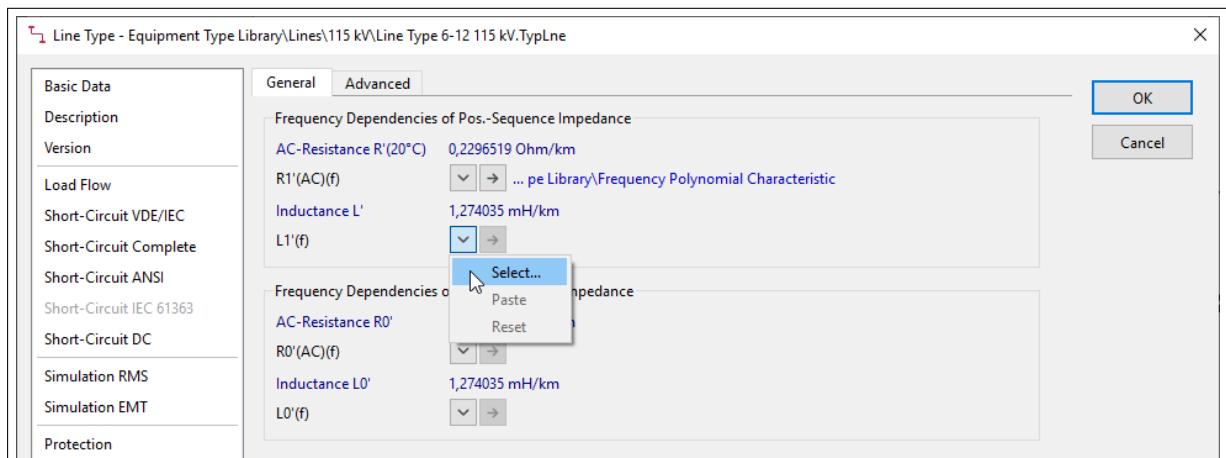


Figure 36.5.2: Frequency Dependencies in a Line Type

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 (*Elmlac*, *Elmlacbi*)

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*).

### 36.5.4 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 36.5 (Modelling Harmonic Sources).

The waveform is calculated according to the following formula:

$$u(t) = \sum_{i=1}^n u(i) \cdot \cos(2\pi(f(i) \cdot t + \varphi(i))) \quad (36.25)$$

where:

<sup>1</sup>Frequency dependencies on spectral densities, considered only in the Frequency Sweep

$i$	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 36.5.3.

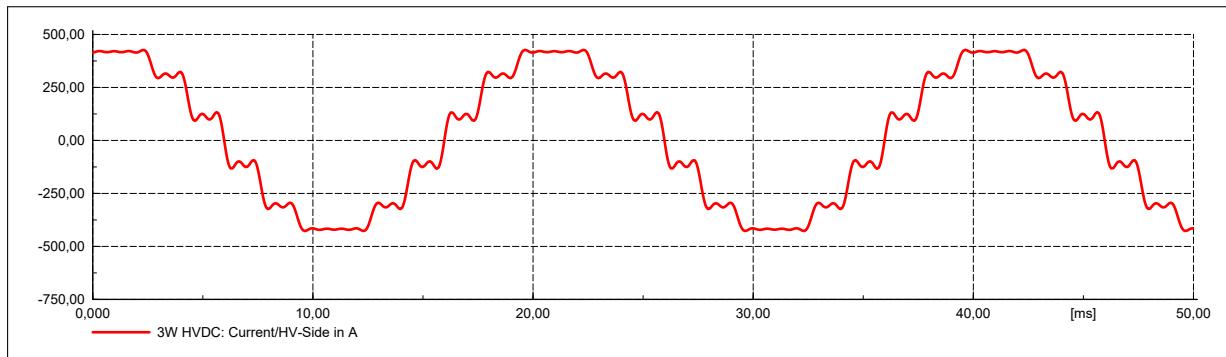


Figure 36.5.3: Use of the Waveform Plot to display Harmonic Distortion

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 usage, settings and tools for this plot type are similar to the curve plot, described in Chapter 19: Reporting and Visualising Results, section 19.7.1, 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 either press the **Calculation** button in the dialog of the plot or select *Calculation* in the context sensitive menu on the plot. The *Settings Waveform Plot* object (*SetWave*) holds the *Step Size* and the *Range* for the calculation of waveforms in the *Waveform Plots* (see Figure 36.5.4).

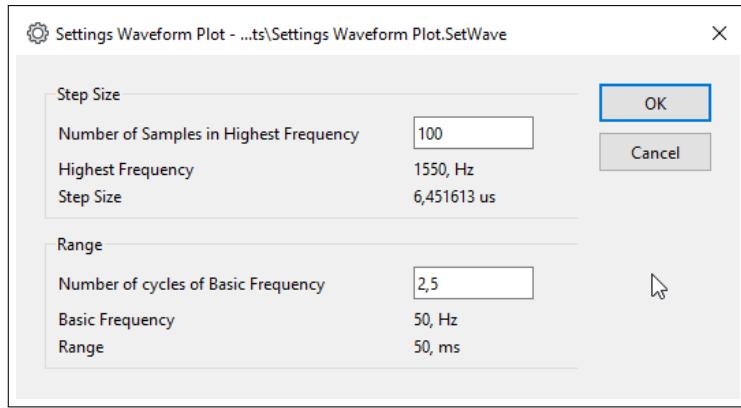


Figure 36.5.4: The waveform plot settings dialog

### 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.

## 36.6 Flicker Analysis (IEC 61400-21)

The IEC standard 61400-21 [14] 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 consumer side which may manifest as 'flicker' (visible flickering effects from light sources), and voltage changes (voltage magnitude being too high or too low).

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 36.6.1 ([Continuous Operation](#)) and 36.6.2 ([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 36.6.6 ([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](#)).

### 36.6.1 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 [14] as:

$$P_{st} = P_{lt} = c(\psi_k, v_a) \cdot \frac{S_n}{S_k} \quad (36.26)$$

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).

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 [14] as:

$$P_{st\Sigma} = P_{lt\Sigma} = \frac{1}{S_k} \sqrt{\sum_{i=1}^{N_{wt}} (c(\psi_k, v_a) \cdot S_{n,i})^2} \quad (36.27)$$

where  $N_{wt}$  is the number of wind turbine generators at the PCC.

### 36.6.2 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 [14] as:

$$P_{st} = 18 \cdot N_{10}^{0.31} \cdot k_f(\psi_k) \cdot \frac{S_n}{S_k} \quad (36.28)$$

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).

$$P_{lt} = 8 \cdot N_{120}^{0.31} \cdot k_f(\psi_k) \cdot \frac{S_n}{S_k} \quad (36.29)$$

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 [14] as:

$$P_{st\Sigma} = \frac{18}{S_k} \left[ \sum_{i=1}^{N_{wt}} N_{10,i} \cdot (k_{f,i}(\psi_k) \cdot S_{n,i})^{3.2} \right]^{0.31} \quad (36.30)$$

Likewise, the long-term flicker disturbance factor under switching operations is defined as:

$$P_{lt\Sigma} = \frac{8}{S_k} \left[ \sum_{i=1}^{N_{wt}} N_{120,i} \cdot (k_{f,i}(\psi_k) \cdot S_{n,i})^{3.2} \right]^{0.31} \quad (36.31)$$

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 [14]:

$$d = 100 \cdot (k_u(\psi_k) \cdot \frac{S_n}{S_k}) \quad (36.32)$$

### 36.6.3 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 36.6.4 (Definition of Flicker Coefficients) and 36.6.5 (Assignment of Flicker Coefficients).

### 36.6.4 Definition of Flicker Coefficients

Flicker coefficients are defined in *PowerFactory* by means of the *Flicker Coefficients* type (*TypFlicker*), as illustrated in Figure 36.6.1. When created, this is stored by default in the *Equipment Type Library* folder in the project tree.

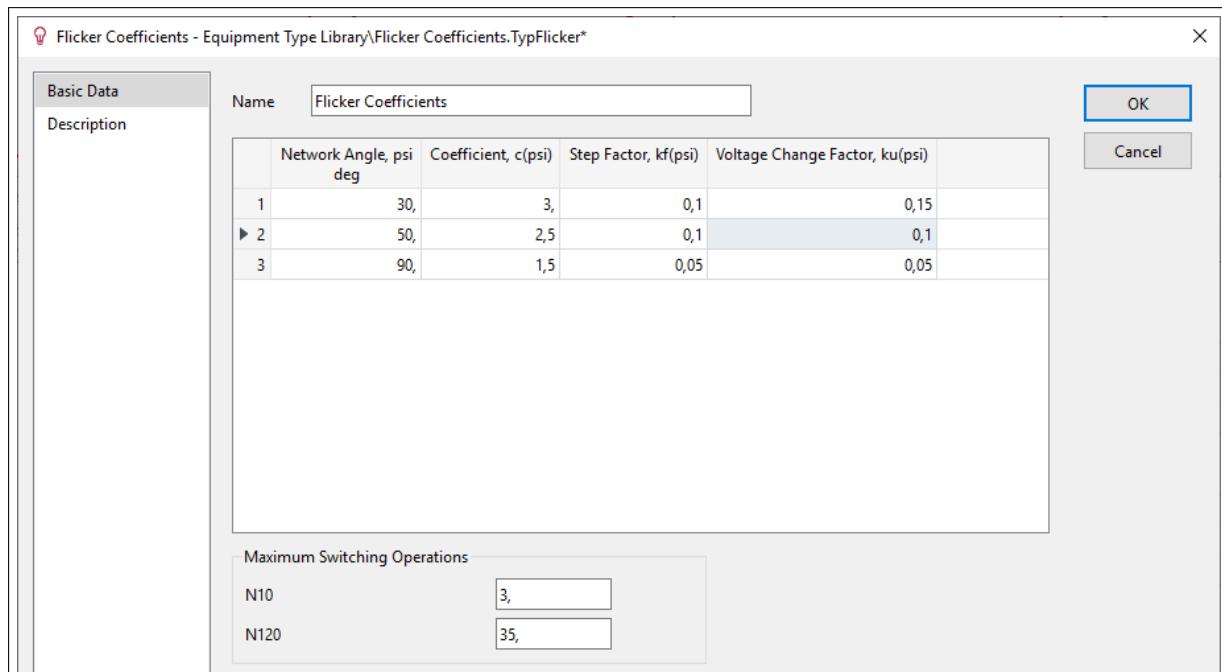


Figure 36.6.1: Definition of Flicker Coefficients using the Flicker Coefficients Type (*TypFlicker*)

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.

### 36.6.5 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.

### 36.6.6 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 [36.6.1 \(Continuous Operation\)](#) and [36.6.2 \(Switching Operations\)](#).

## 36.7 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 \(Basic Options\)](#)), the short-circuit power of the grid under normal operation is available in the calculation results.

### 36.7.1 Balanced Harmonic Load Flow

For the balanced harmonic load flow, the calculation of the short-circuit power,  $Sk$ , at each bus is as follows:

$$Sk = \frac{1}{|Z_{bus}|} \quad (\text{MVA}) \quad (36.33)$$

where  $Z_{bus}$  is the impedance calculated at the bus.

The calculation of the impedance angle,  $psik$ , at each bus is as follows:

$$psik = \angle Z_{bus} \quad (\text{degrees}) \quad (36.34)$$

where  $Z_{bus}$  is the impedance calculated at the bus.

### 36.7.2 Unbalanced Harmonic Load Flow

For the unbalanced harmonic load flow, the calculation of the short-circuit power,  $Sk$ , at each bus is as follows:

$$Sk = \frac{1}{|Z1_{bus}|} \quad (\text{MVA}) \quad (36.35)$$

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:

$$psik = \angle Z1_{bus} \quad (\text{degrees}) \quad (36.36)$$

where  $Z1_{bus}$  is the positive sequence impedance calculated at the bus.

#### 36.7.2.1 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):

$$z2tz1kV = \frac{|Z2|}{|Z1|} \quad (36.37)$$

$$x0tx1kV = \frac{X0}{X1} \quad (36.38)$$

$$r0tx0kV = \frac{R0}{X0} \quad (36.39)$$

### 36.7.3 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 36.7.1 (Balanced Harmonic Load Flow), and 36.7.2 (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 36.7.2.1 (Calculation of Impedance Ratios).

### 36.7.4 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 36.7.3 (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.

## 36.8 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.

### 36.8.1 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 [12] and [16]. These guidelines consider the following aspects of power quality for 50Hz networks, operating at low-voltage (LV), medium-voltage (MV) or high-voltage (HV) levels:

- Voltage changes and flicker
- Voltage unbalance
- Harmonics
- Commutation notches
- Interharmonic voltages

**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 voltage rise 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 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 [12] and [16]. Dialog options pertaining to HV networks are only available for Connection Request elements which are connected to a HV PCC.

### 36.8.1.1 Basic Options

#### Method

- **According to D-A-CH-CZ.** Assesses network disturbances according to [12] and [16].

#### Calculations

All calculations are carried out according to the selection of *Method* described above.

- **Voltage rise (generating stations only)** Calculates the voltage rise 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  $P_{st}$ ,  $P_{lt}$  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.
  - 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.** Calculates the voltage unbalance factor,  $k_U$ , and assesses this according to appropriate limits.
- **Harmonics.** 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.
- **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.

- **Interharmonic voltages.** 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 (HV Networks only)

- **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 [16]), which is also dependent upon the harmonic order. It should be noted that the highest resonance factors within the ranges stated in [16] are chosen.
- **Calculation of emission limits.** Calculates the emission limits according to either the simplified method or the general (detailed) method, as defined in [16].
- **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  $SkV_{akt} = SkV$ .

#### 36.8.1.2 Outputs

##### Report

Prints a Connection Request Assessment summary report in *PowerFactory*'s output window. By selecting the option *Detailed Output* a more detailed report is shown.

## 36.8.2 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 [13] and [17]. 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;

### 36.8.2.1 Basic Options

#### Method

- **According to BDEW, 4th Supplement.** Assesses network disturbances according to [13] and [17].

#### Calculations

All calculations are carried out according to the selection of *Method* described above.

- **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 overloadings 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*s, *ElmVsc*, *ElmVscmono* 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  $P_{lt}$  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.

#### 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*.

#### 36.8.2.2 Outputs

##### Report

Prints a Connection Request Assessment summary report in *PowerFactory*'s output window. By activating the option *Detailed Output* a more detailed report is shown.

### 36.8.3 Connection Request Assessment: VDE-AR-N 4105

The selection of *VDE-AR-N 4105* as the *Method* in the Connection Request Assessment command carries out a power quality assessment according to [15] and [19]. This guideline considers 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
- Admissible 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 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;

#### 36.8.3.1 Basic Options

##### Method

- **According to VDE-AR-N 4105.** Assesses network disturbances according to [15] in version 2011 and [19] in version 2018.

## 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.

- **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 [36.7](#)) at the junction point and at the supply transformer busbar and verifies the results against the VDE-AR-N 4105 limits.
- **Loading of network components.** 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-AR-N 4105 Connection Request elements. If no overloading occurs, the five highest loadings in each case 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 MV network is outaged. The elements that are considered as generation (for outaging) are: *Elm-Sym*, *ElmAsm*, *ElmGenstat*, *ElmPvsys*, *ElmVsc*, *ElmVscmono* 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.
- **Rapid voltage changes.** Calculates the sudden voltage change *d* for each generation unit at the junction point, and assesses these according to appropriate limits.
- **Rapid voltage changes due to switching of the entire plant.** The rapid voltage change as a result of switching of 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 *Plt* of the individual generation units 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 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. For the audio frequency ripple control it verifies if the level is within the permitted range.
- **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 short-circuit current injection defined in any VDE-AR-N 4105 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 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.

### 36.8.3.2 Outputs

#### Report

Prints a Connection Request Assessment summary report in *PowerFactory*'s output window. By activating the option *Detailed Output* a more detailed report is shown.

## 36.8.4 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 [20]. It is valid from the year 2018 and replaces the BDEW Medium-Voltage Guideline (including the 4th Supplement) from 2008 described in section 36.8.2. 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  $P_{A,max} \geq 135kW$ .
- 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  $P_{E,max} \geq 30kW$ .

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 [36.8.3](#)) 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)

#### 36.8.4.1 Basic Options

##### Method

- **According to VDE-AR-N 4110.** Assesses network disturbances according to [20].

##### Calculations

All calculations are carried out according to the selection of *Method* described above.

- **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 the VDE-AR-N 4110 limits.
  - For the calculation of the limits, the following elements are taken into account if selected: 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*, *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.
- **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  $P/I$  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.
- **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.

### 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*.

#### 36.8.4.2 Outputs

##### Report

Prints a Connection Request Assessment summary report in *PowerFactory*'s output window. By activating the option *Detailed Output* a more detailed report is shown.

### 36.8.5 Connection Request Assessment Report

Following the execution of a Connection Request Assessment, a detailed report for each Connection Request element is printed in *PowerFactory*'s output window. The report is divided into the following sections:

- Assessment status (overall, and per calculation). For HV networks, the Assessment Level (1 or 2, according to [16]) at which the Connection Request was approved or not is also output.
- Basic Data

- PCC Data
- Results per Calculation (depending on user selection of Calculations in Connection Request Assessment command dialog.)

## 36.9 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 (*PowerFactory* variable names are italicised):

- Harmonic order (-);
- Frequency (Hz);
- HD (%) (for terminals);
- Voltage across inductor (p.u.) (*url*) (for shunts/filters);
- Voltage across capacitor (p.u.) (*uc*) (for shunts/filters);
- Current through inductor (A) (*IL*) (for shunts/filters);
- Current through resistor Rp (A) (*IRp*) (for shunts/filters);
- Current through capacitor C (A) (*IC*) (for shunts/filters);
- Voltage across capacitor C1 (p.u.) (*uc1*) (for shunts/filters);
- Voltage across capacitor C2 (p.u.) (*uc2*) (for shunts/filters);
- Voltage across resistor Rp (p.u.) (*urp*) (for shunts/filters);
- Voltage across inductor (kV) (*Url*) (for shunts/filters);
- Voltage across capacitor (kV) (*Uc*) (for shunts/filters);
- Voltage across capacitor C1 (kV) (*Uc1*) (for shunts/filters);
- Voltage across capacitor C2 (kV) (*Uc2*) (for shunts/filters);
- Voltage across resistor Rp (kV) (*Urp*) (for shunts/filters);
- Voltage across inductor L1 (p.u.) (*url1*) (for harmonic filters);
- Voltage across resistor Rp1 (p.u.) (*urp1*) (for harmonic filters);
- Current through inductor L1 (A) (*IL1*) (for harmonic filters);
- Current through resistor Rp1 (A) (*IRp1*) (for harmonic filters);
- Current through capacitor C1 (A) (*IC1*) (for harmonic filters);
- Voltage across inductor L2 (p.u.) (*url2*) (for harmonic filters);
- Voltage across resistor Rp2 (p.u.) (*urp2*) (for harmonic filters);
- Current through inductor L2 (A) (*IL2*) (for harmonic filters);
- Current through resistor Rp2 (A) (*IRp2*) (for harmonic filters);
- Current through capacitor C2 (A) (*IC2*) (for harmonic filters);
- Voltage across inductor L1 (kV) (*url1*) (for harmonic filters);

- Voltage across resistor Rp1 (kV) (*urp1*) (for harmonic filters);
- Voltage across inductor L2 (kV) (*url2*) (for harmonic filters);
- Voltage across resistor Rp2 (kV) (*urp2*) (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 Set* and then selecting the variables for recording within these sets. These steps are described in sections [36.9.1 \(Definition of Variable Sets\)](#) and [36.9.2 \(Selection of Result Variables within a Variable Set\)](#), respectively.

### 36.9.1 Definition of Variable Sets

To define a *Variable Set*, 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 *Define* → *Results for Harmonic Load Flow*; or *Define* → *Results for Frequency Sweep*. This will add a new (but still empty) variable set 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 [19.6 \(Results Objects\)](#).

To access the variable sets, 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 set manager dialog will open which displays the list of all defined variable sets for that calculation. After the variable set has been created and its variables have been defined, each variable set contains the variables of interest for a single object. A window is opened automatically following the definition of a new variable set, as shown in Figure [36.9.1](#), displaying the list of variable sets. In Figure [36.9.1](#), three variable sets have been defined for three different network elements.

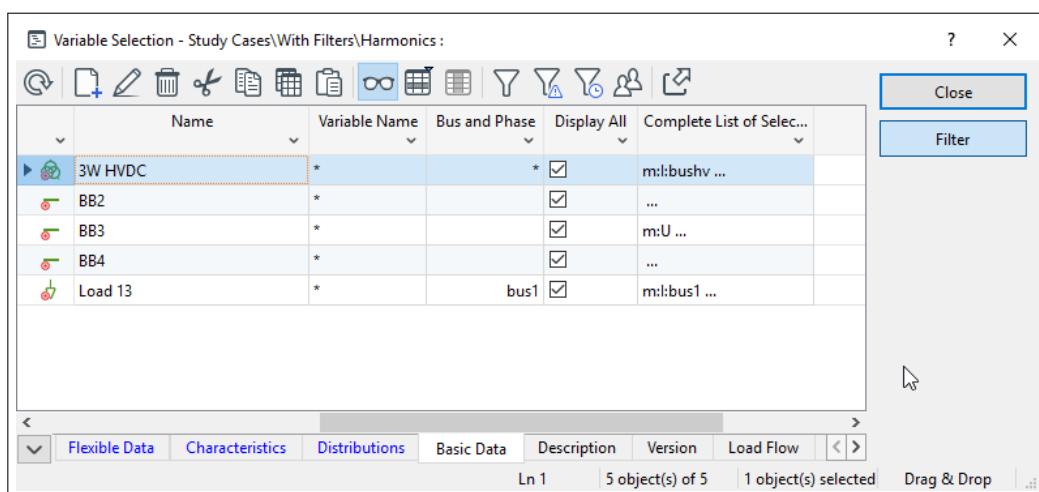


Figure 36.9.1: Example of a List of Variable Sets

A new variable set can also be defined by clicking on the *New* button (), shown in the top left corner

of Figure 36.9.1. By doing this, the *Variable Set* dialog will appear. To proceed with selecting the result variables for the variable set, see Section 36.9.2 (Selection of Result Variables within a Variable Set). For further information on variable sets, refer to Chapter 19: Reporting and Visualising Results.

## 36.9.2 Selection of Result Variables within a Variable Set

The selection of result variables for a variable set can only proceed when the column labelled *Object* for any defined variable set (as shown in Figure 36.9.1) is set. This can be done by either double-clicking the appropriate cell of the *Object* column, or by right-clicking the cell and selecting *Select Element....*. This binds the variable set to a specific object or network element.

A single variable set from the variable sets list can be accessed (and the desired variables defined) by either double-clicking on the icon in the corresponding row, or by right-clicking on the icon and selecting the *Edit* menu option. The *Variable Set* object (*IntMon*) dialog opens. By selecting the *Power Quality/Harmonics* page of this dialog, a list of all result variables that are available for the selected object (applicable to harmonics analysis or frequency sweep) is then available for selection.

Result variables may be added or removed from the set of selected variables by highlighting the desired variable and pressing the left or right arrow buttons << >>. Additionally, different variables are available for selection depending on the selection made from the *Variable Set* drop-down list. For further information on variable sets, refer to Chapter 19: Reporting and Visualising Results.

# Chapter 37

## Flickermeter

### 37.1 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 [11]. *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.

### 37.2 Flickermeter (IEC 61000-4-15)

#### 37.2.1 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 [11]:

$$P_{st} = \sqrt[3]{\frac{(0,0314 \cdot P_{0,1}) + (0,0525 \cdot P_{1s}) + (0,0657 \cdot P_{3s})}{(0,28 \cdot P_{10s}) + (0,08 \cdot P_{50s})}} \quad (37.1)$$

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 [11]:

$$P_{50s} = \frac{P_{30} + P_{50} + P_{80}}{3} \quad (37.2)$$

$$P_{10s} = \frac{P_6 + P_8 + P_{10} + P_{13} + P_{17}}{5} \quad (37.3)$$

$$P_{3s} = \frac{P_{2,2} + P_3 + P_4}{3} \quad (37.4)$$

$$P_{1s} = \frac{P_{0,7} + P_1 + P_{1,5}}{3} \quad (37.5)$$

### 37.2.2 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 [11]:

$$P_{lt} = \sqrt[3]{\frac{\sum_{i=1}^N P_{sti}^3}{N}} \quad (37.6)$$

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 [11] that when  $N = 1$ ,  $Plt = Pst$ .

## 37.3 Flickermeter Calculation

### 37.3.1 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* .

### 37.3.2 Data Source

#### 37.3.2.1 File Input

**Import data from:** specifies the type of data file containing the input data. There are five file types available for selection.

**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 *Power Factory 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

### 37.3.2.2 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.

### 37.3.3 Signal Settings

#### 37.3.3.1 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.

---

#### 37.3.3.2 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.

### 37.3.4 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 37.3.1), however the scaling factor and cut-off frequency are user-definable parameters and are described below.

#### 37.3.4.1 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.

#### 37.3.4.2 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.

#### 37.3.4.3 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 37.3.1: 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 37.3.1. 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 in the Data Manager in *PowerFactory*, refer to Chapter 10: Data Manager, Section 10.6(The Flexible Data Page Tab).

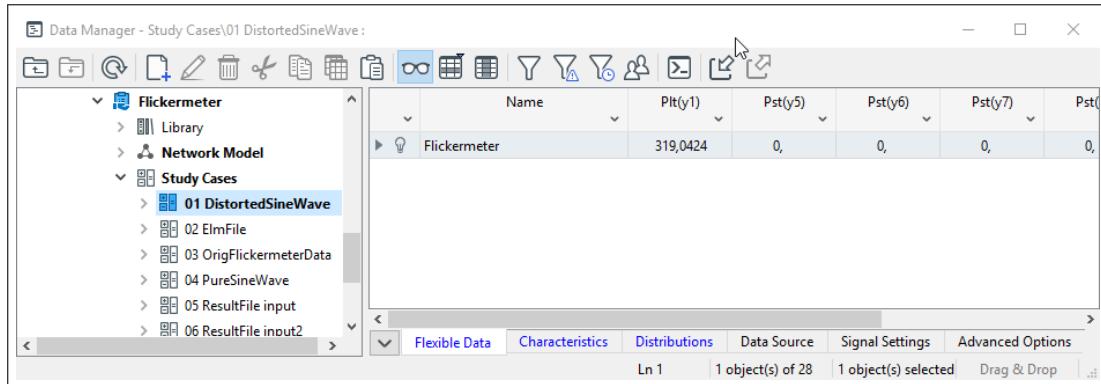


Figure 37.3.1: Using Flexible Data to Access Flickermeter Results

### 37.3.5 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.

#### 37.3.5.1 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 37.3.2, 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 37.3.2 (Data Source) for information on other Flickermeter command options.

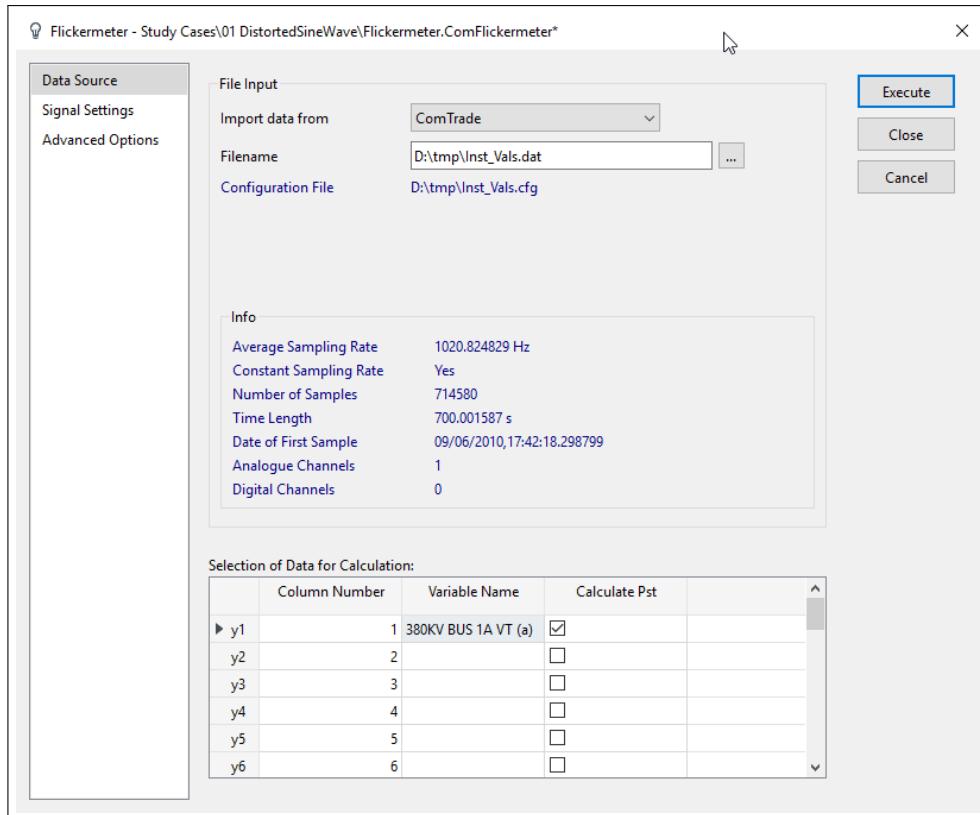


Figure 37.3.2: Configuration of Flickermeter Command for ComTrade Input File

### 37.3.5.2 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 37.3.2 (Data Source) for information on other Flickermeter command options.

### 37.3.5.3 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 37.3.2 (Data Source) for information on other Flickermeter command options.

### 37.3.5.4 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 37.3.3. 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 37.3.3, 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 37.3.2 (Data Source) for information on other Flickermeter command options.

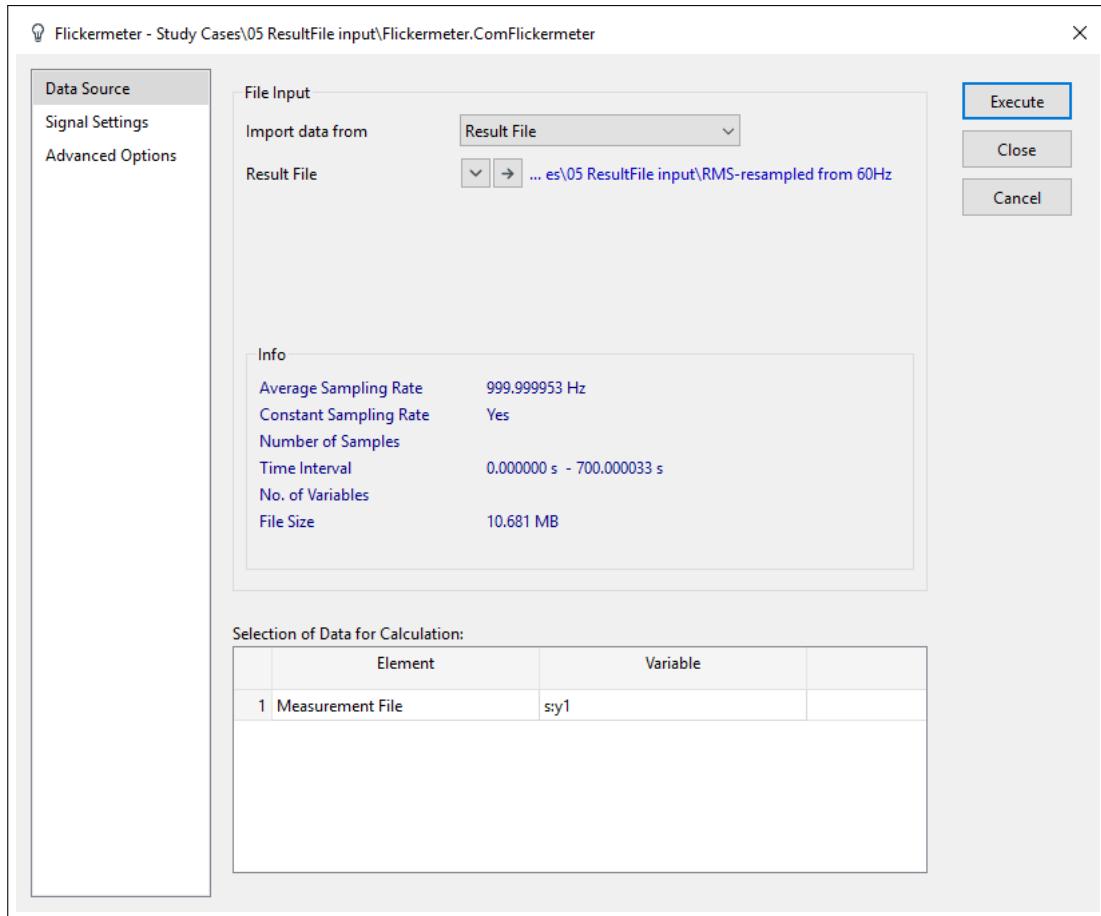


Figure 37.3.3: Configuration of Flickermeter Command for Results File

# Chapter 38

## Optimal Power Flow

### 38.1 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.

An OPF calculation in *PowerFactory* can be initiated by one of the following means:

- Via the main menu *Calculation* → *Optimal Power Flow...* ; or
- By selecting *Optimal Power Flow / Unit Commitment* from the *Change Toolbox* button (▼) and then click on the *Calculate Optimal Power Flow* icon .

In both cases, the calculation is started by pressing the **Execute** button in the OPF command dialog.

### 38.2 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.

#### 38.2.1 AC Optimisation - Basic Options

##### 38.2.1.1 Objective Function

The Optimal Power Flow command dialog, configured for AC optimisation, has a selection of five distinct objective functions. These are:

- Minimisation of losses (total)
- Minimisation of losses (selection)
- Minimisation of costs
- Minimisation of load shedding
- Maximisation of reactive power reserve

- Minimisation of control variable deviations

### **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 *Define → General Set*. 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.

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*), as shown in figure 38.2.1.

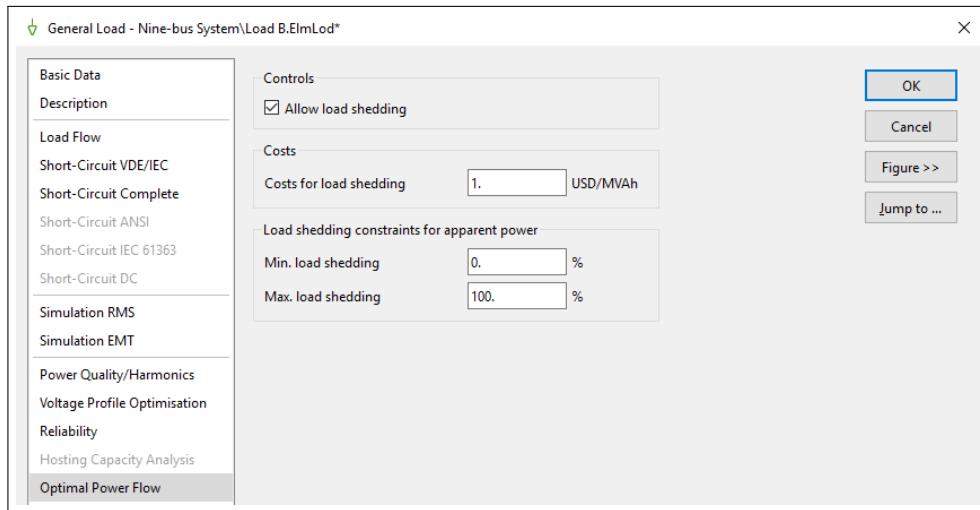


Figure 38.2.1: Editing a Load Element (*ElmLod*) for *Minimisation of Load Shedding*

### 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 nominal 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.

### 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.

#### 38.2.1.2 Controls

The global control parameters can be selected on the *Basic Options* page of the OPF dialog. 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*))
2. Generator/SVS reactive power dispatch (*ElmSym*, *ElmSvs*, *ElmXnet*, *ElmGenstat*, *ElmAsm* (Only if DFIG and type is *TypAsmo*, *ElmPvsys*))
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)
4. Step-Voltage Regulator (*ElmVoltreg*):
  - S-Tap Position (continuous or discrete)
  - L-Tap Position (continuous or discrete)
5. Switchable shunts (*ElmShnt*, *ElmSvs*):
  - Shunts: number of steps (continuous or discrete)

It should be noted that the load scaling factors 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.

These global controls determine which element controls 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 *Basic Options* page of the Optimal Power Flow command dialog.

For example, if the control parameter *Tap position HV-side* of a 3-winding transformer is enabled (as shown in figure 38.2.3), it will only be included in the OPF as a control parameter if the corresponding option *Transformer tap positions* is enabled in the OPF command dialog.

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.

### Setting Individual Model-Based Controls

Each control can be individually selected to take part in the optimisation. Specifically, for each generator, each transformer, each shunt and each static var system, the user can check the corresponding *Controls* flag on the *Optimal Flow Page* page of the element's dialog. Network elements and their available controls are listed below:

## 1. Generators (*ElmSym*, *ElmXnet*, *ElmGenstat*, *ElmPvsys*, *ElmAsm* (only if DFIG and type is *Ty-pAsmo*))

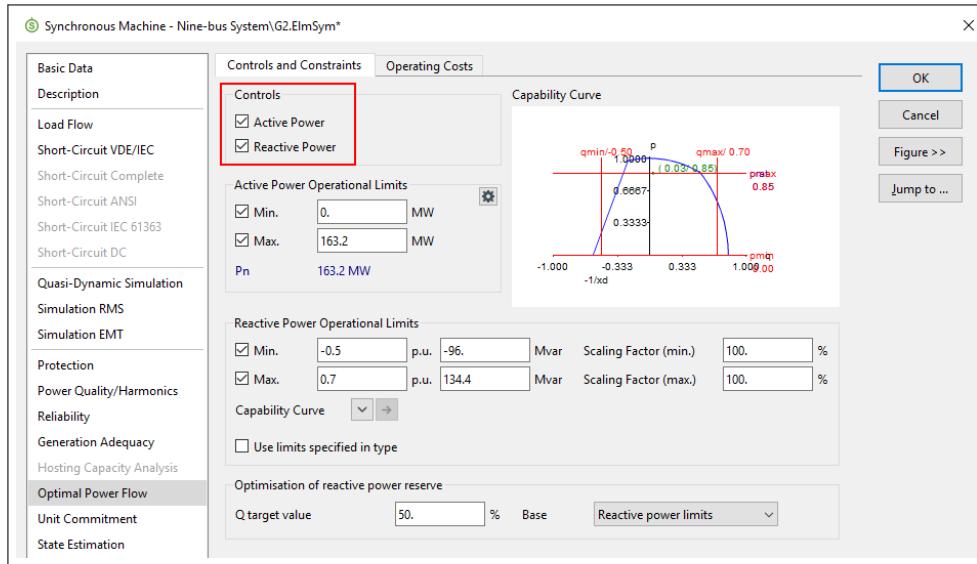


Figure 38.2.2: Active and Reactive Power Controls of a Synchronous Machine (*ElmSym*)

## 2. 2- 3- and 4-Winding Transformers

If a transformer has the *Tap Position* option selected, the user can further select the associated *Control Mode* to be used. This determines whether the tap position will be treated as a continuous or a discrete control parameter in OPF. Note that 3- and 4-winding transformers have up to three/four tap changers which may individually be used as either continuous or discrete control parameters in OPF.

Figure 38.2.3 shows the *Controls* section of the dialog for a 3-winding transformer. It should be noted that the *Optimise* section with the selection of *Pre- and post-fault position* or *Only pre-fault position* are only considered by the DC OPF.

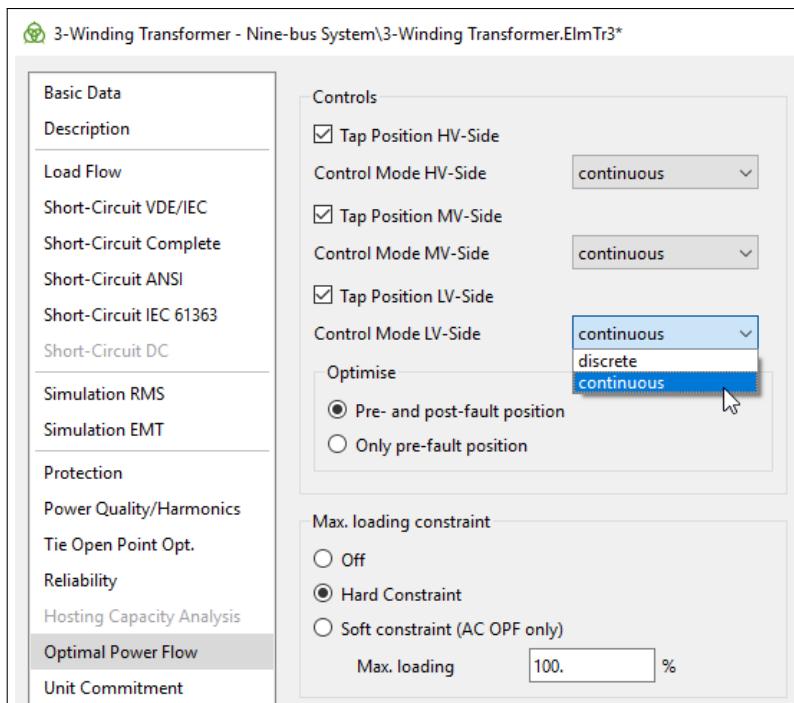


Figure 38.2.3: Tap Position Control for a 3-Winding Transformer

### 3. Shunts

In a similar fashion to transformers, the number of steps for a shunt may serve as either a continuous or a discrete optimisation parameter.

### 4. Static Var Systems

The control variable of a static var system enters the optimisation process as a continuous variable.

### 5. PWM Converter

The power over an HVDC link can be controlled in the PWM Converter similar to a generator in the OPF. Therefore it is included in the generator active and reactive power control category. If the control shall be disabled for PWM Converts this has to be done on the individual element.

#### 38.2.1.3 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*, *ElmVsmono*):
  - 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
- 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 nominal 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 15.3: Boundaries.

#### 38.2.1.4 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  $f$  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.

### 38.2.1.5 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 inclusion of the following variables in the *Flexible Data* tab (for synchronous machines and grids) is suggested. The Variable Set must be set to 'Calculation Parameter' as indicated below, and the actual variable names are given in parentheses.

Synchronous machines:

- Active Power (c:P:bus1)
- Reactive Power (c:Q:bus1)
- Apparent Power (c:S:bus1)
- Voltage Magnitude (c:u:bus1)

Grids:

- Total Production Cost, including costs through external grids (c:cst\_disp). It should be noted that the production costs are expressed in the same units utilised in the production cost tables of the individual generator elements.
- Active Power Losses (c:LossP)
- Reactive Power Losses (c:LossQ)
- Active Power Generation (c:GenP)
- Reactive Power Generation (c:GenQ)

In addition to these results, the complete set of variables from conventional load flow calculations is available. For further information on defining Flexible Data in *PowerFactory*, refer to Chapter 10: Data Manager, Section 10.6.

A text report is also available and can be generated by clicking on the *Output Calculation Analysis* icon on the main toolbar. This offers various templates for detailed result documentation.

### 38.2.2 AC Optimisation - Initialisation

The non-linear optimisation requires initialisation to generate an initial starting condition. The *Iteration* page of the OPF dialog allows the user to select the initialisation method.

**Load Flow**

Displays the load flow command which is used for initialisation in the case that no flat start initialisation is used.

**Initialise by flat-start**

The user may choose whether the initialisation is performed by a load flow calculation or by a flat start. If it is known in advance that the final solution of the optimisation is close to a valid load flow solution, initialisation using a load flow calculation results in faster convergence.

**No flat initialisation (use load flow result)**

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 second page 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.

### 38.2.3 AC Optimisation - Advanced Options

**Penalty Weighting Factor**

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.

**Soft constraints****Penalty factor for soft constraints**

Initial value of the penalty factor for soft constraints is 1000.

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 *Basic Options* page all the limits of a chosen class are interpreted as soft constraints if the option *Enforce soft ...* is selected.

### 38.2.4 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:

$$L(\vec{x}, \vec{s}, \vec{\lambda}) = f(\vec{x}) - \mu \cdot \sum_i \log(s_i) + \vec{\lambda}^T \cdot [g(\vec{x}) + h(\vec{x}) + s] \quad (38.1)$$

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 Criteria

#### ***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 %).

#### ***Max. Change of Objective Function***

Used when *Convergence of Objective Function* option *values of objective function become constant* is selected. The user enters a value (in %), below which the Lagrangian is considered to have converged.

#### ***Max. Value for Gradient of Objective Function***

Used when *Convergence of Objective Function* option *gradient of objective function converges to zero* is selected. The user enters an absolute value, below which the Lagrangian is considered to have converged.

#### ***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.

**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.

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.

### 38.2.5 AC Optimisation - Output

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. Two options are available to select the level of detail contained in output messages. These options are available in the *Output* page of the OPF dialog and are described below.

#### 38.2.5.1 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 variable names are shown parenthesised in italics):

- The current error of the constraint nodal equations (in VA) (*Err.Nodes*);
- The current error of the constraint model equations (*Err.ModelEqu*);
- The current error of the inequality constraints (*eInequ*);
- The current value of the gradient of the Lagrangian function (*gradLagFunc*);
- The current value of the Lagrangian function (*LagFunc*);
- The current value of the objective function **f** to be minimised (*ObjFunc*);
- The current value of the penalty function **fpen** (*PenFunc*);
- The current values of the relaxation factors (Rlx1, Rlx2) for the primal and dual variables;
- The current value of the penalty factor  $\mu$  (*PenFac*).

#### 38.2.5.2 Show max. nodal and model equation error elements

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).

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.

### 38.2.5.3 Show marginal costs of constraints

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.

Figure 38.2.4 shows an example of the output printed when the option *Show marginal costs of constraints* is selected and the *Number of reported cost factors per iteration* is set to 2. It indicates that the upper voltage limit of terminal “23 BUS 23” is being the most restrictive one and keeps the algorithm from achieving even better results.

Iteration: 15		
Marginal costs of inequality constraints(Iter: 15)		
Network element	Cost factor	Constraint
sym_14_SC	4.441787e+00	Max.Active Power Limit
13 BUS 13	2.657815e+00	Upper Voltage Limit

Iteration: 16		
Marginal costs of inequality constraints(Iter: 16)		
Network element	Cost factor	Constraint
sym_14_SC	3.680603e+00	Max.Active Power Limit
sym_15_6	2.183717e+00	Max.Active Power Limit

Iteration: 17		
Marginal costs of inequality constraints(Iter: 17)		
Network element	Cost factor	Constraint
sym_14_SC	3.377744e+00	Max.Active Power Limit
sym_15_6	2.031379e+00	Max.Active Power Limit

Figure 38.2.4: Example output when *Show marginal costs of constraints* is selected

## 38.3 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). *PowerFactory* uses a standard LP-solver (based on the simplex method and a branch-and-bound algorithm) which ascertains whether the solution is feasible. 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. The DC load flow solution includes the following calculated parameters (parameter names are given in italics):

- For terminals:
  - Voltage Angle ( $\phi_{iu}$  [deg])
  - Voltage Magnitude ( $u$  [p.u.]; assumed to be 1.0 p.u. in DC calculation)

- Voltage Magnitude ( $upc$  [%]; assumed to be 100 % in DC calculation)
- Line-Ground Voltage Magnitude ( $U$  [kV])
- Line-Line Voltage Magnitude ( $U1$  [kV])
- For branches:
  - Active Power Flow ( $P$  [MW])
  - Active Power Losses ( $Ploss$  [MW]; assumed to be 0 MW in DC calculation)
  - Reactive Power Flow ( $Q$  [Mvar]; assumed to be 0 Mvar in DC calculation)
  - Reactive Power Losses ( $Qloss$  [Mvar]; assumed to be 0 Mvar in DC calculation)
  - Loading ( $loading$  [%]; Loading with respect to continuous rating)

The following parameters are calculated in addition to the results found by the DC load flow:

- For generators:
  - $c:avgCosts$   
The fixed cost factor [\$/MWh] used in the objective function (i.e. average cost considering the costs at the generator's active power limits).
  - $c:Pdisp$   
Optimal power dispatch for generator.
  - $c:cst\_disp$   
Production costs in optimal solution:  
 $cst\_disp = costs * Pdisp$
- For Transformers:
  - $c:nntap$   
Optimal tap position.
- For loads:
  - $c:Pdisp$   
Optimal load shedding for load.

### 38.3.1 DC Optimisation - Basic Options

#### 38.3.1.1 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).

##### Min. Generator Dispatch Change

Minimises the change in generator dispatch from the generators' initial value.

### 38.3.1.2 Controls

The basic role of each control for the DC Optimisation method is as described for the AC optimisation method in section [38.2.1.2](#).

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.

### 38.3.1.3 Constraints

The three constraints are as described for the AC optimisation method in section [38.2.1.3](#).

For DC optimisation the following constraint is 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.

#### Handling

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.

*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 constraints can be chosen on an individual basis per boundary element (*ElmBoundary*).

### 38.3.1.4 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. The shadow prices are available as results for the *PowerFactory* elements listed below (result variable names are given followed by their corresponding unit). These result variable names are available as *Calculation Parameters* when defining variable sets for each element. For more information on defining variable sets, refer to Chapter 13: Study Cases, Section 19.3 (Variable Sets).

- Line (*ElmLne*):
  - *ShadowPrice* in \$/MWh (Shadow price)
- 2-Winding Transformer (*ElmTr2*):
  - *ShadowPrice* in \$/MWh Shadow price (loading constraint)
  - *ShadTapMax* in \$/MWh Shadow price (Maximum Tap constraint)
  - *ShadTapMin* in \$/MWh Shadow price (Minimum Tap constraint)
- 3-Winding Transformer (*ElmTr3*):
  - *ShadowPrice* in \$/MWh (Shadow price (loading constraint))
  - *ShadTapMaxLV* in \$/MWh (Shadow price (Maximum Tap constraint (LV)))
  - *ShadTapMinLV* in \$/MWh (Shadow price (Minimum Tap constraint (LV)))
  - *ShadTapMaxMV* in \$/MWh (Shadow price (Maximum Tap constraint (MV)))
  - *ShadTapMinMV* in \$/MWh (Shadow price (Minimum Tap constraint (MV)))
  - *ShadTapMaxHV* in \$/MWh (Shadow price (Maximum Tap constraint (HV)))
  - *ShadTapMinHV* in \$/MWh (Shadow price (Minimum Tap constraint (HV)))
- Boundary (*ElmBoundary*):
  - *ShadowMaxP* in \$/MWh (Shadow price (max. total active power constraint))
  - *ShadowMinP* in \$/MWh (Shadow price (min. total active power constraint))
- Synchronous Machine (*ElmSym*):
  - *ShadowMaxP* in \$/MWh (Shadow price (upper limit active power))
  - *ShadowMinP* in \$/MWh (Shadow price (lower limit active power))
- External Grid (*ElmXnet*):
  - *ShadowMaxP* in \$/MWh (Shadow price (upper limit active power))
  - *ShadowMinP* in \$/MWh (Shadow price (lower limit active power))
- General Load (*ElmLod*):
  - *ShadowMaxP* in \$/MWh (Shadow price (max. load shedding))
  - *ShadowMinP* in \$/MWh (Shadow price (min. load shedding))

### 38.3.2 DC Optimisation - Initialisation

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).

### 38.3.3 DC Optimisation - Advanced Options

#### Load Shedding Options

If *Allow Load Shedding* is among the selected Controls (see Section 38.3.1: Basic Options) on the *Basic Options* tab, an additional term will be added to the objective function. The weight of this term can be controlled using the *Penalty Factor* in the *Load Shedding Options* section of the OPF dialog.

The following term will be added to the objective function, where  $\omega$  is the specified *Penalty Factor*, and  $c$  is the cost factor of load  $i$ :

$$\omega \sum_{j=1}^{n_{Co}} \sum_{i=1}^{n_{Load}} c_i |Load_i^j - Load_i^{curr}| \quad (38.2)$$

#### Transformer Tap Deviation Control

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.

##### **Use Penalty Factor for Tap Deviation**

If enabled, the following additional term is added to the objective function:

$$\omega \sum_{i=1}^{n_{Tr}} |tap_i^0 - tap_i^{curr}| \quad (38.3)$$

##### **Penalty Factor**

Specifies the weighting factor for the additional objective function term above.

#### 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.

#### Additional Settings

##### **Check for Constraint Violations after Optimisation**

If this option is selected the violated constraints are written to the output window after the OPF.

##### **Use Presolve procedure**

If selected, the LP is checked for linear dependencies of constraints. They will be eliminated and only the corresponding (smaller) system is solved.

### 38.3.4 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).

##### Max. Acceptable Error for Constraints

Maximum relative error (in %) by which a constraint can be violated while still being considered a feasible solution.

It should be noted that when *Max. Number of Iterations* is set to '1', the LP is solved without outer loops.

#### 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.

## 38.4 Contingency Constrained DC Optimisation (LP Method)

The Contingency Constrained DC Optimisation performs an OPF using DC optimisation (as described in Section 38.3: 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 offers the following result variables (as well as those provided by the DC load flow, as described in Section 38.3: DC Optimisation (Linear Programming)):

- For generators:
  - *c:Pdisp*  
Optimal generation for each contingency case. The optimum generation for each contingency case is stored as a parameter event object in the corresponding contingency object (*ComOutage*). Thus, each contingency object will hold parameter events for each selected generator (the name of the parameter event is the name of the generator). The parameter event reflects the optimal generation for that generator in the given contingency case.
- For Transformers:
  - *c:nntap*  
Optimal tap positions for each contingency case. The optimum tap positions for each contingency case are stored as a parameter event object in the corresponding contingency case object (*ComOutage*). Thus, each contingency object (*ComOutage*) will hold parameter events for each selected transformer (the name of the parameter event is the name of the transformer). The parameter event reflects the optimal tap position for that transformer in the given contingency case
  - *c:mxTpChng (\_l,\_m, \_h)*  
*mxTapChng* is the maximum tap change deviation between the optimal base case tap position and the optimal tap position considering all contingencies. For 3-winding transformers, HV-, MV- and LV-side tap changes are calculated individually.
- For loads:
  - *c:Pdisp*  
Optimal load shedding for each contingency case. The optimum load shedding for each contingency case is stored as a parameter event object in the corresponding contingency case object (*ComOutage*). Thus, each contingency object will hold parameter events for each selected load (the name of the parameter event is the name of the load). The parameter event reflects the optimal load shedding for that load in the given contingency case.

### 38.4.1 Contingency Constrained DC Optimisation - Basic Options

#### 38.4.1.1 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.

### 38.4.1.2 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 38.3.1: 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.

### 38.4.1.3 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 38.2.1 (Basic Options).

The user can select from the following control variables:

- **Generator Active Power Dispatch (*ElmSym*, *ElmXnet*)**

#### **Dispatch in Contingencies**

- **Use base case dispatch:** for all contingency cases, use the generator dispatch from the base case. Using this setting, a single control variable is introduced to the system for each selected generator. The total number of generator controls in this case equals the number of selected generators and/or external grids.
- **Allow different dispatch:** for each contingency case, allow a generator dispatch different to that used in the base case. Using this setting, for each selected generator, a control variable is introduced for the base case and for each contingency case. This option must be selected from the drop-down box when the objective function *Min. Generator Dispatch Change (Pre-to-Postfault)* has been selected. The total number of generator controls in this case equals: (number of selected generators) \* (1 + number of selected contingencies)

- **Transformer Tap Positions (*ElmTr2*, *ElmTr3*, *ElmTr4*)**

#### **Tap Positions in Contingencies**

- **Use base case tap positions:** for all contingency cases, use the transformer tap positions from the base case. Using this setting, a single control variable is introduced to the system for each selected transformer. The total number of tap controls in this case equals the number of selected transformers.
- **Allow different tap positions:** for each contingency case, allow tap positions different to those used in the base case. Using this setting, for each selected transformer, a control variable is introduced for the base case and for each contingency case. This option must be selected from the drop-down box when the objective function *Min. Transformer Tap Change (Pre-to-Postfault)* has been selected. The total number of tap controls in this case equals: (number of selected transformers) \* (1 + number of selected contingencies)

- **Allow Load Shedding (*ElmLod*)**

A separate control variable is introduced to the system for the base case and for each contingency case. This control variable can be selected in conjunction with any objective function. The total number of load controls equals: (number of selected loads)\*(1 + number of selected contingencies)

#### 38.4.1.4 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:  $2^*(\text{number of contingencies})$ .

In addition to the constraints provided for DC optimisation (for further information see Section 38.3.1: Basic Options), 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. This leads to two constraints in LP formulation for the base case tap position calculation, and to:  $2 \times (1 + \text{number of contingencies})$  constraints for contingency case calculations.

##### **Handling**

Active power dispatch constraints can be chosen on an individual basis (via a checkbox) per generator.

Tap position constraints will be implicitly imposed whenever the corresponding tap is a designated control variable. The tap position limits are defined in the transformer's assigned *Type*.

*Loading* constraints can be chosen on an individual basis (via a checkbox) per line element (*ElmLne*) and per transformer element (*ElmTr2*, *ElmTr3*). Once a loading constraint for a specific line or transformer is imposed, it will be considered by all contingencies contained in the contingency list. 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 constraints can be chosen on an individual basis per boundary (*ElmBoundary*). Once a boundary constraint for either the maximum total active power limit or minimum total active power limit is imposed, it will be considered by all contingencies in the contingency list.

The list of contingencies to be considered by the OPF is selected by choosing a specific contingency analysis command (parameter *Contingency Analysis* in the OPF dialog, *Basic Options* tab), which contains in its folder the contingency objects (*ComOutage*) to be considered.

#### 38.4.2 Contingency Constrained DC Optimisation - Initialisation

As described for DC optimisation. Refer to Section 38.3.2 (Initialisation).

#### 38.4.3 Contingency Constrained DC Optimisation - Advanced Options

As described for DC optimisation. Refer to Section 38.3.3 (Advanced Options).

### 38.4.4 Contingency Constrained DC Optimisation - Iteration Control

As described for DC optimisation. Refer to Section 38.3.4 (Iteration Control).

### 38.4.5 Contingency Constrained DC Optimisation - 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, the *Output of Results* command button  on the main toolbar becomes active. This command allows the printing of various reports. 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.

## 38.5 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 Optimal Power Flow calculation.

### 38.5.1 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 option *no flat initialisation* 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*.

### 38.5.2 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 quite unlikely 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 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.

### 38.5.3 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.

### 38.5.4 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. Don't 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 39

# Unit Commitment and Dispatch Optimisation

## 39.1 Introduction

The **Unit Commitment and Dispatch Optimisation**  allows the user to complement the traditional network simulation with a market simulation, 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 such to minimise overall operating costs. The module combines functionalities of a 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.

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.

## 39.2 Application Cases for the Unit Commitment

The Unit Commitment and Dispatch optimisation module can be used in various set ups 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.

### 39.2.1 Full Unit Commitment

The main application is to execute a full Unit Commitment for a network. A Unit Commitment is optimising various control variables in an electrical network, like 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 time and ramp limits. A Unit Commitment is considering 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 load shedding. Each control variable has assigned costs as base for the optimisation. In *PowerFactory* the Unit Commitment can directly provide a full redispatch solution since the network constraints can be directly regarded for the base case and selected contingency cases in each time step. Usually all costs in the system are minimised while regarding all network constraints directly in one execution of the command. If the Unit Commitment shall be independent from the previous dispatch the additional redispatch costs should not be regarded.

### 39.2.2 Market Simulation

The Unit Commitment and Redispatch Optimisation module can also be used to execute a market simulation without considering any network constraints. Thus, the resulting dispatch represents the solution of a free, unconstrained market in which the cheapest units are generating the needed power independent from the feed-in point. A market simulation may be needed to determine the power plant schedule of the unregulated market and derive the needed redispatch in a second iteration.

The constraints of the generating units like ramps and minimum downtimes can optionally be regarded 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 regarded.

### 39.2.3 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 like line loadings with and without considering contingencies. The results from the market simulation can also be imported to *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.

## 39.3 Unit Commitment Command

### 39.3.1 Basic Options

#### 39.3.1.1 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.

### 39.3.1.2 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

In the Algorithm settings (see [39.3.6.3](#)) a contingency filter can be defined to restrict the consideration of the individual constraints to the most relevant ones.

If the **Use linearised calculation** is selected the contingency analysis will execute feasible contingencies with a fast linearisation method (for AC or DC).

### 39.3.1.3 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.

## 39.3.2 Objective Function

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 overwrites the settings in the corresponding network elements. The available function components are:

- **Minimise generator operating costs:** The generator operating cost 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 additional generator 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 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 control variable deviations of transformers, shunts and HVDCs:** The objective function participation tries to keep taps and HVDC-controls as close as possible to the values prior to the optimisation, i.e., the Load Flow tap/control. Costs for deviations are defined within the individual elements, which are ElmTr2, ElmTr3, ElmTr4, ElmVoltreg, ElmShnt, ElmVsc, ElmVsclmono. When minimising the taps/shunt controls or HVDC-controls it can mean keeping the positions 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 objective function) or as close as possible to the optimised values of the previous timestep.

### 39.3.3 Controls/Constraints

#### 39.3.3.1 Controls

The Unit Commitment and Dispatch Optimisation needs controls to optimise the objective function. A control variable is only selected if it is specified in the network element and globally in the Unit Commitment / Dispatch Optimisation command. The supported controls are:

- **Active power dispatch:** Synchronous generators (ElmSym), external grids (ElmXnet), static generators (ElmGenstat), PV-systems (ElmPvsys) or doublyfed induction machines (certain ElmAsm models) may contribute such a control variable. HVDCs (ElmVsc, ElmVsclmonos) 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 always 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 elements itself.
- **Load shedding:** Loads can be shed to eliminate overloadings, 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 (ElmPvsys) or doublyfed induction machines (certain ElmAsm models), HVDCs (ElmVsc, ElmVsclmonos) may contribute such a control variable.
- **Switchable shunts (only AC):** This control may assist when it comes to voltage constraints / stability. The general representation as continuous or discrete controls in the Unit Commitment simulation is configured on the advanced page of the Unit Commitment / Dispatch Optimisation command. Please note that this only applies to controls that are selected to be discrete controls in the elements itself.

On the “Advanced” tab there are additional setting available regarding the controls:

- **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.
- **Allow curtailment of variable renewable energy sources:** Generators marked as VRE can be shed to eliminate overloading, 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.
- **HVDC active/reactive power controls** are important for the redirection of power flows in order to reduce overloading. It can be changed when active or reactive power controls are selected.
- **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.

### 39.3.3.2 Constraints

Constraints can be defined in the network elements and selected for consideration in the Unit Commitment dialog. A constraint is only 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.

- **Control variable constraints** are defined in the network elements as limits of the control variables. It can be selected whether the limits should be considered or not via list boxes on the right side of the control variables. The control variable constraints are always hard constraints if enabled.
  - Active power limits
  - Transformer tap positions limits
  - Load shedding limits
  - Reactive Power limits (only AC)
  - Shunt tap positions limits (only AC)
- **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.
  - **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 [39.6](#)).

The “Advanced” constraint options are:

- **Enforce control variable power balance:** This option will force the unit commitment to do a redispatch only to keep the power balance 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.

- **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 over-loaded transformer that is supplying a network area with no controls and a fixed power demand.

### 39.3.4 Results/Output

The Unit Commitment and Dispatch Optimisation simulation writes three result files during execution. The main result file is the “Unit Commitment (summary)” where the overall results simulation are stored. 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 load flow results for each time step after the adaptation of the control variables.

The available settings on this page are:

**Record all optimal control variables:** Should assist the user and simply records all optimisation control variables in the optimal state. If the checkmark **Record redispatch costs per time step** is selected the redispatch results, such as redispatch costs, are additionally recorded for every timepoint.

**Calculate and record load flow results with optimal controls:** The optimal controls are calculated by the solver. The user can select whether these controls are simply stored in the result file with a fast timesweep or whether a real second load flow sweep should be calculated verifying the optimal solution. With the latter option, we can also verify and record transmission constraints of the grid. This can be done by enabling the checkbox **Record all constraint result variables**.

**Result file Time sweep results (after optimisation):** Optimal results are recorded in this result file. One row corresponds to one time step. The result file is dense and similar to the one of the Quasi-Dynamic Simulation

**Result file Summary results:** Similar to the Probabilistic Analysis, summary quantities of control variables can be recorded here, without being defined in the large time sweep result file. This result file will thus contain only one row. Variables are not configurable by the user but are implicitly given by the controls during the optimisation. The results are used for reports later on.

**Contingency results:** Similar to the contingency time sweep calculation, a list of sparse sub-result files will hold the contingency results when optimised controls are applied. Each such sub-result file has contingency-results as rows.

**Result file Time sweep results (before optimisation):** The load flow results before the optimisation are recorded in this result file. At least the original status of all control variables is recorded automatically (needed for optimized results and creating reports). The original status of the constraints can also be recorded automatically if the corresponding checkbox is ticked. One row corresponds to one time step. The result file is dense and similar to the Quasi-Dynamic Simulation result file. Additional variables to be recorded can be defined by the user.

**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.

### 39.3.5 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.

### 39.3.6 Algorithm

#### 39.3.6.1 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.

**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.

**Costs for soft constraint violations:** Soft constraints are constraints that need not necessarily be satisfied by the optimiser. An additional positive variable is added to the LP for such constraints and penalise its size with this coefficient in the objective function. Whether a constraint is soft or hard has to be defined in the network elements on the Unit Commitment page. If a constraint is marked as soft, it will also be marked soft in all its contingencies.

#### 39.3.6.2 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 usage of the external solvers needs an additional *PowerFactory* license module called “Unit Commitment and Dispatch Optimisation Interface” as well as licenses of the external solvers. 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 “Unit Commitment and Dispatch Optimisation Interface” licence also allows the export of the MILP problem as an AMPL-file (.nl) so that the file can be solved by a user specific solver.

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*.

**User defined parameters** offers a table with solver input parameters like the relative optimality gap (*mipGapRel*) and the absolute optimality gap (*mipGapAbs*). The table specific for each solver and the default of each parameter is shown if the specific parameter is disabled. If enabled the parameters can be modified and the initial value may vary from the default solver value.

### 39.3.6.3 Constraint Filter

**Constraint 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 on the constraint to be overloaded by the algorithm. This setting allows the algorithm to arrange the constraints into general, lazy and low priority depending on a specified margin. They are then incorporated differently into the algorithm. This is done to improve memory usage and performance. The usage/application can be adapted to be for loading constraints only, for boundary constraints only or for both. Please note that, if the filter is active, constraints above the callback margin will not be added to the optimisation. However, if they are violated after the optimisation, they will be reported when verify optimal solution is selected. Also note that the values for general/lazy/low priority should be increasing.

**Constraints filter for voltage constraints:** When voltage constraints are formulated the margin to the max/min allowed values before the optimisation is calculated. This value indicates the likelihood on the constraint to be overloaded by the algorithm. This setting allows the algorithm to arrange the constraints into general, lazy and low priority depending on a specified margin. They are incorporated differently into the algorithm. Please note that, if the filter is active, constraints above the callback margin will not be added to the optimisation. However, if they are violated after the optimisation, they will be reported when verify optimal solution is selected. This is done to improve memory usage and performance. Also note that the values for general/lazy/low priority should be increasing.

**Max number of contingency constraints per element:** If either of the constraint filters is activated and contingencies are selected, the maximum number of contingency constraints which are formulated for grid elements can be limited. The criterion is based on the effect of a contingency on the element and the margin.

### 39.3.6.4 Effectiveness

**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 new calculation is always needed when the topology of the network is changing. Optional this can be triggered after a fixed number of calculation steps.

**Effectiveness thresholds:** When loading/boundary/voltage constraints are formulated, the effectiveness of generators, transformers, loads, HVDCs and shunts is calculated in order to estimate the constraint values with varying controls. These thresholds provide the minimum effect to consider such a sensitivity.

## 39.4 Handling of results

### 39.4.1 Reports

The toolbar of the Unit Commitment and Dispatch Optimisation includes a Report command for the Unit Commitment and Dispatch Optimisation . The reports available are:

- **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.
- **Unit Commitment optimal solution report** This report shows the resulting values for the control variables and the according costs for each participating network element in the Unit Commitment.
- **Non-Convergent Load Flow Cases** This report lists the points in time where the load flow is not converging before and after the optimisation.
- **Quasi-Dynamic Simulation Report: Loading Ranges** This report displays the elements loaded over a threshold within a specified time range.
- **Quasi-Dynamic Simulation Report: Voltage Ranges** This report displays the terminals violating a defined voltage band within a specified time range.

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.

### 39.4.2 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.

### 39.4.3 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 redispatch cost or the 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.

### 39.4.4 Load Unit Commitment Results

The “Load Time Sweep Results” functionality  allows to reload the results of a specific time step from the result file to the network elements. This enables the user to investigate critical hours in the network model.

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** functionality reloads the results of a specific point in time from the result file to the network elements. This enables the user to investigate the results of critical hours in the network model in an efficient way without executing a load flow calculation.

The **Assign time characteristics with optimal controls** functionality enables the user to write back the resulting optimal controls from the result file to characteristics in a new study case 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 do 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.

### 39.4.5 Flexible Data Page

There are two additional sets of statistics variables available for certain elements after the execution of the Unit Commitment and Dispatch Optimisation in the variable selection on the flexible data page.

- **Statistics: Currents, Voltages and Powers** This variable set contains statistical values for control variables like for example average, minimum, maximum and variance over the complete simulation time.
- **Statistics: Calculation Parameter** This variable set contains statistical values like redispatch costs and amounts or loadings.

The other pages like “Calculation Parameter” and “Currents, Voltages and Powers” 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.

## 39.5 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

### 39.5.1 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 overloadings in the system. The reactive power of generating units can not be curtailed.

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 to **Allow the 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 [39.6.1](#))

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.

## 39.5.2 Operating Costs

The **Operating costs** for the generating units and external grids can be entered on the operational cost tap 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, where these costs are in USD/MWh (The currency unit can be selected in the project settings / ElmPvsys is entered in USD/kWh). 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.

The **Generator Cost Curve** is described in [39.5.2.1](#)

The resulting cost curve from the entered operation costs has to be linearised in both cases for the market simulation. 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.

### 39.5.2.1 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 CO<sub>2</sub> 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 pu): The rated efficiency is based on the nominal 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 nominal 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:

$$VariableCosts_i = \left( \frac{FuelC + OtThC + EmisC * EmisInt_i}{Eff_i} + OtElC \right) * ActivePower_i \quad (39.1)$$

So the resulting online costs for each operating point i of the generating unit are:

$$OnlineCosts_i = FixC + VariableCosts_i \quad (39.2)$$

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 (39.5.2).

### 39.5.3 Redispatch Costs

**Additional redispatch cost** can only be entered if the generating unit is not a VRE. The upward and downward redispatch cost can be entered separately for the active power. The “Redispatch costs for reactive power change” are entered in USD/Mvarh and have no differentiation between capacitive or inductive reactive power. The additional redispatch costs penalise the deviation from the load flow values prior to the optimisation. If the generating unit is flagged as a VRE only the **Costs for curtailment** can be specified.

### 39.5.4 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.

### 39.5.5 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 pu/h. The start up and shut down ramp limit can be entered in MW or pu. 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.

## 39.6 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 (39.5) or with several generators via a virtual power plant (39.7). A storage model can only be linked to one generating unit or one virtual power plant. There are two 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**.

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 (Only for hydro)

The rolling horizon (39.3.6) functionality will cause some drawbacks in the optimisation of storage units, since the storage is then only optimised for the individual time range.

### 39.6.1 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 pu 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
- Use efficiency at nominal power
- Use efficiency at max. operational limit
- Use efficiency at min. operational limit

The recalculation is done exactly with the determined control values.

## 39.7 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 39.6). 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 [39.5](#)). 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 nominal power
- According to dispatched power
- According to generation shift keys

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. It is not possible to use the linearised method for the contingency calculation if using virtual power plants.

## 39.8 Other Network Elements

### 39.8.1 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. If discrete is selected, the actual algorithm depends on the setting of the Unit Commitment Advanced tap on the Controls/Constraints page cf. [39.3.3.1](#).

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.

### 39.8.2 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.

### 39.8.3 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.

### 39.8.4 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 pu and specified as hard or soft constraints.

### 39.8.5 Lines

The **maximum loading** of lines can be defined as a hard or soft constraint.

### 39.8.6 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.

### 39.8.7 HVDC Converters

The reactive power flow over HVDC branches can be controlled by the Unit Commitment. The minimum and maximum power flow limits can be set in the element (ElmVsc, ElmVscmono) and Penalty costs in USD/MVAh can be assigned.

## 39.9 Troubleshooting

### 39.9.1 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.

### 39.9.2 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.

### 39.9.3 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.

### 39.9.4 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.

### 39.9.5 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.

### 39.9.6 Rolling horizon and time dependencies

The rolling horizon (39.3.6) 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.

# Chapter 40

## Transmission Network Tools

### 40.1 Introduction

The chapter presents the transmission networks tools available in *PowerFactory*:

- PV Curves Calculation (section [40.2.1](#))
- PV Curves Plot (section [40.2.2](#))
- QV Curves Calculation (section [40.3.1](#))
- QV Curves Plot (section [40.3.2](#))
- PTDF Calculation (section [40.4](#))
- Transfer Capacity Analysis (section [40.5](#))

These tools are accessible as illustrated in Figure [40.1.1](#), and this chapter introduces each respective tool, providing general descriptions and details pertaining to the command dialogs.

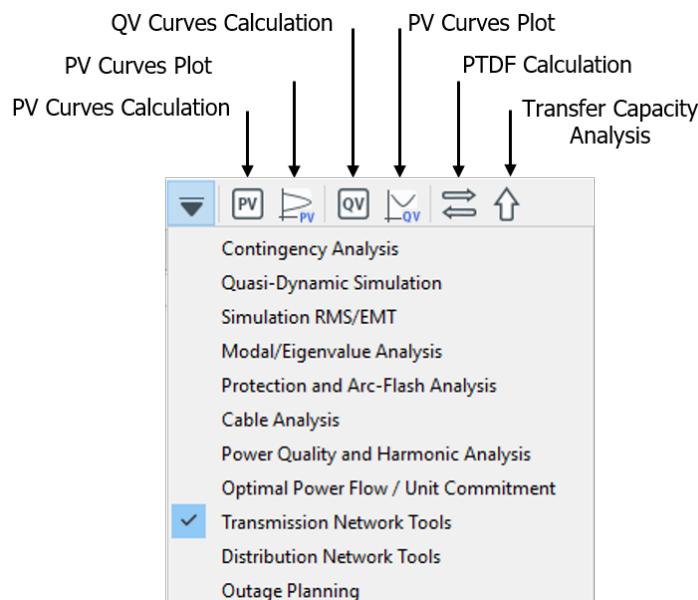


Figure 40.1.1: Accessing Transmission Network tools

## 40.2 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 40.2.1.

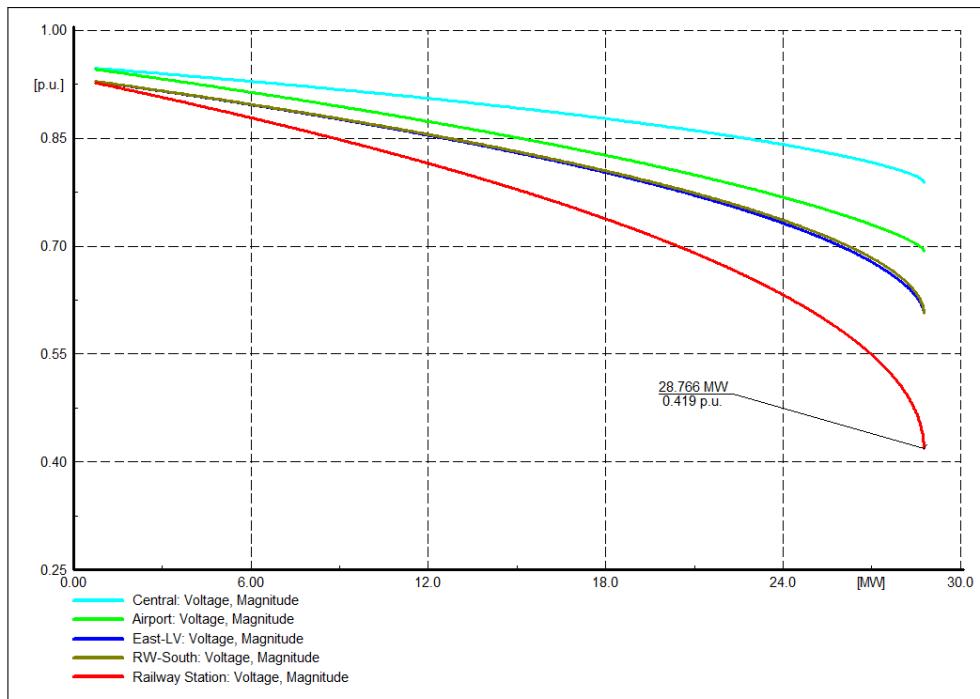


Figure 40.2.1: Network PV curve

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 40.2.1.
- PV Curves Plot (plots the PV curve); see Section 40.2.2.

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 *Calculate → PV Curves...*

### 40.2.1 PV Curves Calculation

The different pages options of the PV Curves Calculation command are explained below.

#### 40.2.1.1 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*. Zero- and negative loads are neglected.

### 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*).

## 40.2.1.2 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 [40.2.3.1](#).

## 40.2.2 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 [40.2.2.1](#) - Section [40.2.3.4](#).

### 40.2.2.1 Data input

The results file specified in, and created by, the PV Curves Calculation command (see Section [40.2.1](#)).

### 40.2.2.2 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.

### 40.2.2.3 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.

## 40.2.3 Outputs and Results

### 40.2.3.1 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.

### 40.2.3.2 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.

### 40.2.3.3 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.

### 40.2.3.4 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.

## 40.3 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 40.3.1.

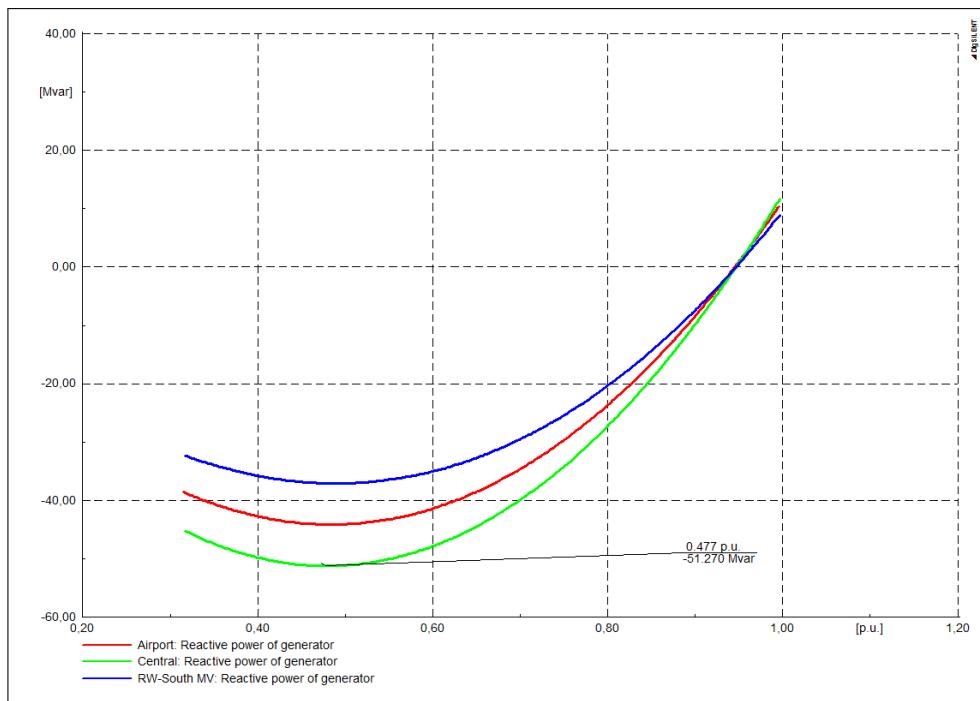


Figure 40.3.1: Network QV curves

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 40.3.1.
- QV Curves Plot  (plots the QV curve); see Section 40.3.2.

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 *Calculate → QV Curves...*.

## 40.3.1 QV Curves Calculation

The different pages options of the QV Curves Calculation command are explained below.

### 40.3.1.1 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.

### 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*).

#### 40.3.1.2 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.

#### 40.3.1.3 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.

#### 40.3.1.4 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.

#### 40.3.2 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.

##### 40.3.2.1 Basic Options

**Data Input:** the results file specified in, and created by, the QV Curves Calculation command (see Section 40.3.1).

**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.

##### 40.3.2.2 Active Power Injections

If several active power injections are calculated when executing the QV Curves Calculation command (see Section 40.3.1.3), it is possible to select between plotting all the active power injections or select only some cases.

##### 40.3.2.3 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.

## 40.4 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 [47](#).

## 40.4.1 Calculation Options

### 40.4.1.1 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

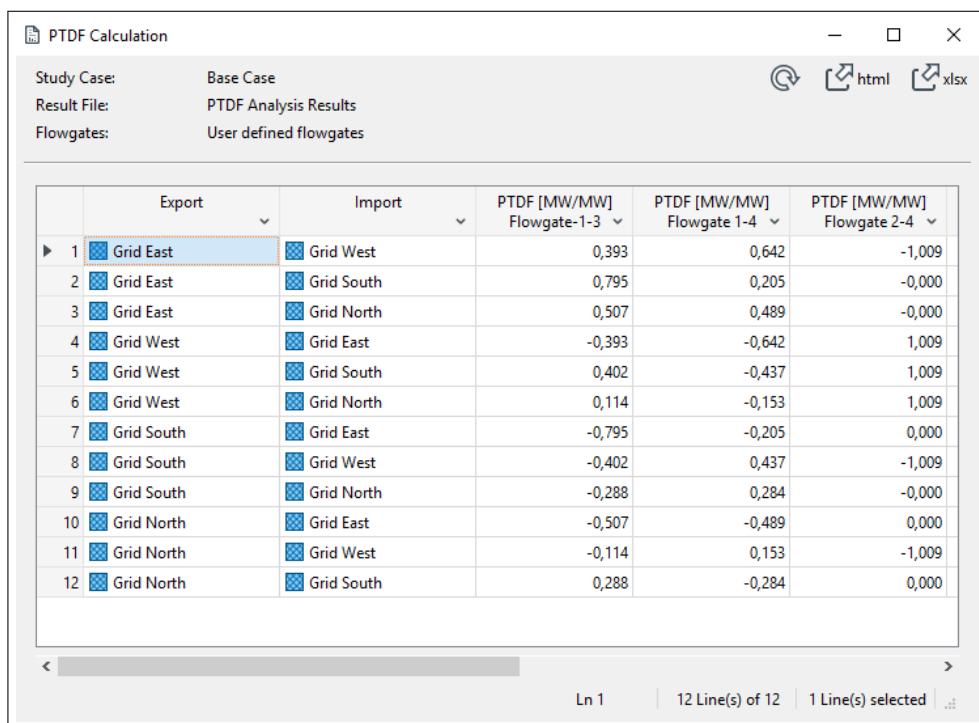
If generators are selected as scaling elements, the user can select between different scaling modes:

- *Nominal active power*: generation shift is proportionally scaled for the generators according to their nominal 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.

#### 40.4.1.2 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 40.4.1. 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". At the top, there are three tabs: "Study Case: Base Case", "Result File: PTDF Analysis Results", and "Flowgates: User defined flowgates". To the right of these tabs 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
► 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	0,489	-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	-0,489	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 buttons: '< >', 'Ln 1', '12 Line(s) of 12', '1 Line(s) selected', and '...'. There are also icons for 'html' and 'xlsx' at the top right of the table area.

Figure 40.4.1: 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 47).

#### 40.4.1.3 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.

## 40.5 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.

### 40.5.1 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.

#### Contingency Constrained

The analysis is carried out considering the contingencies defined in the Contingency Analysis command.

#### 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 (*ElmArea*) 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*.

#### Scaling elements - for Region-based calculation

The user may select between generators and loads. By default, all relevant elements are scaled, but on the *Scaling elements* tab, 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.

## 40.5.2 Constraints

### Consider thermal constraints (branches)

If this option 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.

### Consider voltage limits (terminals)

If this option 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.

### 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. The user has the option to consider the active power limits for only certain objects, which are selected as a set.

### Only consider constraints of selected region(s)

With this option, a specific region, where the constraint will be verified, can be defined. Single or multiple areas, zones and grids are supported as regions.

## 40.5.3 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

The results of the Transfer Capacity Analysis are printed in the output window as shown in Figure 40.5.1. 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 <b>SG1</b> with active power set to zero is treated as the synchronous compensator and cannot be scaled.				
Generator dispatch status for generator <b>SG2</b> is set to fixed. It will be ignored.				
...from <b>Grid North</b> to <b>Grid West</b> .				
Consider active power limits is enabled.				
<hr/>				
Iteration	Exporting Region	Importing Region	Transfer	Iteration Status
Number	Generation (MW)	Generation (MW)	(MW)	
0	<b>Grid North</b>	<b>Grid West</b>		
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	1155,92	thermal violation
5	3925,26	1716,51	1111,96	thermal violation
6	3902,22	1739,55	1085,04	OK
7	3913,74	1728,03	1100,00	OK
8	3915,50	1722,27	1105,99	thermal violation
9	3916,42	1725,15	1103,00	thermal violation
10	3915,18	1726,59	1101,50	OK
<hr/>				
Generation in the <b>Grid West</b> region reached limiting value. Calculation process stopped.				
Minimum step size reached. Calculation process stopped.				
The total transfer capacity (TTC) for the last feasible solution is 1101,50 (MW).				
<hr/>				
Transfer capacity analysis successfully calculated.				

Figure 40.5.1: Transfer Capacity Analysis output

#### 40.5.4 Iteration Control

In the *Initial Conditions* the *Initial scaling factor* of the scaling elements 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.

#### 40.5.5 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 40.5.2 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.

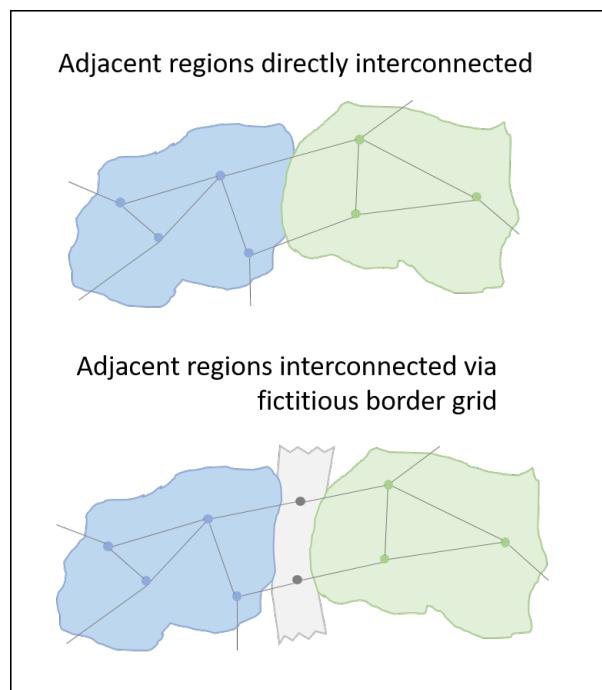


Figure 40.5.2: Network regions without and with fictitious border network

### Generators scaling mode

If generators are selected as scaling elements, the user can select between different scaling modes:

- *Nominal active power*: generation shift is proportionally scaled for the generators according to their nominal 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.

# Chapter 41

## Distribution Network Tools

### 41.1 Introduction

The chapter presents the *PowerFactory* tools for assessment and optimisation of distribution networks. The areas of analysis are highlighted in Figure 41.1.1. Each section of this chapter introduces the tool, presenting a general description, the objective function, the optimisation procedure, and the command dialogs.

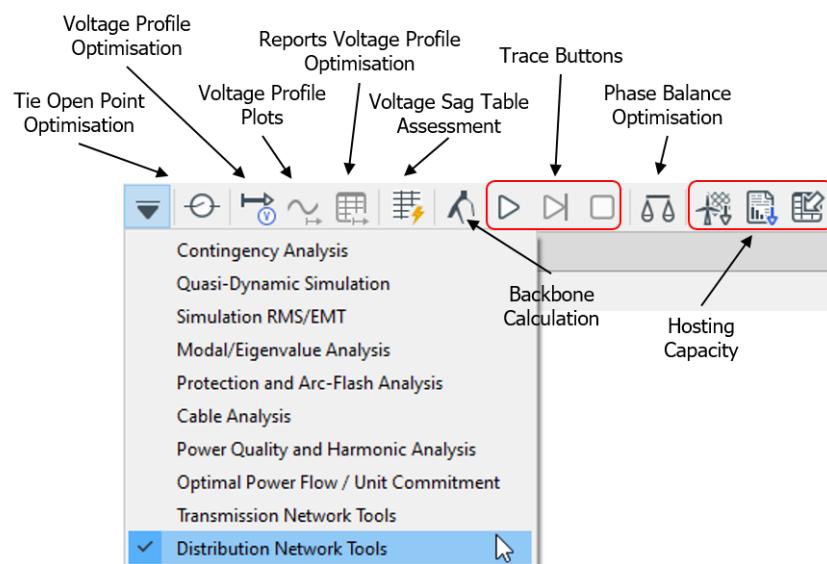


Figure 41.1.1: How to access the Distribution Network tools

### 41.2 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.

### 41.2.1 Calculation Options

#### 41.2.1.1 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.

#### 41.2.1.2 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.

### 41.2.2 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 *Calculate... → Voltage sag table...* option, or by initiating the command directly from the main 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 *Show → 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 *Define...* → *General Set* from the context-sensitive 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.

### 41.2.3 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 41.2.1.

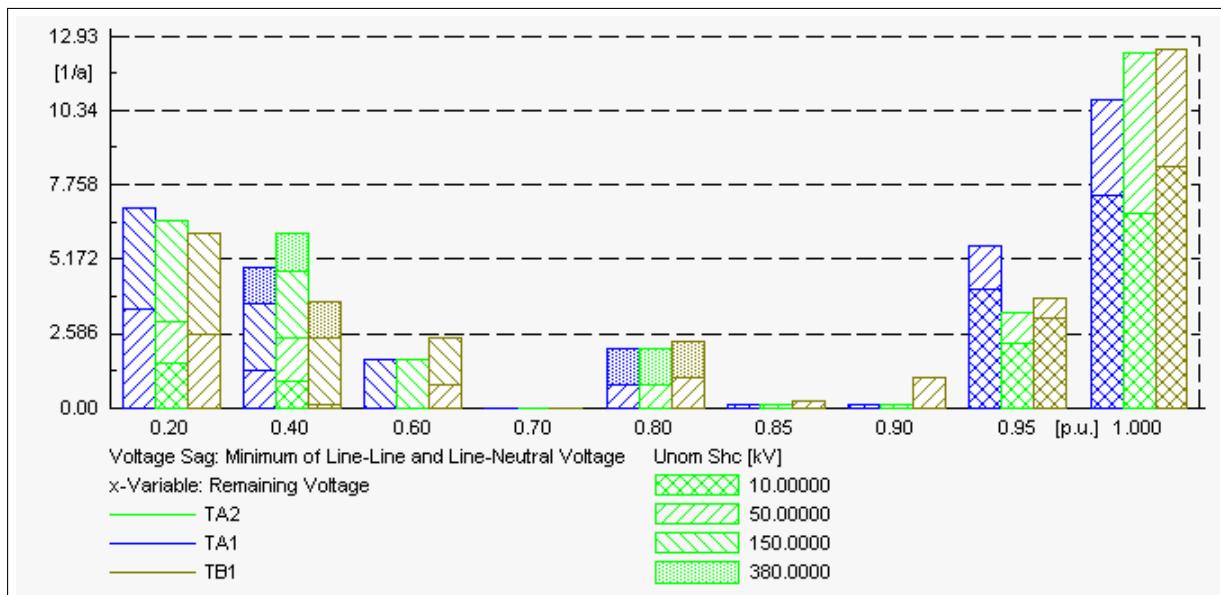


Figure 41.2.1: Example Voltage Sag Plot

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.

## 41.3 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 Tools, as shown in Figure 41.1.1.

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 41.3.1a.
- **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 41.3.1b.

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” tab of the *Load Flow Calculation*, but the *Load Flow Calculation* settings remain unchanged.

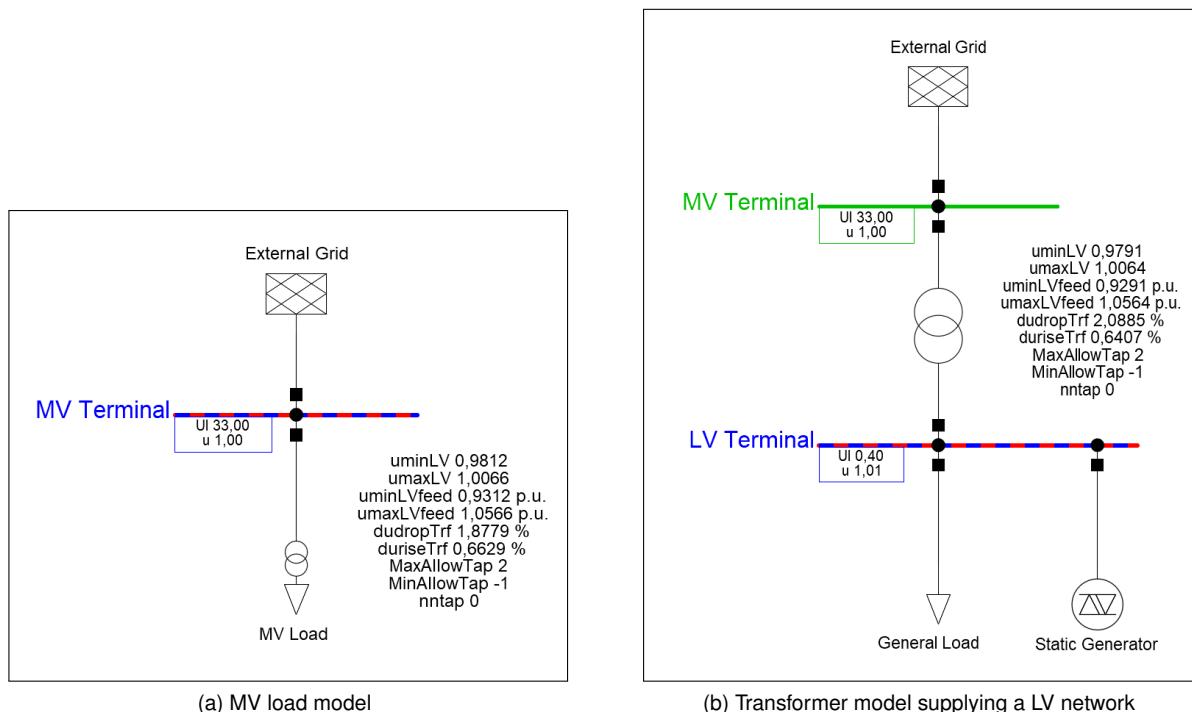


Figure 41.3.1: Use of different transformer representations for the Voltage Profile Optimisation

**Note:** In the MV load model, the transformer tap changer is represented on the LV side of the transformer.

### 41.3.1 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 41.3.2, 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.

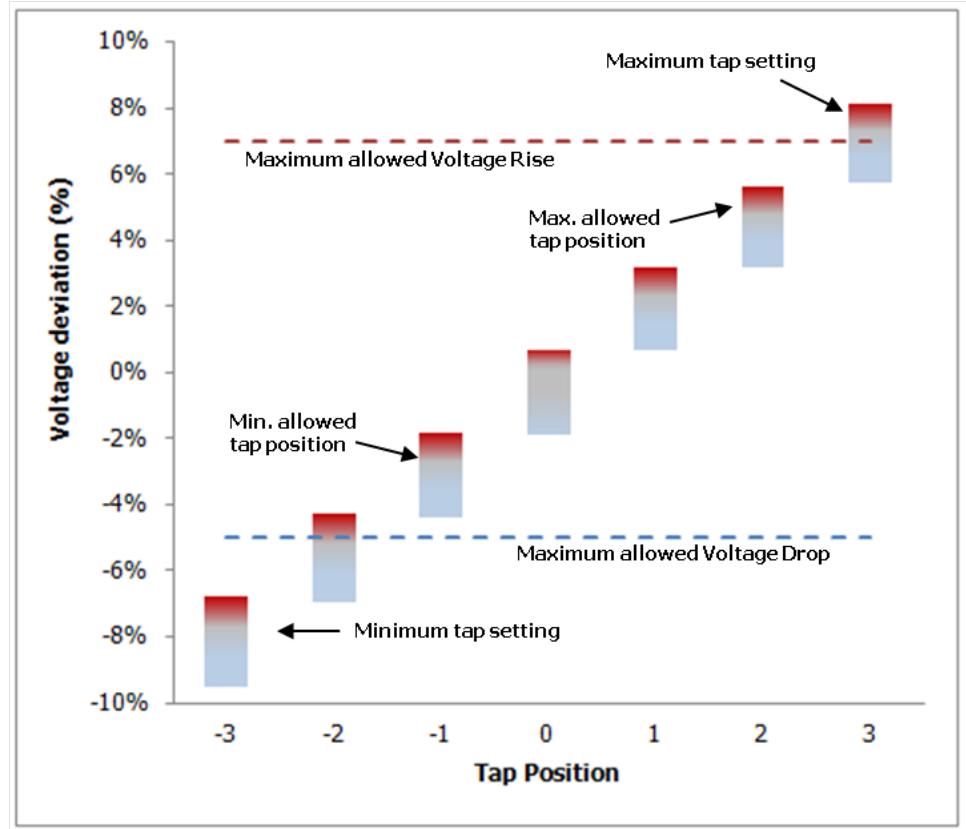


Figure 41.3.2: Distribution transformer tap limits

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 41.3.3, 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 41.3.3 shows an example for the objective function *Maximisation of generation*. The lower tap limit (position “0” in Figure 41.3.3) is selected in order to minimise the voltage rise.

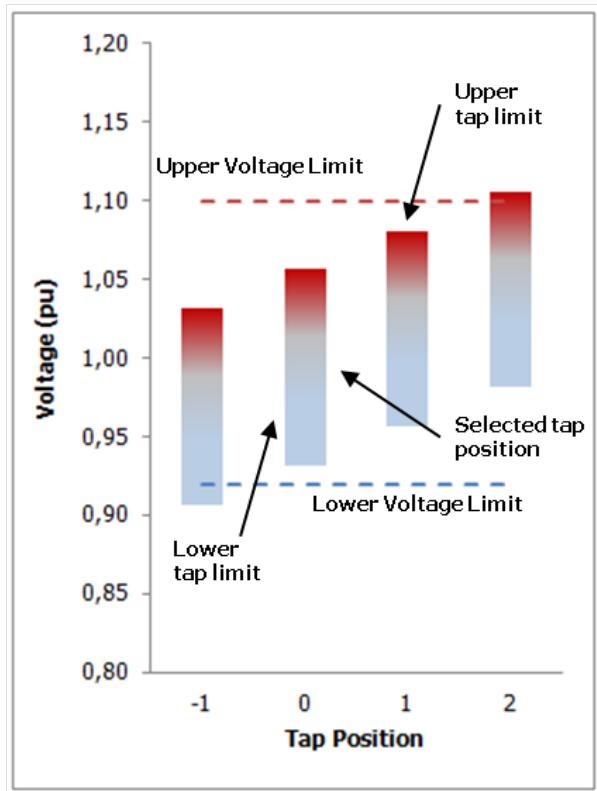


Figure 41.3.3: Voltage limits for LV grids with the objective function *Maximisation of generation* selected

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.

### 41.3.2 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.

### 41.3.3 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.

#### 41.3.4 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.

#### 41.3.5 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” (grid icon). An example of the Optimal Transformer Tap Positions section of the report is shown below in Figure 41.3.4 (results consistent with Figure 41.3.1 and the discussion in Section 41.3.1). 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										
MV Load Element	Transformer Type Data			Allowed	Opt. Tap	Voltages				
	Ratio	Limits	Neut. Add.V.			Tap	/Tap	Position	HV	LV
									LV-Grid (Min)	LV-Grid (Max)
	[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 41.3.4: 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 (grid icon), 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 (grid icon). Figure 41.3.5 shows an example plot for the consumption case only:

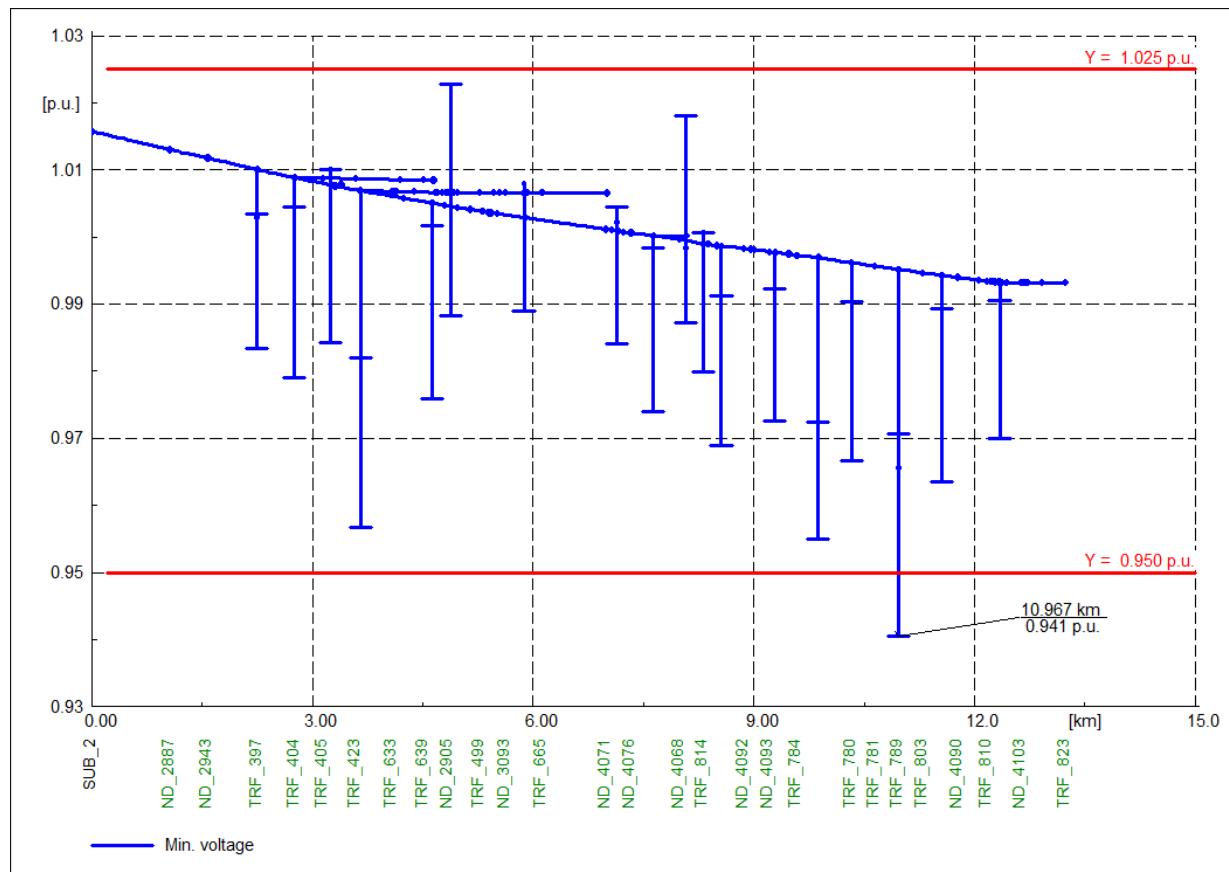


Figure 41.3.5: Voltage profile plot for the consumption case

## 41.4 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 chapter 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 Tie Open Point Optimisation Command can accessed as shown in Figure 41.1.1

### 41.4.1 Tie Open Point Optimisation Background

The function of the Tie Open Point Optimisation (TOPO) tool is best explained using an example. Consider the network illustrated in Figure 41.4.1

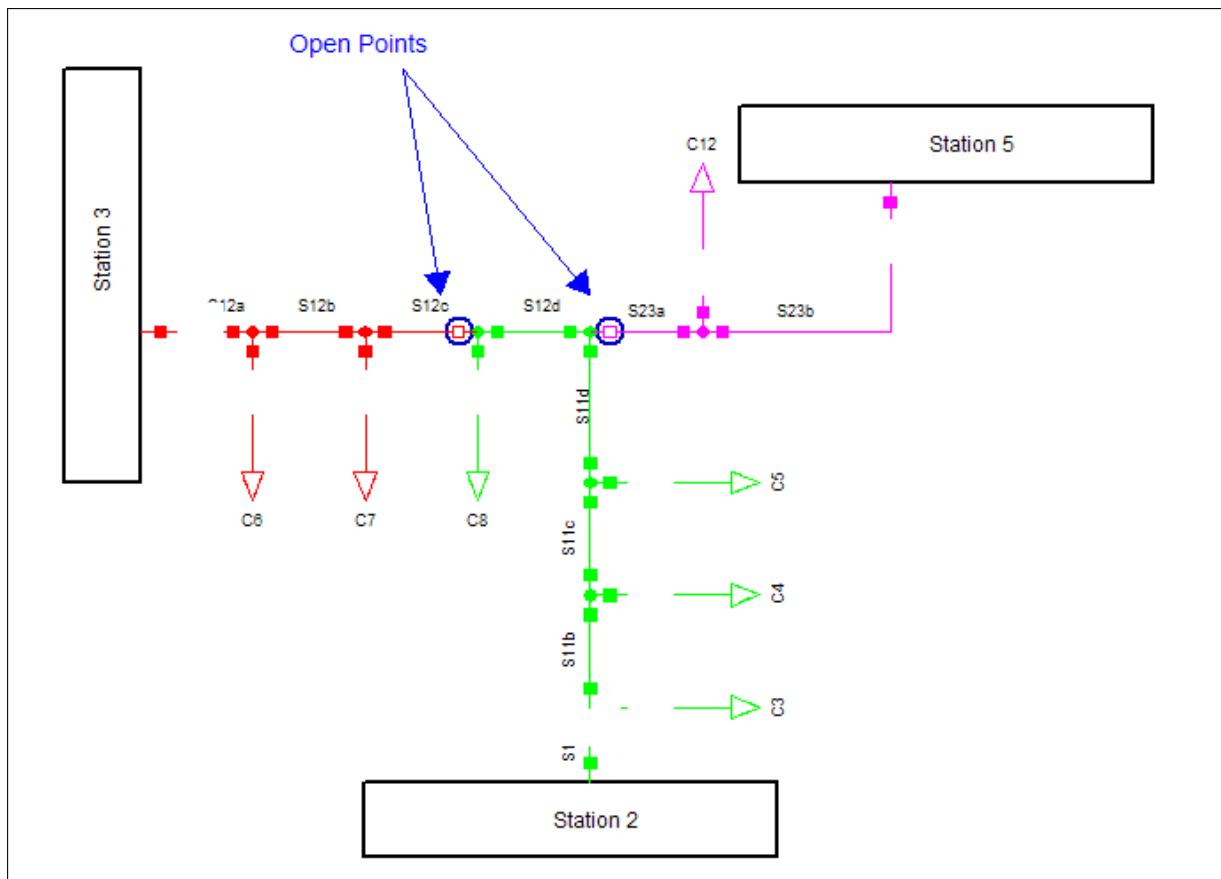


Figure 41.4.1: Example network for Tie Open Point Optimisation

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.

## 41.4.2 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 the Advanced Options.
- How to configure Reliability 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 *Define → Feeder*. Alternatively, for fast creation of multiple feeders right-click the bus the feeder/s are connected to and select the option *Define → Feeder*. More information on feeders and feeder creation can be found in Chapter 15: Grouping Objects, Section 15.5.

### 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 Tools*.
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 41.8 (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.
8. Optional: Change the “Saving of solution” option. The two options are as follows:
  - Change Existing Network (Operation Scenario). This is the default option. The TOPO tool modifies the base network model. Note that if a variation is active, the changes will be implemented in the variation.
  - 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.
9. Optional: Disable the “Report” flag. This control, enabled by default, allows you to turn off the automatic printing of an ASCII report to the output window.
10. Optional: Select the Before Optimisation and After Optimisation results objects.

### 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 [41.8](#) for details about this method and its input parameter.

## 41.5 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 Tools* as shown in Figure [41.1.1](#).
- From the *Data Manager* select and then right-click previously defined feeders and click *Calculate* → *Backbone Calculation*....
- From the main menu, select *Calculation* → *Distribution Network Tools*→ *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 41.5.4). The **Check Backbone** button is used to verify that the backbone object still defines a valid inter-feeder path matching its calculated parameters.

## 41.5.1 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).

## 41.5.2 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).

### 41.5.3 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 Tools → Start trace...*

### 41.5.4 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 *TopologyFeeders*. Click on Colour Settings, and on the Feeders page select *Highlight backbones*.

Figure 41.5.1 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 41.5.3 for details of how to trace the Backbone restoration steps.

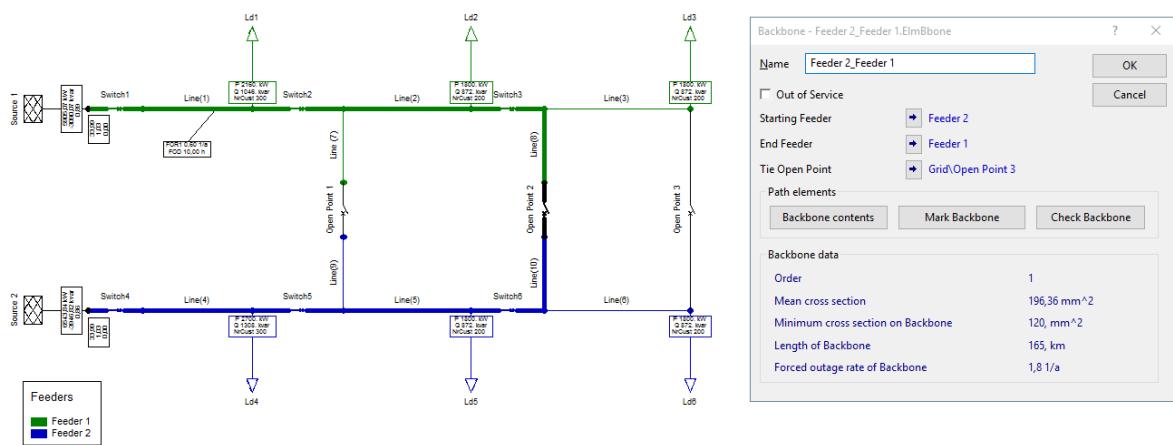


Figure 41.5.1: Example Backbone Calculation

## 41.6 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).

To access the OCP tool, select the OCP toolbar from the toolbar selection window as illustrated in Figure 41.6.1.

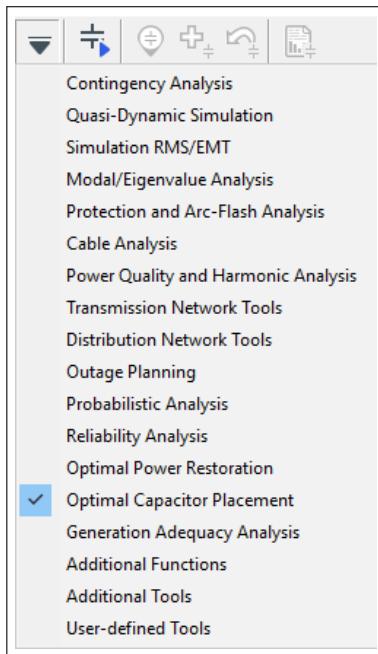


Figure 41.6.1: Optimal Capacitor Placement Tools

The buttons in the OCP toolbar are as follows:

- The main Optimal Capacitor Placement command is started with the *Calculate Optimal Capacitor Placement* icon ( $\frac{+}{\rightarrow}$ ). The command and the various user-defined options are described in detail in Sections 49.3.1 to 45.3.3.
- 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 ( $\oplus$ ).
- Following a successful OCP, the list of proposed capacitors can be accessed with the *Show New Capacitors* icon ( $\frac{+}{\pm}$ ).
- The *Remove previous solution* icon ( $\frac{\circlearrowleft}{\pm}$ ) deletes the results (removes all placed capacitors) from a previous OCP routine.
- To list all results from the OCP in a ASCII text report printed to the output window use the *Output Calculation Analysis* icon ( $\frac{\text{file}}{+}$ ). The report also displays the original system losses and voltage constraint costs and such costs after the installation of the proposed capacitors.

### 41.6.1 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:

$$TotalCosts = CLosses + \sum_{i=1}^m (CCap_i) + \sum_{i=1}^n (CVoltViol_i) \quad (41.1)$$

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' tab of the OCP command dialog ('vmin' and 'vmax' parameters, see Section 49.3.1: Basic Options Page). The penalty factors are defined in the 'Advanced Options' tab of the same command ('weight' and 'weight2' fields, see Section 45.3.3: 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:

$$CVoltViol = w1 \cdot \Delta U \quad (41.2)$$

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' tab.

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' tab.
- $w2$  is the penalty factor (parameter 'weight2') for voltage outside the admissible voltage band in \$/% from the 'Advanced Options' tab.

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 41.6.2 illustrates the concept of the voltage band violation cost.

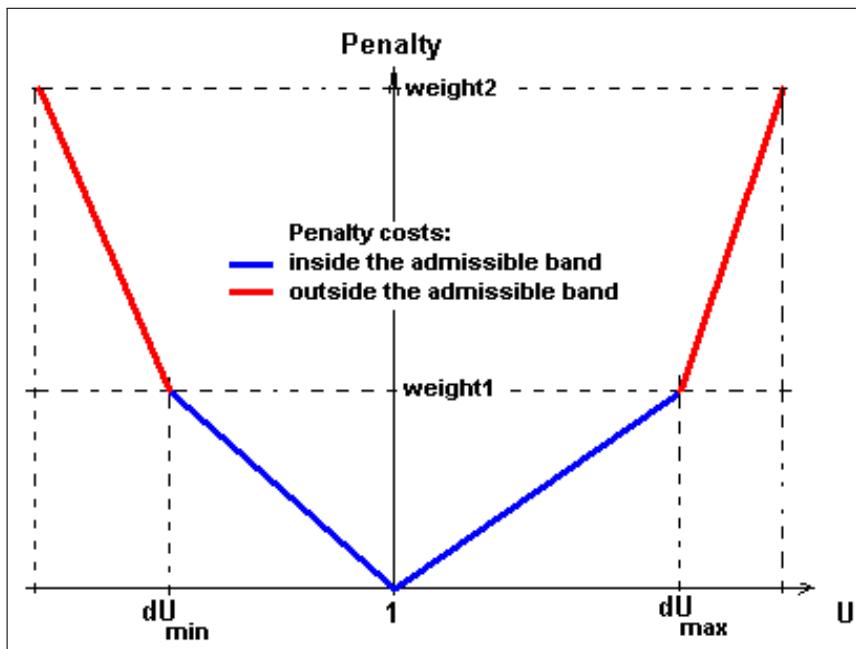


Figure 41.6.2: Fictitious cost assigned by voltage band violations

## 41.6.2 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 41.6.5 for how the load states are determined.

### 41.6.3 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 15.5 (Feeders)).

#### Method

- Optimisation; This option calculates the optimal placement for capacitors using the methodology described in Section 41.6.2. 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 41.6.1: 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:

$$TC = MC \times 8760 \times L \quad (41.3)$$

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 41.6.5), then the losses calculation becomes a summation over each time state considered.

**Note:** The default energy cost units are \$/kWh. However, this can be changed to Euro or Sterling (£) via the project settings from the main menu bar. *Edit → Project... Project Settings → Input Variables tab → Currency Unit.*

---

### 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 45.3.3). 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 41.6.7.

#### 41.6.4 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 nominal 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.

### 41.6.5 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.

### 41.6.6 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 41.6.2.

#### 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.

### 41.6.7 Results

The last 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.

#### Output Calculation Analysis

This *Output Calculation Analysis* icon () generates 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 41.6.3.

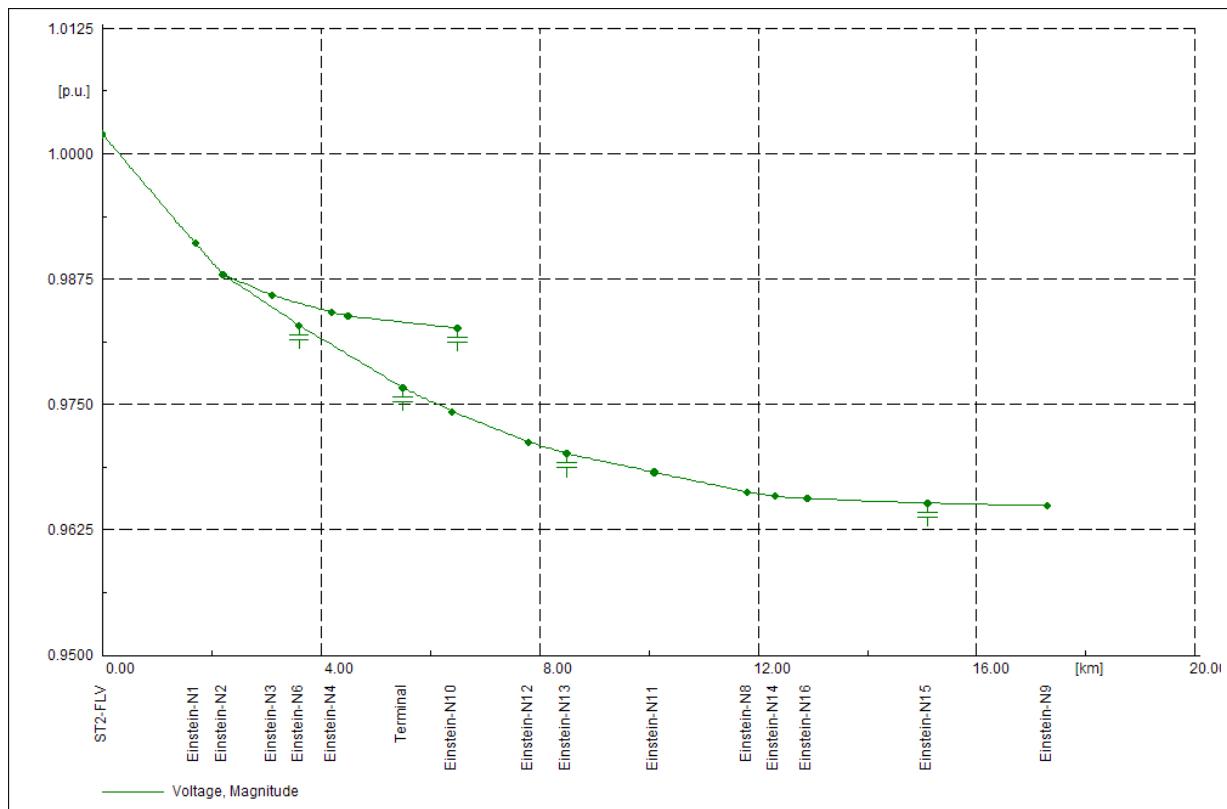


Figure 41.6.3: Voltage profile plot showing the new capacitors after an Optimal Capacitor Optimisation.

### 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). This button is like an 'Undo' for the 'Optimal Capacitor Placement'.

## 41.7 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.

### 41.7.1 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.

## 41.7.2 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 [41.8](#).

#### 41.7.3 Considered elements

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.

#### 41.7.4 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.

#### 41.7.5 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.

### 41.8 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

### 41.8.1 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 41.8.1 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.

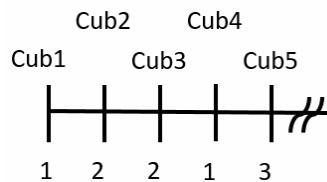


Figure 41.8.1: Sample genotype for Phase Balance Optimisation simulation

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 41.8.2.

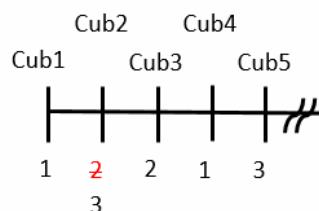


Figure 41.8.2: Sample genotype mutation for Phase Balance Optimisation simulation

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.

### 41.8.2 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.

## 41.9 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:

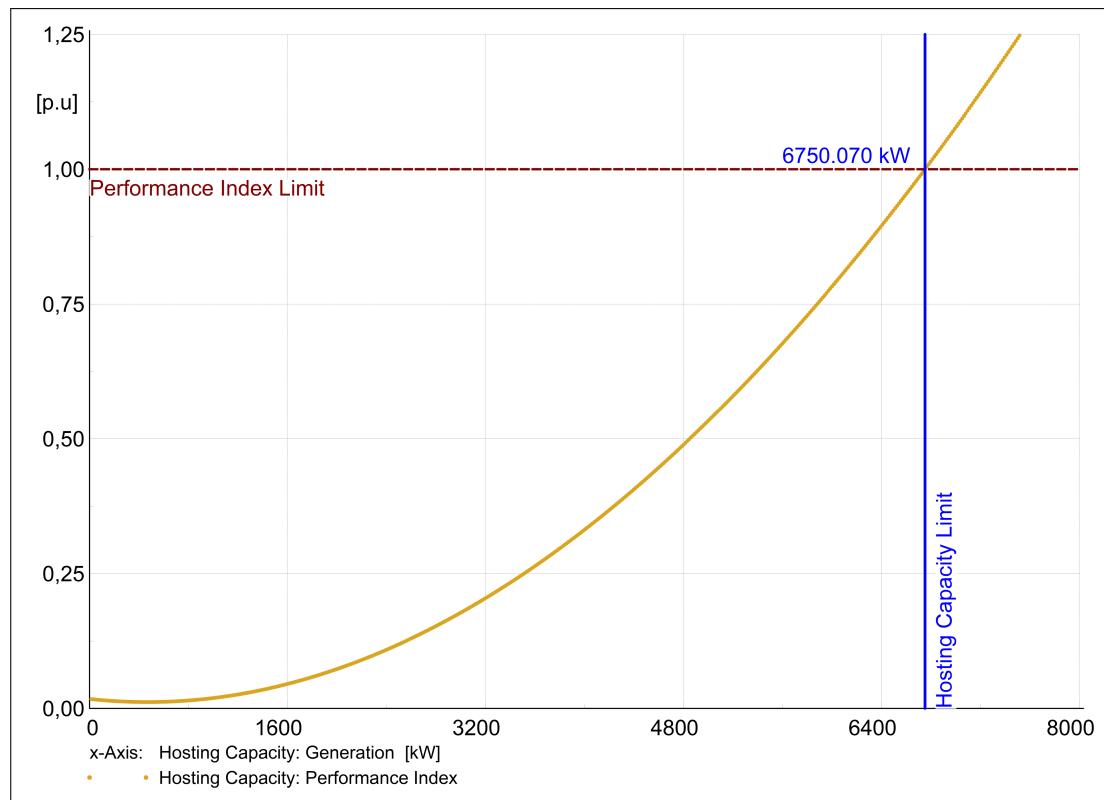


Figure 41.9.1: Performance Index limiting the max. amount of generation/load

### 41.9.1 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 41.9.2, 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 41.9.2.1) 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 41.9.2.3 and 41.9.2.4), 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 41.9.2.1, 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 41.9.2.3) 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 41.9.2.4). A load flow calculation is executed and the constraints defined by the user on the *Constraints* page (see Section 41.9.2.2) 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.

## 41.9.2 Hosting Capacity Analysis Configuration

The Hosting Capacity analysis can be opened by pressing *Hosting Capacity Analysis* button  in the Distribution Network Tool toolbar. More information about the available settings can be found in the following subsections.

### 41.9.2.1 Basic Options Page

#### Hosting sites

Contains one or multiple:

- Terminal(s)
- Feeder(s)
- Area(s)
- Zone(s)
- Grid(s)

If a grouping object is selected, the calculation will extract the relevant terminals.

Hosting sites can either be selected from the network graphic, Network Model or Data Manager, by right clicking on the selection and choosing *Calculate → Hosting Capacity Analysis...* or through the command. In the former case, a general selection is automatically assigned. In the latter case, the user has to click on  and then *Select....*. A selection browser will open, from which the desired object(s) can be selected (multi-selection is possible). After confirming with **OK**, a general set is also created automatically. In both cases, the set can be edited by clicking on .

#### 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.

### 41.9.2.2 Constraints

#### Thermal limits:

- Maximum thermal loading of elements can be defined globally or individual constraints per element can be considered.

#### Voltage limits:

1. upper and lower voltage limit can be entered as global constraint or individual constraints per node can be defined.

**Important:** The following three settings only apply if the *Advanced Option Ignore terminals which are not part of a feeder* is selected or the selection of *Hosting Sites* only contains one or more feeders.

2. within the setting *voltage regulator tap change*, a limit of the number of tap changes can be defined.
  - the maximum number of taps for voltage regulator and capacitor can be defined.
3. The *Permissible voltage change* setting analyses the maximum voltage change ratio between no DER infeed and full DER infeed.
  - For this analysis, a separate Load Flow Command can be defined.
  - The assessment is based on the generators selected within the *Advanced tab, Generators to be switched off*.
4. The *Max. voltage unbalance factor* is a relevant constraint in case of having the *Basic Option System type* selected to be *unbalanced*.

#### **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. Ignore all constraints for nominal voltage below or above the given value.
2. Constraints check
  - *Per feeder*: Only violated constraints within feeders will be assessed.
  - *Whole system*: Constraints are checked within the whole system.
3. 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.
4. Short circuit calculation
- The Short Circuit calculation can be executed at all terminals within the feeder or only at the DER location.

#### 41.9.2.3 Configuration

Depending on the *Calculation objective* selected on the Basic Options page (see Section 41.9.2.1), 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 41.9.1.
  - *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 41.9.1: 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 41.9.2.1). 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 41.9.2.1). Depending on the settings on the *Power Quality* tab of the *Constraints* page (see Section 41.9.2.2), 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 the 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 the 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 load only to those terminals where no load element is already connected. Where loads are directly connected to the analysed terminals, they will be taken and scaled instead.

If the *Connect new load* option is enabled, the following two settings can be configured:

- *Initial load active power:* Active power value from which the algorithm starts
- *Load power factor:* The Power Factor used for the load can be specified

#### 41.9.2.4 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.

#### 41.9.2.5 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.

---

#### 41.9.2.6 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 41.9.2.1). 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.

#### 41.9.2.7 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 computing manager:* Refer to Section 22.4 for more information.

### 41.9.3 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.

#### 41.9.3.1 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 9.10 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*.

### 41.9.3.2 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 Tools 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 table 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 table report, displaying the connectable power per phase.
- *Show apparent power*: If selected, a column will be added to the table 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. For example: If the maximum voltage is 1.038 p.u., the limit is 1.04 p.u. and the threshold is 99.8%, the voltage value will be coloured. A maximum voltage of 1.037 p.u. however won't be coloured.

**Title:** Link to the title block. More information about the Title block can be found in Section [9.3.8](#).

**Used format:** Links to the different format templates for the table reports.

#### 41.9.3.3 Load Hosting Capacity Results

By clicking on the *Load Hosting Capacity Analysis Results* button  in the Distribution Network Tools 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.

# Chapter 42

## Outage Planning

### 42.1 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 Planning toolbar is provided, to facilitate the handling of the Planned Outage objects.

### 42.2 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)*.

#### 42.2.1 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, *Define* → *Planned Outage* 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.

### 42.2.2 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 drop-down menu. Elements can then be selected to populate the Outaged components list.

### 42.2.3 Recurrent Outages

It is possible to define recurrence patterns for a planned outage. The box *Recurrent* is checked and then the recurrence pattern can be edited. There are flexible options for specifying the recurrence of the outage, and the outage is then applicable according to this specification within the start and end dates and times defined on the Basic Data page.

### 42.2.4 Adding Additional Events to an Outage

A Planned Outage object can be modified to include events in addition to the outaging 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 Planning toolbar (see section 42.3.4) 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.

## 42.3 Handling Planned Outages using the Outage Planning toolbar

The Outage Planning toolbar offers the following options to facilitate working with Planned Outages.

### 42.3.1 Show Planned Outages

Pressing this button will bring up a filter of all the outage objects in the Outages folder of the Operational Library.

### 42.3.2 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. Outages are applied in sequential order of start date, but if outages have the same start date, then there is a Priority flag on the Planned Outage object which is used to determine which takes precedence. The parameter is called priority, and the lower the number the higher the priority of that outage. (If the priorities are also the same, the outages will then just be applied in alphabetical order of name.)

### 42.3.3 Reset All Planned Outages

This button will reset all the Planned Outages which have been applied.

### 42.3.4 Start Recording

If this button is pressed, a filter of possible outages in which to record events is presented. Only un-applied outages appear in this list. 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.

### 42.3.5 Outage Schedule Report

This button is used to generate a list of all Planned Outages which have not yet been applied to the network. 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 un-applied Planned Outages is presented in tabular format, including a bar-chart to give an overview of the outage plan.

# Chapter 43

# Probabilistic Analysis

## 43.1 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”.

## 43.2 Technical Background

### 43.2.1 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 43.3.1, whereas the technical background is described here.

#### 43.2.1.1 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  $b$ :

$$F(a \leq X \leq b) = \int_a^b f(t)dt \quad (43.1)$$

- **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.

#### 43.2.1.2 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:

$$f(k; p) = p^k(1 - p)^{1-k} \quad for k \in \{0,1\} \quad (43.2)$$

Mean and standard deviation of this function are calculated by:

$$\mu = p, \quad \sigma = \sqrt{p(1 - p)} \quad (43.3)$$

##### 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 ( $w$ ):

$$p_k = \frac{w_k}{\sum_{i=1}^N w_i} \quad (43.4)$$

**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:

$$f(t) = \begin{cases} 0 & \text{for } t < 0, \\ \lambda e^{-\lambda t} & \text{for } t \geq 0 \end{cases} \quad (43.5)$$

Mean and standard deviation of this function are calculated by:

$$\mu = \frac{1}{\lambda}, \quad \sigma = \frac{1}{\lambda} \quad (43.6)$$

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:

$$f(k; p) = (1 - p)^k p \quad \text{for } k \in \{0, 1, 2, 3, \dots\} \quad (43.7)$$

Mean and standard deviation of this function are calculated by:

$$\mu = \frac{1 - p}{p}, \quad \sigma = \sqrt{\frac{1 - p}{p^2}} \quad (43.8)$$

**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:

$$f(t) = \begin{cases} 0 & \text{for } t \leq 0, \\ \frac{1}{t\sigma_L\sqrt{2\pi}} \exp\left(-\frac{(\log t - \mu_L)^2}{2\sigma_L^2}\right) & \text{for } t > 0 \end{cases} \quad (43.9)$$

Mean and standard deviation of this function are calculated by:

$$\mu = \exp(\mu_L + \frac{\sigma_L^2}{2}), \quad \sigma = \sqrt{(\exp \sigma_L^2 - 1) \exp(2\mu_L + \sigma_L^2)} \quad (43.10)$$

**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:

$$f(t) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(t-\mu)^2}{2\sigma^2}} \quad (43.11)$$

If  $\mu = 0$  and  $\sigma = 1$  the standard normal distribution with the probability density function is obtained:

$$f(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}t^2} \quad (43.12)$$

**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:

$$f(t) = \begin{cases} \frac{1}{b-a} & \text{for } a \leq t \leq b, \\ 0 & \text{for } t < a \text{ or } t > b \end{cases} \quad (43.13)$$

The interval boundaries  $a$  and  $b$  can be calculated from the input parameters *Mean* ( $\mu$ ) and *Magnitude* in *PowerFactory* by:

$$a = \text{Mean} - \text{Magnitude} \quad b = \text{Mean} + \text{Magnitude} \quad (43.14)$$

The standard deviation can then be calculated by:

$$\sigma = \sqrt{\frac{1}{12}(b-a)^2} = \sqrt{\frac{1}{12}(2 \cdot \text{Magnitude})^2} \quad (43.15)$$

### 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:

$$f(t) = \begin{cases} 0 & \text{for } t < 0, \\ \frac{\alpha}{\beta} \left(\frac{t}{\beta}\right)^{\alpha-1} \exp[-(\frac{t}{\beta})^\alpha] & \text{for } t \geq 0 \end{cases} \quad (43.16)$$

Mean and standard deviation of this distribution are calculated by:

$$\mu = \beta \Gamma(1 + \frac{1}{\alpha}), \quad \sigma = \sqrt{\beta^2 \left[ \Gamma\left(1 + \frac{2}{\alpha}\right) - \Gamma^2\left(1 + \frac{1}{\alpha}\right) \right]}, \quad (43.17)$$

in which  $\Gamma(x)$  is the Gamma function:

$$\Gamma(x) = \int_0^{\infty} t^{x-1} e^{-t} dt \quad \text{for } x > 0. \quad (43.18)$$

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  $\sigma = 4.169 \text{ m/s}$ .

#### 43.2.1.3 Transformed Distribution

This distribution (*RndTransformed*) can be used to transform any distribution using a Wind Power Curve (*TypPowercurve*, see Section 46.3.2).

Figure 43.2.1 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 (nominal power) of the cumulated distribution curve.

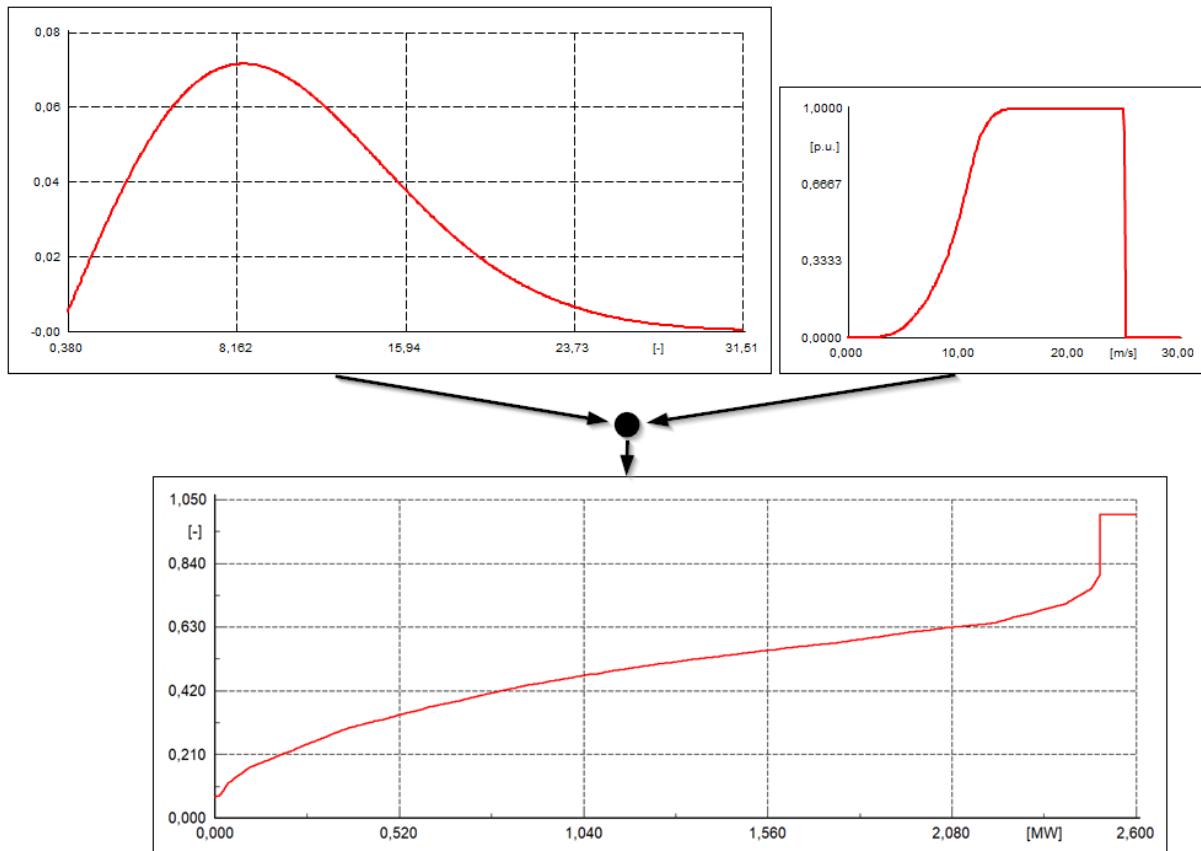


Figure 43.2.1: Process of transforming a distribution

#### 43.2.1.4 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 [43.2.5](#).

## 43.2.2 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.

#### 43.2.2.1 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.

#### 43.2.3 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

### 43.2.3.1 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.

### 43.2.3.2 Monte Carlo Method

The Monte Carlo method draws individual samples randomly, as the left plot of Figure 43.2.2 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:

$$E[f(X)] = \int f(X) dP \approx \sum_{n=1}^N f(X_n) \quad (43.19)$$

where the convergence rate is of order

$$O(1/\sqrt{N}) \quad (43.20)$$

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 43.2.4 for detailed information). These are:

- Mean,
- Standard deviation,
- Maximum,
- Minimum,
- Variance,
- 1st - 5th Moment and
- confidence interval for mean and standard deviation.

### 43.2.3.3 Quasi-Monte Carlo Method

The Quasi-Monte Carlo method uses special sequences, in order to cover the space more uniformly (see right side of Figure 43.2.2). The rate of convergence of a Quasi-Monte-Carlo simulation is given by:

$$O(\ln(N)^d/N) \quad (43.21)$$

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.

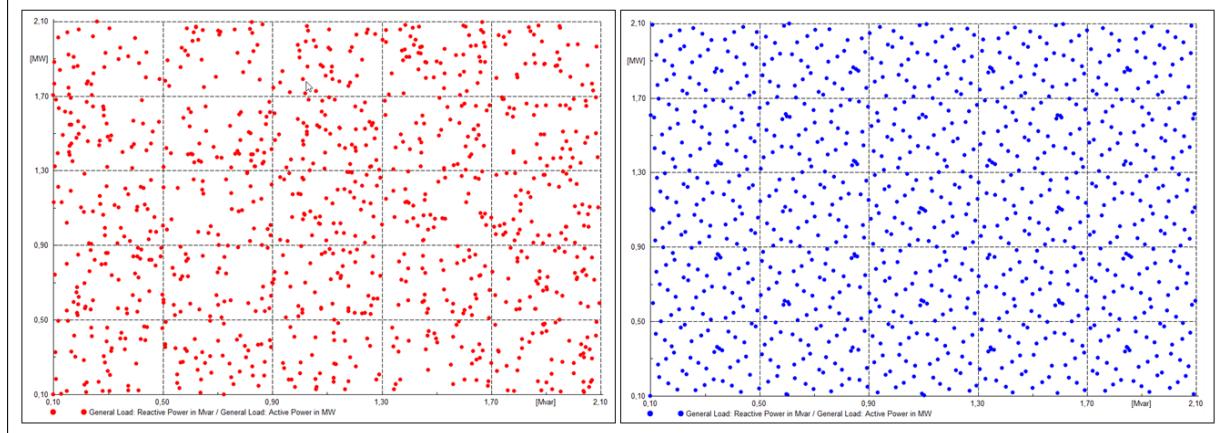


Figure 43.2.2: Comparison between Monte Carlo (left) and Quasi-Monte Carlo (right) samples

#### 43.2.3.4 Comparison of Methods

The difference between the methods lies in the sequence of samples as shown in Figure 43.2.2. 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 43.2.4 for more information about the confidence interval).

Method	MC	QMC
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 43.2.1: Comparison between Monte Carlo (MC) and Quasi-Monte Carlo (QMC)

#### 43.2.4 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:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i \quad (43.22)$$

- **Variance:**

The empirical variance  $s^2(x)$  of a variable is calculated by:

$$s^2(x) = \frac{1}{n-1} \sum_{k=1}^n (x_k - \bar{x})^2 \quad (43.23)$$

- **Standard deviation:**

The empirical standard deviation  $s(x)$  is calculated by:

$$s(x) = \sqrt{s^2(x)} \quad (43.24)$$

- **Moments (up to fifth order):**

The empirical  $l$ -th moment  $m_l(x)$  is calculated by:

$$m_l(x) = \frac{1}{n} \sum_{k=1}^n x_k^l. \quad (43.25)$$

- **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:

$$p(I,x) = \frac{1}{n} \sum_{k=1}^n 1_{x_k \in I} = \frac{1}{n} \#\{x_k \in I : 1 \leq k \leq n\} \quad (43.26)$$

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 43.2.3.3).

---

### 43.2.5 Distribution Estimation

Two estimation methods are available, which are explained in the following subsections.

- Bootstrapping
- Histogram

#### 43.2.5.1 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 43.2.3. A number between zero and one is randomly drawn. The corresponding x-value is then taken for the analysis.

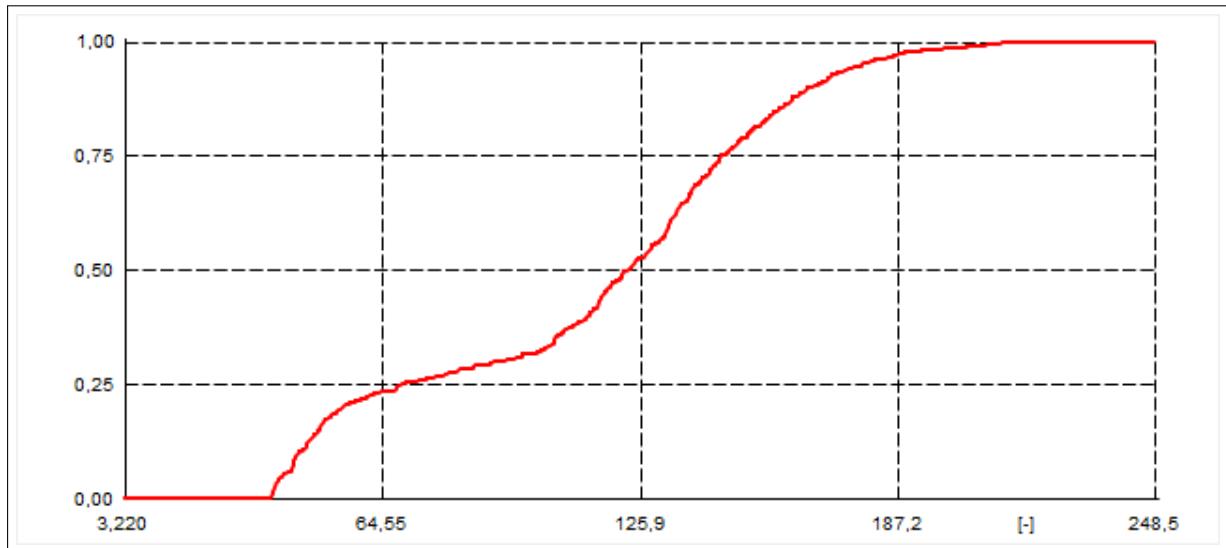


Figure 43.2.3: Bootstrapping - Cumulative distribution function

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:

$$\mu = \frac{1}{n} \sum_{i=1}^N x_i, \quad \sigma = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2} \quad (43.27)$$

#### 43.2.5.2 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.

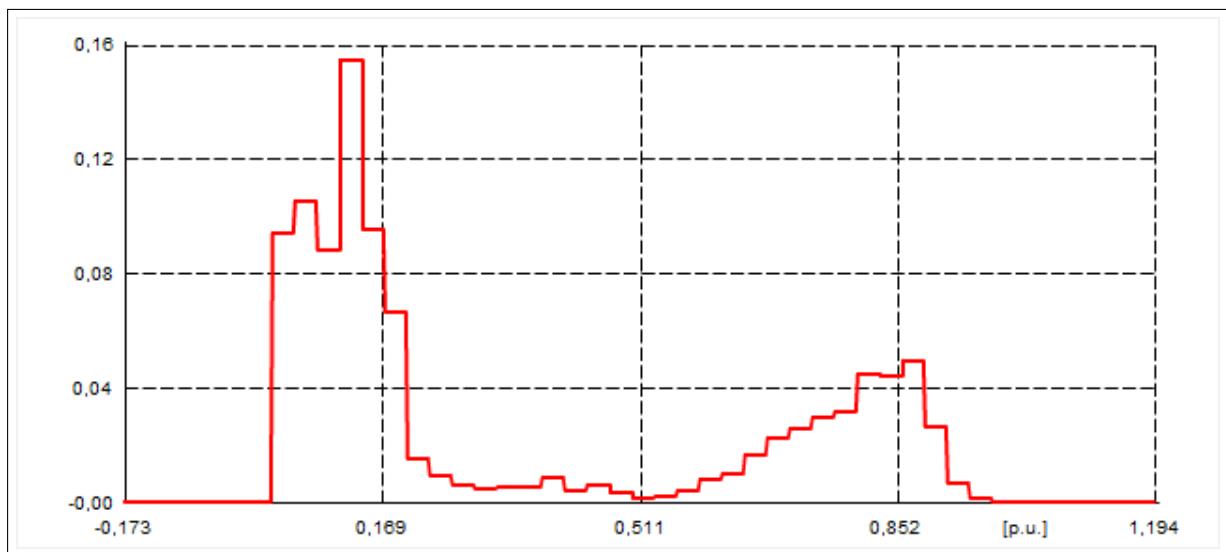
**Automatic:** will choose the maximum number of bins obtained using the Rice Rule and the Freedman-Diaconis choice.

**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 43.2.4 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.



In both cases, the approximate cdf / pdf are given as follows:

$$pdf : f(x) = \sum_{j=0}^n c_j \phi^{(j)}(x) \quad (43.28)$$

$$cdf : F(x) = \sum_{j=0}^n c_j \Phi^{(j)}(x), \quad (43.29)$$

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 [21] or [26].

## 43.3 Object Settings

In the following subsections the settings of the commands and objects related to the Probabilistic Analysis are described.

### 43.3.1 Distributions

#### 43.3.1.1 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 43.3.3).

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).

#### 43.3.1.2 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.

$$r = \frac{d}{100} \cdot p \quad (43.30)$$

- **Multiplicative:** drawn value is multiplied with the set parameter value.

$$r = d \cdot p \quad (43.31)$$

- **Absolute:** drawn value (absolute value) is directly applied to the parameter.

$$r = d \quad (43.32)$$

- **Additive in %:** from the drawn percentage value and the set parameter value an absolute value is calculated, which is added to the set parameter.

$$r = \frac{d}{100} \cdot p + p \quad (43.33)$$

- **Additive:** from the drawn value and the set parameter value an absolute value is calculated, which is added to the set parameter.

$$r = d + p \quad (43.34)$$

- **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.

### 43.3.1.3 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 43.2.1.2.

### 43.3.1.4 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 43.2.1.3.

**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 43.2.1.2.

### 43.3.1.5 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 [43.3.1.7](#).

**Distribution Estimation:** one of the methods described in Section [43.2.5](#) can be selected.

---

**Note:** If a PV System with Solar Calculation Model has been assigned as characteristic, the Bootstrapping method will automatically be applied.

---

### 43.3.1.6 Parameter Distribution - Reference

This reference (*RndRef*) is the equivalent to the Characteristic Reference (*ChaRef*). It connects the distribution with the element's parameter.

### 43.3.1.7 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.

## 43.3.2 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 “Define...” “Distribution correlation”
- Through Element Filter  
Within the Element Filter, it is also possible to mark several elements “Define...” “Distribution correlation”

There is the possibility to choose between the correlation definitions, explained within the following subsections.

### 43.3.2.1 Elliptic Copula - equally

Within this element the correlation between elements and/or parameters can be defined. Once several elements are defined according to Section [43.3.2](#), 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 [43.2.2](#) can be selected.

#### 43.3.2.2 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 [43.2.2](#).

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 [43.2.2.1](#). 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.

#### 43.3.2.3 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 [43.3.1.7](#).
- 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 [43.2.2.1](#).

#### 43.3.3 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 [43.2.1.4](#) and [43.2.5](#).

### 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 [43.2.5](#).

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 [43.3.1.7](#). 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.
- **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 [43.3.2.3](#)) will be created and all characteristics which have been selected on the *Characteristics* page, will be assigned to this copula.

<sup>1</sup>A characteristic is called *active*, if it has been assigned to a parameter within the network.

- **Add to existing correlation:** an existing copula (*ElmCopellcha*, see Section 43.3.2.3) can be selected, to which the characteristics defined on the *Characteristics* page will be added.

### 43.3.4 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 38.  
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 43.2.3, it is possible to choose between the Monte Carlo (43.2.3.2) or Quasi-Monte Carlo (43.2.3.3) 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 43.3.7.
- 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 43.3.7.

#### 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 43.2.4. 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 43.3.7. 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 43.2.3.1.
- 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).

### 43.3.5 Continue Probabilistic Analysis

Within the *Continue Probabilistic Analysis* command , 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”.

### 43.3.6 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.

### 43.3.7 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.

**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 43.2.4) 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 43.3.4). The results files (*ElmRes*) for the Probabilistic Analysis are stored within a folder called *Probabilistic assessment* inside the study case.

---

<sup>2</sup>Some standard parameters for selected object classes have already been predefined.

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 Section [10.3.3](#)) can be defined. This makes it possible for example to record parameters for elements of that class which are inside a specific area.

### 43.3.8 Representation of results

#### 43.3.8.1 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, mean value* or *Format for Edge Elements* → *Branch flow, mean value*). 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 [43.3.1](#).

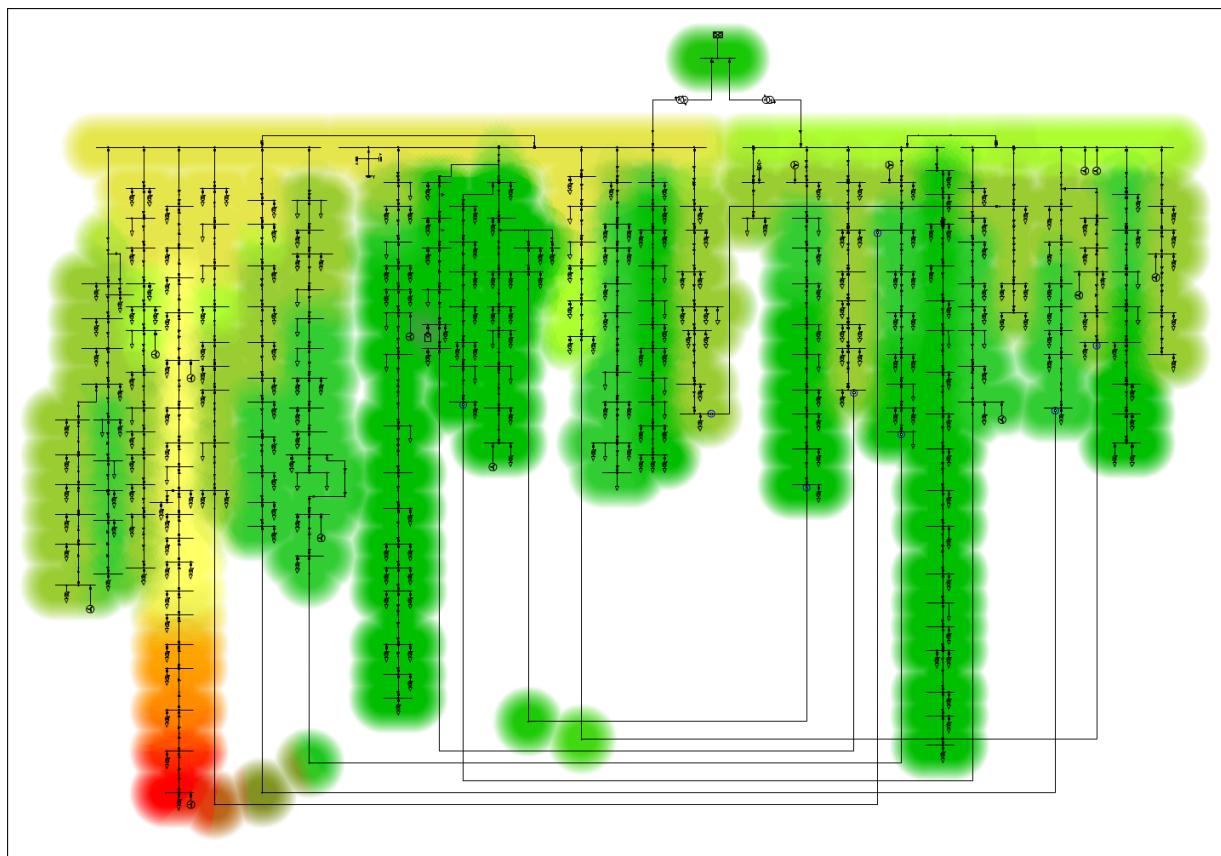


Figure 43.3.1: Colouring of statistics (mean value) of Load Flow quantities in the Single Line Diagram

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 43.2.4.

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.

### 43.3.8.2 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.

### 43.3.8.3 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 43.2.4.

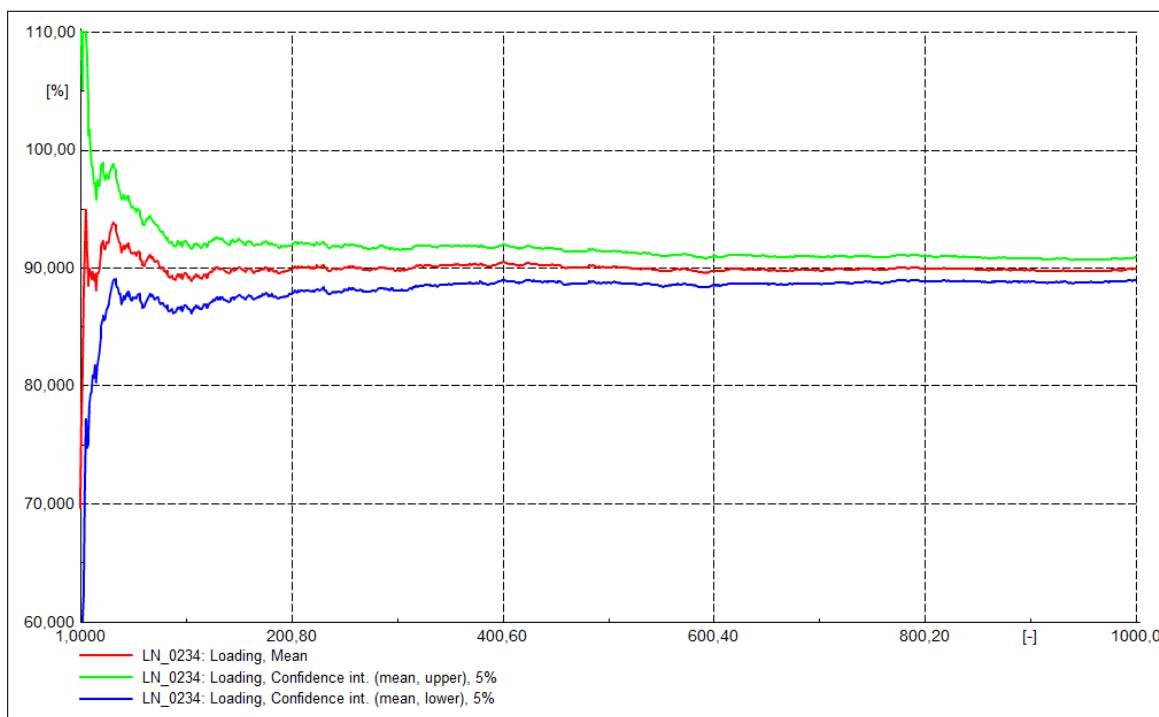


Figure 43.3.2: Convergence Plot of Line Loading Mean value (red) and upper (green) and lower (blue) 5% Confidence interval

### 43.3.8.4 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.

### 43.3.8.5 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 43.2.5, 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 43.3.3 (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.

#### 43.3.8.6 Distribution Fitting Plot

The Distribution Fitting Plot gives the distribution fitting according to the methods, explained in Section 43.2.6. In context with these methods, an order of fitting can additionally be entered for fitting.

The blue cumulative distribution function of Figure 43.3.3 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.

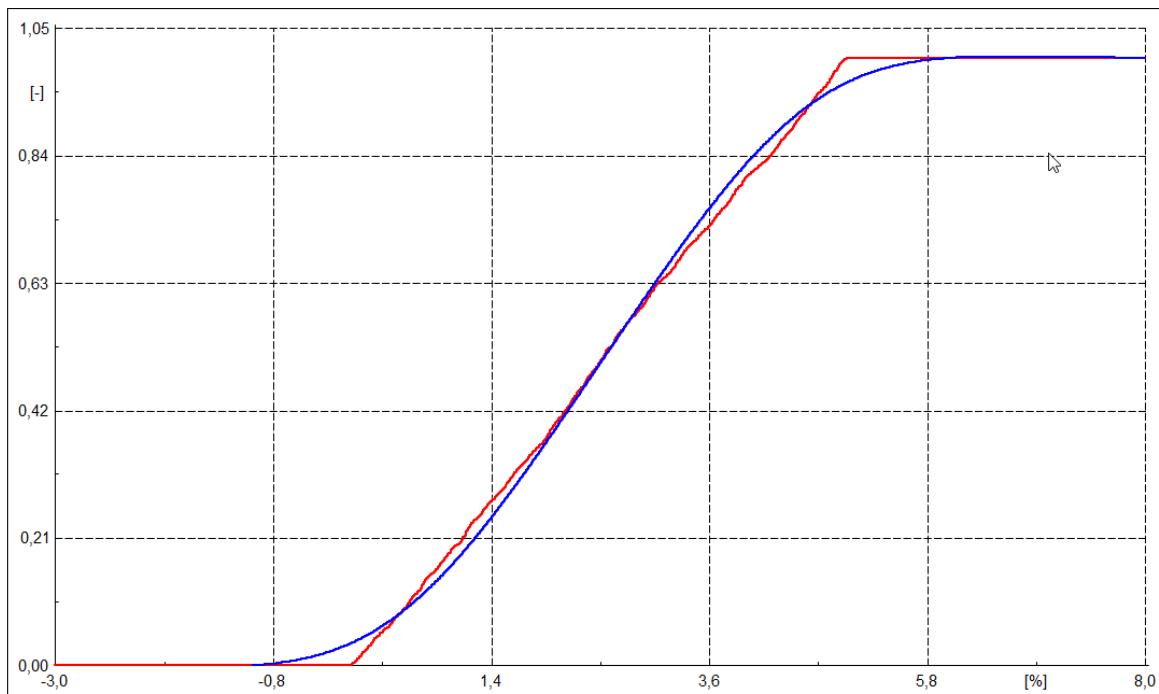


Figure 43.3.3: Cumulative distribution function. Comparison between Distribution Estimation - Bootstrapping (red curve) and Distribution Fitting - Edgeworth Expansion, Ord. 4 (blue curve)

#### 43.3.8.7 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 44

## Reliability Analysis

### 44.1 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'.
- Repair times are normally expressed in [H] = 'hours'.
- Probabilities or expectancies are 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 control 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 reliability assessment functions can be accessed by selecting *Reliability* toolbar from the *Change Toolbox* icon (▼) as illustrated in Figure 44.1.1.

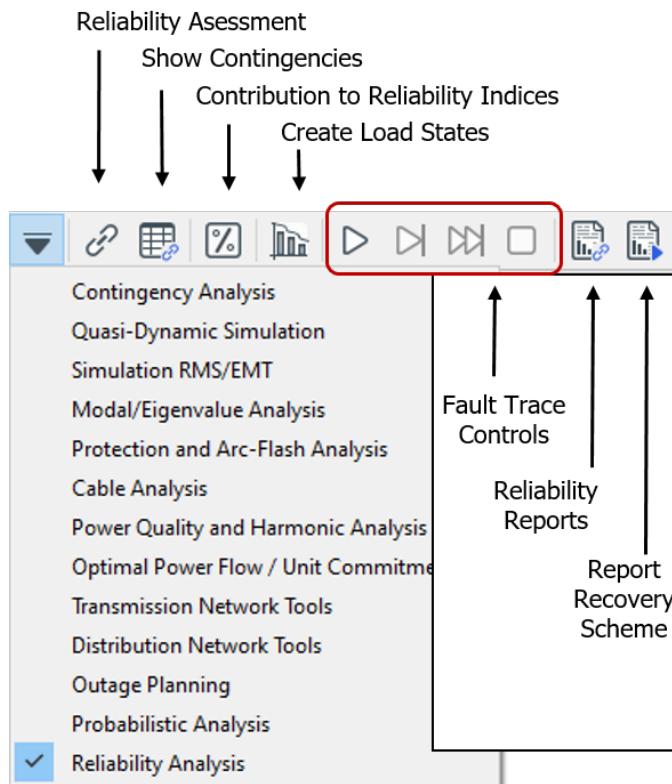


Figure 44.1.1: Reliability Toolbar Selection

The basic user procedure for completing a reliability assessment consists of the following steps as shown in Figure 44.1.2. Steps on the left are compulsory, while steps on the right are optional and can be used to increase the detail of the calculation.

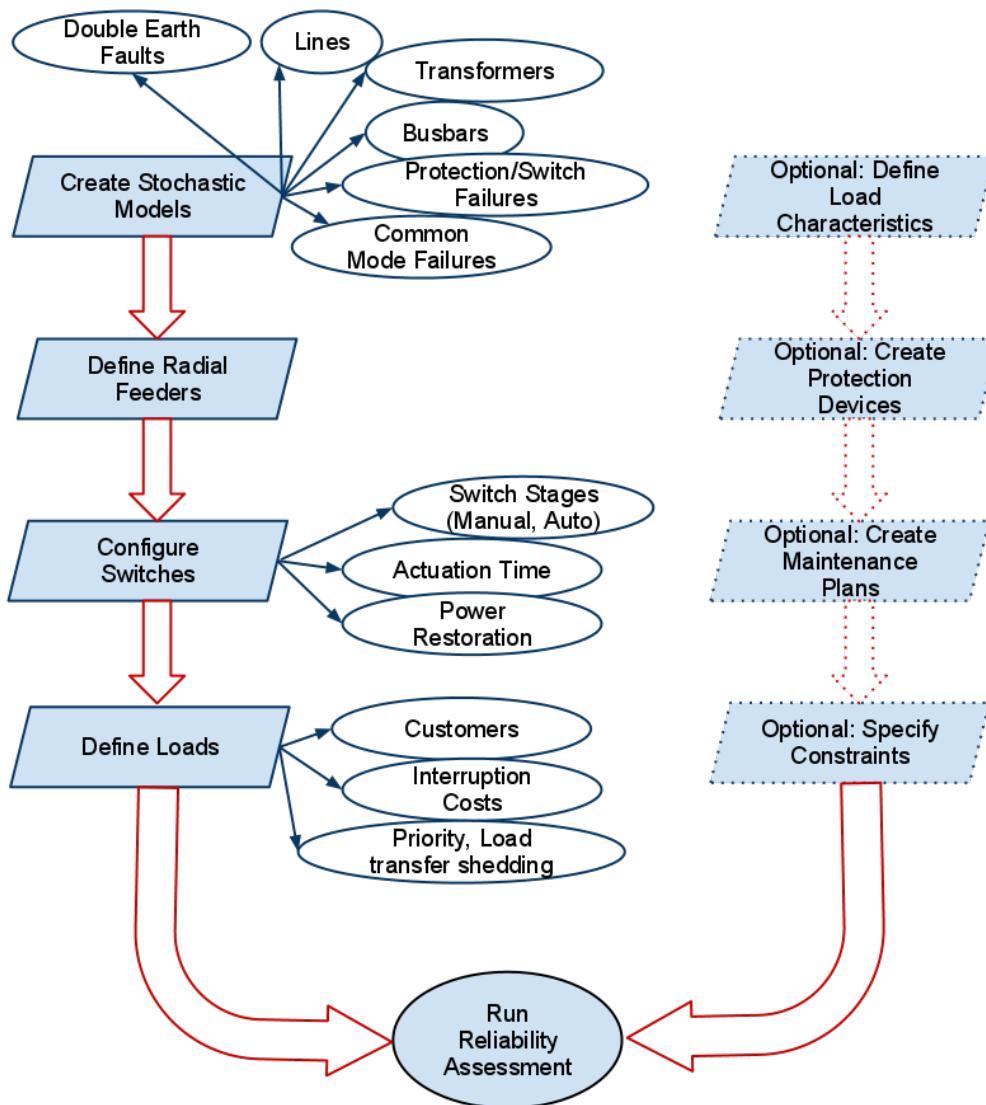


Figure 44.1.2: Reliability Assessment User Procedure

These procedures are explained in detail in the following sections.

## 44.2 Probabilistic Reliability Assessment 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.

### 44.2.1 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.

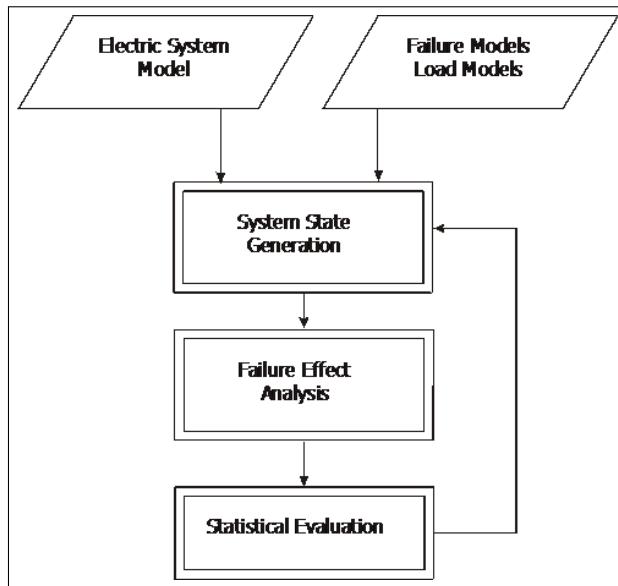


Figure 44.2.1: Reliability Analysis: Basic Flow Diagram

The reliability analysis calculation flow diagram is depicted in Figure 44.2.1. 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 analyse 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:

- Fault clearance by tripping of protection breakers or fuses.
- Fault separation by opening separating switches.
- Power restoration by closing normally open switches.
- Overload alleviation by load transfer and load shedding.
- Voltage constraint alleviation by load shedding (only available when 'Distribution' is selected within the reliability command Basic Options).

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).

#### 44.2.2 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 outage, 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 44.2.2.

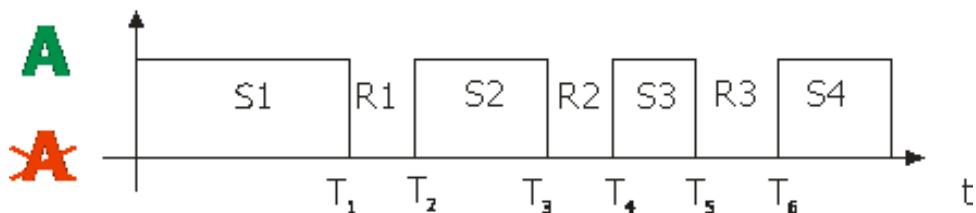


Figure 44.2.2: Line availability states are described by the status of the line (in service or under repair). Each of these states lasts for a certain time.

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  $S_1 = T_1 - T_0$ ,  $S_2 = T_3 - T_2$ , 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  $Q = \text{TTR}/(\text{TTF}+\text{TTR})$ ;

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  $P+Q = 1.0$ .

---

**Reminder:** TTR is the 'Time To Repair', and TTF is the 'Time To Failure'.

---

### 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:

$$\lambda = \frac{140}{10 \cdot 7500} \cdot \frac{1}{a} = 0,00187 \cdot \frac{1}{a} \quad (44.1)$$

$$TTF = \frac{1}{\lambda} = 536a \quad (44.2)$$

$$TTR = \frac{7360}{140} \cdot h = 52,6h = 0,006a \quad (44.3)$$

$$\mu = \frac{1}{TTR} = 167 \cdot \frac{1}{a} \quad (44.4)$$

$$P = \frac{536}{536 + 0,006} = 0,999989 \quad (44.5)$$

$$Q = \frac{0,006}{536 + 0,006} = 6 \frac{\text{min}}{a} \quad (44.6)$$

i.e. the expected outage duration is 6 minutes per annum.

### 44.2.3 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.

Load point indices are calculated for each load (*ElmLod*), 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.

#### 44.2.3.1 Parameter Definitions

In the definitions for the reliability indices, the following parameters are used:

- $C_i$  The number of customers supplied by load point i
- $A_i$  The number of affected customers for an interruption at load point i
- $Fr_k$  The frequency of occurrence of contingency k
- $pr_k$  The probability of occurrence of contingency k
- $C$  The number of customers
- $A$  The number of affected customers
- $L_m$  The total connected kVA interrupted, for each interruption event, m
- $r_m$  Duration of each interruption event, m
- $L_T$  The total connected kVA supplied
- $P_{ci}$  Contracted active power at load point i

#### 44.2.3.2 Load Point Frequency and Expectancy Indices

- ACIF:** Average Customer Interruption Frequency
- ACIT:** Average Customer Interruption Time
- LPIF:** Load Point Interruption Frequency
- LPIT:** Load Point Interruption Time
- LPIC:** Load Point Interruption Costs
- AID:** Average Interruption Duration
- TCIF:** Total Customer Interruption Frequency
- TCIT:** Total Customer Interruption Time
- TPCONTIF:** Total Contracted power Interruption Frequency
- TPCONTIT:** Total Contracted power Interruption Time

These indices are defined as follows:

$$ACIF_i = \sum_k Fr_k \cdot frac_{i,k} \quad \text{Unit : } 1/a \quad (44.7)$$

$$ACIT_i = \sum_k 8760 \cdot Pr_k \cdot frac_{i,k} \quad \text{Unit : } h/a \quad (44.8)$$

$$LPIF_i = \sum_k Fr_k \quad \text{Unit : } 1/a \quad (44.9)$$

$$LPIT_i = \sum_k 8760 \cdot Pr_k \quad \text{Unit : } h/a \quad (44.10)$$

---

**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.

---

$$AID_i = \frac{ACIT_i}{ACIF_i} \quad (44.11)$$

$$TCIF_i = ACIF_i \cdot C_i \quad \text{Unit : } C/a \quad (44.12)$$

$$TCIT_i = ACIT_i \cdot C_i \quad \text{Unit : } Ch/a \quad (44.13)$$

$$TPCONTIF_i = \sum_k Fr_k \cdot frac_{i,k} \cdot P_{c_i} \quad \text{Unit : } MW/a \quad (44.14)$$

$$TPCONTIT_i = \sum_k 8760 \cdot Pr_k \cdot frac_{i,k} \cdot P_{c_i} \quad \text{Unit : } MWh/a \quad (44.15)$$

where

$i$  is the load point index

$k$  is the contingency index

$frac_{i,k}$  is the fraction of the load which is lost at load point  $i$ , for contingency  $k$ .

For unsupplied loads, or for loads that are shed completely,  $frac_{i,k} = 1.0$ .

For loads that are partially shed,  $0.0 <= frac_{i,k} < 1.0$ .

#### 44.2.3.3 System Indices

**SAIFI** *System Average Interruption Frequency Index*, in units of [1/C/a], indicates how often the average customer experiences a sustained interruption during the period specified in the calculation.

**SAIFI\_P** *Average Interruption Frequency (Contracted Power)*, in units of [1/a], indicates how often there are contracted power interruptions during the period of the calculation.

**CAIFI** *Customer Average Interruption Frequency Index*, in units of [1/A/a], 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.

**ASIFI** *Average System Interruption Frequency Index*, in units of [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

**SAIDI** *System Average Interruption Duration Index*, in units of [h/C/a], 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\_P** *Average Interruption Duration (Contracted Power)*, in units of [h/a], indicates the total duration of contracted power interruptions during the period of the calculation.

**CAIDI** *Customer Average Interruption Duration Index*, in units of [H], is the mean time to restore service.

**ASIDI** *Average System Interruption Duration Index*, in units of [h/a], is the equivalent of SAIDI but based on load, rather than customers affected.

**ASAI** *Average Service Availability Index*, this represents the fraction of time that a customer is connected during the defined calculation period.

**ASUI** *Average Service Unavailability Index*, is the probability of having all loads supplied.

**MAIFI** *Momentary Average Interruption Frequency Index*, in units of [1/Ca], evaluates the average frequency of momentary interruptions. The calculation is described in the IEEE Standard 1366 'IEEE Guide for Electric Power Distribution Reliability Indices'.

$$SAIFI = \frac{\sum ACIF_i \cdot C_i}{\sum C_i} \quad \text{Unit : } 1/C/a \quad (44.16)$$

$$SAIFI\_P = \frac{\sum TPCONTIF_i}{\sum PCONTRACT_i} \quad \text{Unit : } 1/a \quad (44.17)$$

$$CAIFI = \frac{\sum ACIF_i \cdot C_i}{\sum A_i} \quad \text{Unit : } 1/A/a \quad (44.18)$$

$$SAIDI = \frac{\sum ACIT_i \cdot C_i}{\sum C_i} \quad \text{Unit : } h/C/a \quad (44.19)$$

$$SAIDI\_P = \frac{\sum TPCONTIT_i}{\sum PCONTRACT_i} \quad \text{Unit : } h/a \quad (44.20)$$

$$CAIDI = \frac{SAIDI}{SAIFI} \quad \text{Unit : } h \quad (44.21)$$

$$ASUI = \frac{\sum ACIT_i \cdot C_i}{8760 \cdot \sum C_i} \quad (44.22)$$

$$ASAII = 1 - ASUI \quad (44.23)$$

$$ASIDI = \frac{\sum (r_m * L_m)}{L_T} \quad \text{Unit : } h/a \quad (44.24)$$

$$ASIFI = \frac{\sum L_m}{L_T} \quad \text{Unit : } 1/a \quad (44.25)$$

$$MAIFI = \frac{\sum IM_i \cdot N_{mi}}{\sum N_i} \quad (44.26)$$

#### 44.2.3.4 Load Point Energy Indices

**LPENS:** Load Point Energy Not Supplied

**LPES:** Load Point Energy Shed

These indices are defined as follows:

$$LPENS_i = ACIT_i \cdot (\widehat{Pd}_i + \widehat{Ps}_i) \quad \text{in } MWh/a \quad (44.27)$$

$$LPES_i = ACIT_i \cdot \widehat{Ps}_i \quad \text{in } MWh/a \quad (44.28)$$

Where

$Pd_i$  is the weighted average amount of power disconnected

$Ps_i$  is the weighted average amount of power shed at load point i.

#### 44.2.3.5 Indices for Busbars/Terminals

**AID:** Average Interruption Duration [H]

**AIF:** Yearly Interruption Frequency [1/y]

**AIT:** Yearly Interruption Time [h/y]

#### 44.2.3.6 System Energy Indices

**ENS** *Energy Not Supplied*, in units of [MWh/a], is the total amount of energy on average not delivered to the system loads.

**SES** *System Energy Shed*, in units of [MWh/a], is the total amount of energy on average expected to be shed in the system.

**AENS** *Average Energy Not Supplied*, in units of [MWh/Ca], is the average amount of energy not supplied, for all customers.

**ACCI** *Average Customer Curtailment Index*, in units of [MWh/Ca], is the average amount of energy not supplied, for all affected customers.

$$ENS = \sum LPENS_i \quad \text{in } MWh/a \quad (44.29)$$

$$SES = \sum LPES_i \quad \text{in } MWh/a \quad (44.30)$$

$$AENS = \frac{ENS}{\sum C_i} \quad \text{in } MWh/Ca \quad (44.31)$$

$$ACCI = \frac{ENS}{\sum A_i} \quad \text{in } MWh/Ca \quad (44.32)$$

#### 44.2.3.7 Load Point Interruption Cost

**LPEIC** is defined as

$$LPEIC_i = \sum LPEIC_{i,k} \quad \text{in } \$/a \quad (44.33)$$

where

$$LPEIC_{i,k}$$

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  $k$ . The interruption costs are calculated differently for different cost functions. All cost functions express the costs as a function of the interruption duration. 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.

#### 44.2.3.8 System Interruption Costs

**EIC** *Expected Interruption Cost*, in units of [M\$/y], is the total expected interruption cost.

**IEAR** *Interrupted Energy Assessment Rate*, in units of [\$/kWh], is the total expected interruption cost per not supplied kWh.

$$EIC = \sum LPEIC_i \quad \text{in } M\$/a \quad (44.34)$$

$$IEAR = \frac{EIC}{ENS} \quad \text{in } \$/kWh \quad (44.35)$$

#### 44.2.4 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 44.2.3.

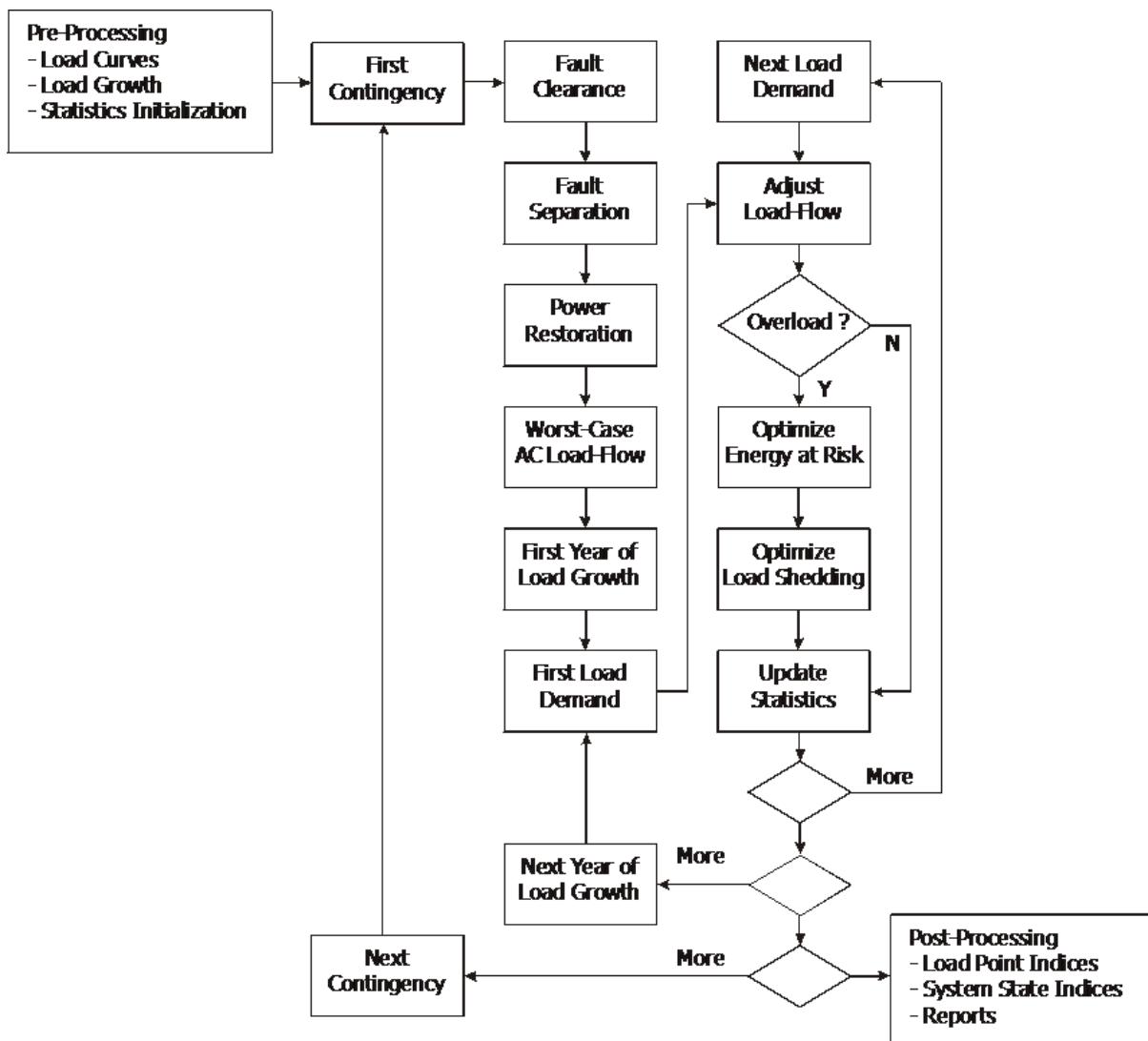


Figure 44.2.3: Overview Flow Diagram for Reliability Assessment by State Enumeration

After the State Enumeration is complete, each simulated system state can be viewed using the 'tracing tool' on the Reliability Toolbar, see Section 45.2 for more information.

## 44.3 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 chapter discusses the following procedures:

- How to Define Stochastic Failure and Repair Models.
- How to Create Feeders for Reliability Assessment.
- How to Configure Switches for the Reliability Assessment.
- Load Modelling for Reliability Assessment.
- Considering Multiple System Demand Levels.
- Defining Fault Clearance Based on Protection Device Location.
- How to Consider Planned Maintenance.

- Specifying Individual Component Constraints.

### 44.3.1 How to Define Stochastic Failure and Repair models

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.
- Line/Cable Stochastic Model.
- Transformer Stochastic Model.
- Distribution Transformer Stochastic Model for MV Loads
- Generator Stochastic Model
- Common Mode Stochastic Model.
- Protection/Switch Failure Model.
- Double Earth Fault Failure Model.

This section describes each of these Stochastic Models and the procedure for defining them.

#### 44.3.1.1 Busbar/Terminal Stochastic Model (StoTypbar)

It is possible to define a Stochastic Model for every busbar and terminal 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 busbars/terminals then you should define it through the object type. Alternatively, if you want to use a Stochastic Model for only one element, then you should define it through the element *Reliability* page.

##### Busbar/Terminal type definition

You can use Stochastic Models defined through types and elements together as required - the element definition always overrides the type definition.

To define a Stochastic Model for a busbar type follow these steps:

1. Open the dialog for the busbar type and select the Reliability tab.
2. Using the 'Stochastic Model' selection control click the black triangle and select the option 'New project type'. The dialog for the 'Bar Type Failures' will appear.
3. Enter the failure data *for the busbar* and the failure data *per connection*. Note that the probability of the busbar failure is the sum of these two failure frequencies. For example a busbar with 3 connections, a failure frequency *for the busbar* of 0.002 and a failure frequency of 0.005 *per connection* will have a total probability of failure of  $0.002 + 3 \cdot 0.005 = 0.017$ .
4. Enter the mean repair duration.
5. Press **OK** twice to return to the element dialog.

##### Busbar/Terminal element definition

To define a Stochastic Model for a busbar element follow these steps:

1. Open the dialog for the busbar *element* and navigate to the Reliability tab.
2. Using the 'Element model' selection control click the black triangle and select the option 'New project type'. The dialog for the 'Bar Type Failures' will appear.

3. Enter the failure data and repair time data as described above for the busbar type.
4. Press **OK** to close the element dialog.

---

**Note:** If you define a stochastic element model for a busbar/terminal that also has a stochastic type model within its corresponding type, the element model overrules the type model.

---

#### 44.3.1.2 Line/Cable Stochastic Model (StoTypline)

It is possible to define a Stochastic Model for every line or cable 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. Using the 'Stochastic Model' selection control click the black triangle and select the option 'New project type'. The dialog for the 'Line Type Failures' will appear.
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 index.
6. Press **OK** twice to return to the element dialog.

##### Cable element definition

To define a Stochastic Model for a line or cable element follow these steps:

1. Open the dialog for the line *element* and navigate to the Reliability tab.
2. Using the 'element model' selection control click the black triangle and select the option 'New project type'. The dialog for the 'Line Type Failures' will appear.
3. Enter the failure data and repair time data as described above for the line type.
4. Press **OK** to return to the element dialog.

#### 44.3.1.3 Transformer Stochastic Model (StoTyptrf)

It is possible to define a Stochastic Model for every transformer 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 then you should define it through the object type reliability page. Alternatively, if you want to use a Stochastic Model for only one transformer 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. Using the 'Stochastic Model' selection control click the black triangle and select the option 'New project type'. The dialog for the 'Transformer Type Failures' will appear.
3. Enter the failure frequency data (1/a).
4. Enter the mean repair duration in hours.
5. Press **OK** twice to return to the element dialog.

#### Transformer element definition

To define a Stochastic Model for a transformer *element* follow these steps:

1. Open the dialog for the transformer *element* and select the Reliability tab.
2. Using the 'element model' selection control click the black triangle and select the option 'New project type'. The dialog for the 'Transformer Type Failures' will appear.
3. Enter the failure data and repair time data as described above for the transformer type.
4. Press **OK** to return to the element dialog.

#### 44.3.1.4 Distribution Transformer Stochastic Model for MV Loads

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* or *type* and select the Reliability tab. As the failure model is based on the transformer (StoTyptrf), the following steps are equivalent to the ones described in Section [44.3.1.3](#).

#### 44.3.1.5 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* and *Generation Adequacy*. For further information refer to Section [46.3.1](#). The Stochastic Model can be defined using the 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 [%]:** Percentage of the nominal power available
- **Probability [%]:** Probability that this state is valid (the sum of all probabilities must 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

### 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. Using the 'Stochastic Model', click the black triangle and select the option 'Select...'. The dialog for the 'Equipment Types' project library 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. Press **OK** to return to the element dialog.

#### 44.3.1.6 Common Mode Stochastic Model

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 *Define* → *Common Mode Failure*.
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 Rows'.
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.

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.

#### 44.3.1.7 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 'Edit Devices'.
2. On the Reliability tab 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.

---

#### 44.3.1.8 Double Earth Faults

A double earth fault in *PowerFactory* is defined as follows: a single earth fault on a component followed by a second simultaneous earth fault on another component.

A double earth fault might occur after voltage rises on healthy phases on a feeder following a single phase to earth fault on the feeder, causes a second phase to earth fault on the same feeder.

Double earth faults occur on lines, transformers (2 Winding and 3 Winding transformers) and busbars, and *PowerFactory* supports adding the conditional probability data for double earth faults for Stochastic Models of these components. The reliability calculation automatically generates a contingency event for every double earth fault that meets the following conditions:

- Both objects are in the same part of the network (supplied by the same transformers).
- The star point of the transformers that supply that part of the network is isolated or compensated (both star point grounded and Peterson Coil enabled).

- The frequency of single earth faults of the first object is  $> 0$
- The probability of double earth fault of the second object is  $> 0$ .

The frequency for single earth faults and the probability of the second earth fault data can be entered on the 'Earth Fault' page of every Stochastic Model. Follow these steps to enter data for a Line Stochastic Model:

1. Open the Stochastic Failure Model for the line (either through the reliability page of the line type or the line elements).
2. Select the Earth Fault page.
3. Enable the option 'Model Earth Faults'
4. Enter the data for the frequency of single earth faults
5. Enter the data for the conditional probability of a second earth fault
6. Enter the Repair duration in hours.
7. Close the Stochastic Model.

---

**Note:** The double earth fault is a conditional probability. Therefore, the probability of one occurring in the network is the probability of an earth fault on component A \* probability of an double earth fault on component B

---

#### 44.3.2 How to Create Feeders for Reliability Calculation

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 *Define → Feeder*; or
- For fast creation of multiple feeders, right click the bus that the feeder/s are to be connected to and select the option *Define → Feeder*. More information on feeders and feeder creation can be found in Chapter 15: Grouping Objects, Section 15.5(Feeders).

When executing the Reliability Assessment in distribution networks with a focus on optimal power restoration, the meshes within feeders are restricted to be of the following kinds:

##### 1. Mesh within the feeder

In this case, the feeder is supplied from one point and the mesh is within the feeder itself.

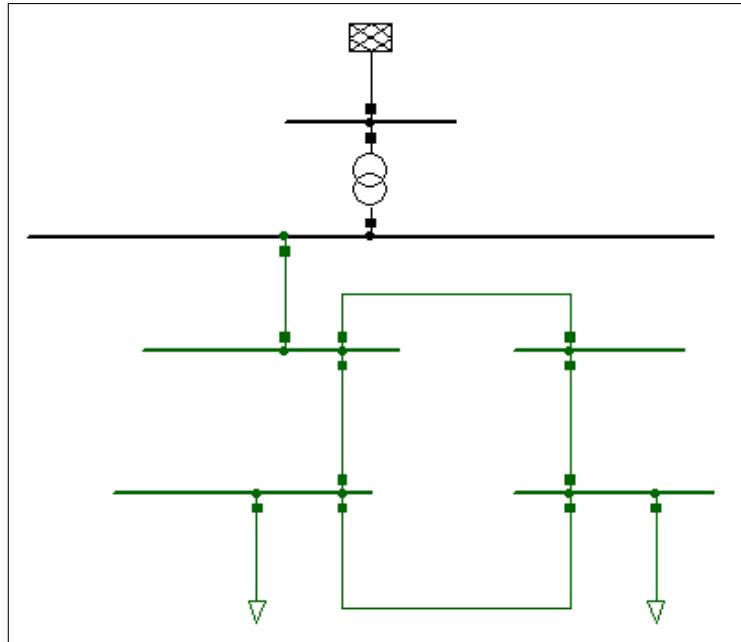


Figure 44.3.1: Mesh within feeder

## 2. Two feeders, starting from the same terminal, are connected

In this case, the feeders are connected and therefore result in a mesh as shown in figure 44.3.2.

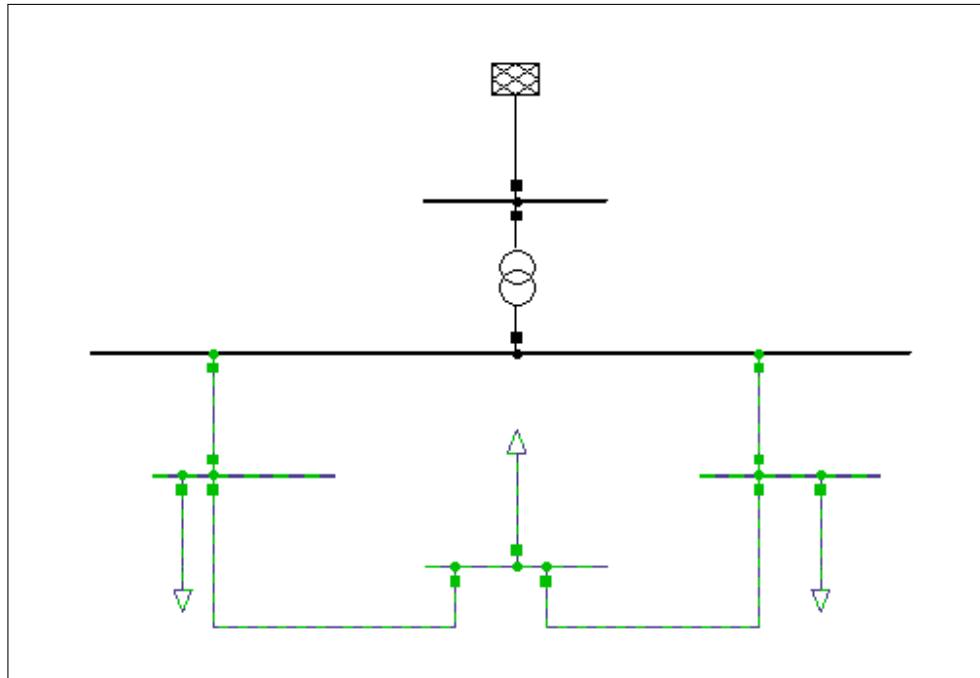


Figure 44.3.2: Mesh containing two feeders starting from the same terminal

### 44.3.3 Configuring Switches for the Reliability Calculation

A critical component of the Failure Effect Analysis (FEA), is the behaviour of the switches in the network model. Switches in *PowerFactory* are classified into four different categories:

- Circuit Breakers; Typically these are automatic and controlled by relays and through remote communications. They are used for clearing faults and for closing 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.

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. 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 'edit 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 44.4.1: How to run the Reliability Assessment. 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.
  - From Branch to Node; If this option is selected, the switch will only be used to restore power if the post restoration power flow is in the direction from Branch to Node. 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 post restoration power flow is in the direction from Node to Branch. The switch will not be used for power restoration in the opposite direction.
  - 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 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.

The complete switching times depend on the following settings:

- Switching procedure for fault separation / power restoration (page FEA of reliability command)
- Access time (of terminal for global terminals, of station for terminals inside stations)
- Local access time (of terminal for global terminals, of station for terminals inside stations)
- Time to open remote controlled switches (page FEA of reliability command) for remote controlled switches

- Time to actuate switch (page reliability of switch) for any other switch.

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.
  - \* Any other switch: Access Time + Time to actuate switch
- Sequential (previous switching time is considered):
  - \* Remote controlled switches: Are actuated at the time entered in the reliability command.
  - \* Any other switch:
    - If a manual switch was actuated before:
      - → If the switch is located in the same station as the switch previously actuated: Last switch time + Time to actuate switch
      - If switch is in a different substation: Last switch time + Local access time + Time to actuate switch
    - Last switch time + Access time + Time to actuate switch

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.

---

#### 44.3.4 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.

##### 44.3.4.1 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. To do this:

1. Open the dialog for the target load element.
2. Select the Reliability page.
3. In the 'Number of connected customers' field, enter the number of customers that this load represents.
4. Repeat this process for each load in the system you are modelling.

##### 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.

#### 44.3.4.2 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.

---

#### 44.3.5 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 18.2.5 (Vector Characteristics with Time Scales).

#### 44.3.6 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).

#### 44.3.7 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).

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 tab. "Fault Clearance: Consider as switch with protection device"

---

#### 44.3.8 How to Consider Planned Maintenance

The *PowerFactory* reliability calculation supports the definition and automatic inclusion of planned network maintenance. Maintenance is implemented with a planned outage object. These objects are found within the 'Outages' sub-folder within the project 'Operational Library'. The following steps describe the procedure for creating a planned outage:

1. On the single line diagram (or within the Data Manager), select the object (or objects) that you would like to define an outage for.
2. Right-click the selected object/s and from the menu that appears choose the option *Define* → *Planned Outage*. The dialog box for the planned outage will appear.
3. Using the Start Time selection control '...', enter the time that the outage begins.
4. Using the End Time selection control '...', enter the time that the outage ends.
5. Optional: Adjust the Outage Type. Typically you would leave this on the default 'Outage of Element' option, but if you wanted to model a generator derating, then you would choose the 'Generator Derating' option.

---

**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  $6 \cdot 3 = 18$  contingency cases.

---

#### 44.3.9 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 tab 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.

#### 44.3.10 Consider switching rules

Reliability Analysis in *PowerFactory* allows the user to consider predefined switching rules within substations according to chapter 11.2.7.4. Switching-rules are executed directly after protection operation.

### 44.4 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.

#### 44.4.1 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 *Calculate* → *Reliability Assessment* or → *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.

##### 44.4.1.1 Basic Options

The following options are available on the Basic Options page Reliability Assessment Command dialog.

###### Load Flow

This button is a link to the load-flow calculation command used for the analysis. The load demand is calculated using this load-flow. In addition, its settings are used for the constraint evaluation load-flows.

###### 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

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 (*Select*), 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 checkboxes:

- Busbars/Terminals; This flag should be enabled for *PowerFactory* to create Busbar and terminal contingencies.
- Lines/Cables; This flag should be enabled for *PowerFactory* to create Line/Cable contingencies.
- Transformers; This flag should be enabled for *PowerFactory* to create transformer contingencies.
- Generators; This flag should be enabled for *PowerFactory* to create generator contingencies (Load flow analysis only).
- Common Mode; This flag should be enabled for *PowerFactory* to create Common Mode contingencies. See section [44.3.1.6](#) (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 [44.3.1.8](#) (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 [44.3.1.7](#) (Protection/Switch Failures) for more information.
- Spurious protection operation; as explained in section [44.3.1.7](#).
- Backup protection maloperation; as explained in section [44.3.1.7](#).

#### 44.4.1.2 Outputs

The following options are available on the *Outputs* tab 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.

##### Perform Evaluation of Results File

The Reliability Analysis automatically writes all simulation results to a results object specified above. After completing the Reliability Calculation, *PowerFactory* automatically evaluates the results object to compute the reliability indices. This button allows you to re-evaluate a results file that has previously been created by this or another reliability calculation command. 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.

### Report

Displays the form used for the output report. Report settings can be inspected and the format selected by clicking on the → button.

### 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.

#### 44.4.1.3 Protection

##### Fault Clearance Breakers

- **Use all circuit breakers:** all switches in the system whose *Usage* is set to *Circuit Breaker* can be used for fault clearance.
- **Use only circuit breakers 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.

##### Create Contingencies

These settings are the same as in “Automatic Contingency Definition”, described in section [44.4.1.1](#). For convenience they are displayed within this tab as well.

#### 44.4.1.4 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.

### Power restoration

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 and a directly-bordering feeder)
- Enabled with load transfer (tie open points can be moved between a bordering feeder and a second-level bordering feeder)

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.

### Enhanced restoration

If this checkbox is enabled, the restoration uses a more precise algorithm to solve load flow convergence issues during the restoration process. Setting this option can decrease the performance of the Reliability assessment.

#### Consider a possible backward recovery of a feeder in a primary substation

Whenever the start point of a feeder is de-energised after a fault ( e.g. on a HV/MV-transformer), backward recovery can be used to resupply this feeder. Backward recovery allows the restoration through a substation. It is started in a case where the standard recovery could not resupply all loads after the first stage of the recovery, e.g. because of constraint violations or because of de-energised isolated feeders. 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, especially, for example, for dedicated low-load feeders between substations or isolated feeders (see figure below).

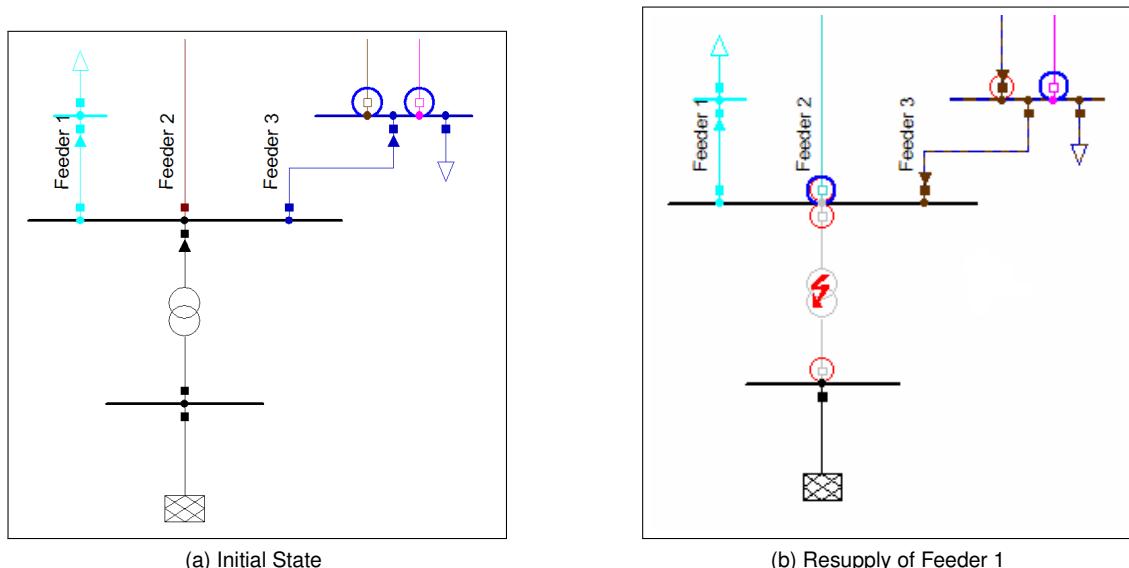


Figure 44.4.1: Example of Backward Recovery

The Backward recovery option, located within the “Restoration” Option, can be set as follows.

- Do not allow: no backward recovery will be used for restoration
- Allow but prefer standard recovery: backward recovery is allowed, but standard recovery will be preferred for restoration.
- 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.

#### 44.4.1.5 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).

#### 44.4.1.6 Constraints

The settings for global constraints are defined within this page. The options are as follows:

##### 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 tab 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 tab 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 tab 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.

---

**Note:** Voltage constraints are only available when the 'Distribution' analysis option is selected under 'Basic Options'. Thermal constraints are available when either the 'Transmission' or 'Distribution' analysis option is selected.

---

#### 44.4.1.7 Maintenance

This tab allows you to enable or disable the consideration of Maintenance based on the Planned Outages you have defined. See Section [44.3.9](#), 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.

#### 44.4.1.8 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 load states, load distribution states or Load and Generator States according to section [44.3.6](#) 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 printed to the output window and the reliability calculation will stop. Otherwise the following two buttons 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.

#### 44.4.1.9 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 will be saved in the corresponding contingency.

#### Stop calculation if base case is overloaded

If this option is set, the reliability assessment will stop if the base case is overloaded. If not, a user defined threshold can be specified.

#### Calculation of SAIFI/SAIDI according to IEEE 1366

- **Do not consider interruptions shorter than or equal to:** this option enables the possibility to not consider interruptions shorter than a user defined duration for the calculation of SAIFI/SAIDI according to section [44.2.3.3](#).

#### 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:** removes all switch events associated with the contingencies stored inside the command.
- **Delete load events:** removes all load events associated with the contingencies stored inside the command.

#### Failures, correction of forced outage rate

This option performs an automatic correction/normalisation of the reliability indices to allow for the fact that not all unlikely but possible contingencies have been considered in the analysis. For instance, n-3 contingencies have a non-zero probability.

---

**Note:** 'Forced outage' refers to the unplanned removal of a primary component from the system due to one or more failures in the system.

---

#### 44.4.1.10 Parallel Computing

Parallel calculation of the *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.

**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 [22.4](#).

## 44.5 Results of the Reliability Analysis

### 44.5.1 Contribution to Reliability Indices

Contribution means the effects of the calculated contingencies to the reliability indices of all loads or a selection of loads. This chapter describes an optional post processing step, which can be useful to analyse the influence of a particular component or group of components on the calculated reliability indices. This enables the identification of components that can be targeted for upgrade to improve reliability, or to examine the impact of improved switch automation for example. This sub-section describes the post-processing functionality that can be used for these purposes.

To analyse the contributions to the following indices

1. SAIFI,
2. SAIDI,
3. ASIFI,
4. ASIDI,
5. ENS,
6. EIC,

the Reliability Assessment Calculation has to be executed once, or it has to be ensured that the currently activated study case contains results of a previously executed reliability analysis.

- Choose the *Contributions to Reliability Indices* button () in the Reliability Analysis toolbar menu.
- Choose between the contribution to all loads or to a user defined selection of loads. Loads can be selected according to the following groupings.
  - Grids,
  - Feeders,
  - Zones,
  - Areas.
  - and General Selections.
- Furthermore, it is possible to analyse the contribution of network elements to reliability indices of one Load, which can be a General Load, LV-Loads or MV-Load.
- After setting the required set, the user can execute the post process.

---

**Note:** The Contributions to Reliability Indices is a post-processing command. Once calculated the contributions for a selection, it is possible to change the selection or re-run the command for all Loads without executing the Reliability Assessment once again.

---

### 44.5.2 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 or load filter) or according to the diagram colouring. This sub-section describes the first two of these methods. The third method is described in chapter [44.5.2.2](#).

#### 44.5.2.1 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:

- AID: Average Interruption Duration.
- 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. This is demonstrated in Figure 44.5.1

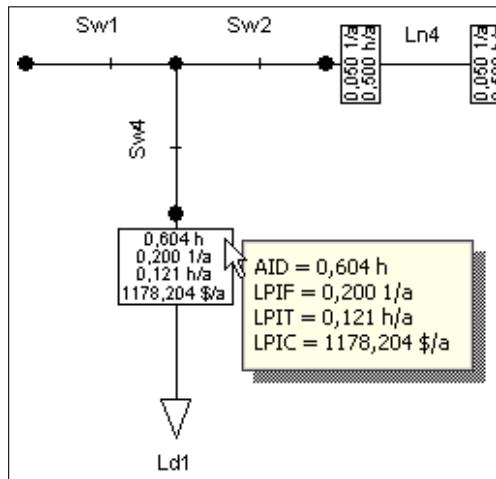


Figure 44.5.1: Single Line Diagram Graphic Showing the Load Point Indices Results

**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)

#### 44.5.2.2 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 colouring is according to results of the whole system or the selection that has been chosen as described in chapter 44.5.1. 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.

### 44.5.3 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.

---

### 44.5.4 Reliability Reports

The Report can be accessed through the *Reliability Reports* button  within the Reliability Analysis toolbar. The Report is based on the whole system or the selection that has been chosen as described in chapter 44.5.1. The report offers the following functionalities.

#### ASCII Report

This report is writing the results into the output window.

- **System Summary:** reports a calculation summary of the Reliability Assessment together with calculated system indices.
- **Load Interruptions:** reports the following indices for all Loads within the selection.
  - TCIT
  - TCIF
  - AID
  - LPENS
  - LPIC
  - ACIF
  - ACIT
- **Node Interruptions:** reports the following indices, focused on nodes.
  - AIT
  - AIF
  - AID
- **Contribution of Component Classes:** reports the contribution of Lines, Cables, Transformers, Terminals, Generators, Common Mode Failures and Double-Earth Faults to the system-indices, that are available for contributions mentioned in 44.5.1.

### **Tabular report of Contributions**

This Report returns the contribution of one single element to the overall system-indices in a tabular form. This contribution is given as absolute value and in per-cent.

# Chapter 45

## 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 45.0.1

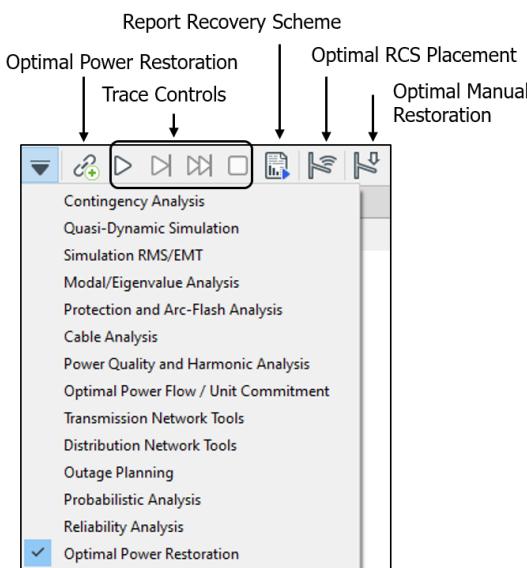


Figure 45.0.1: Optimal Power Restoration Selection

### 45.1 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 44.3.3.

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 45.0.1, or by right-clicking an element and selecting *Calculate* → *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.

### 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 over-loading. 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 45.1.1 shows a simple network containing four loads, several circuit breakers (CB) and disconnectors (DS) and a back-feed switch (BF).

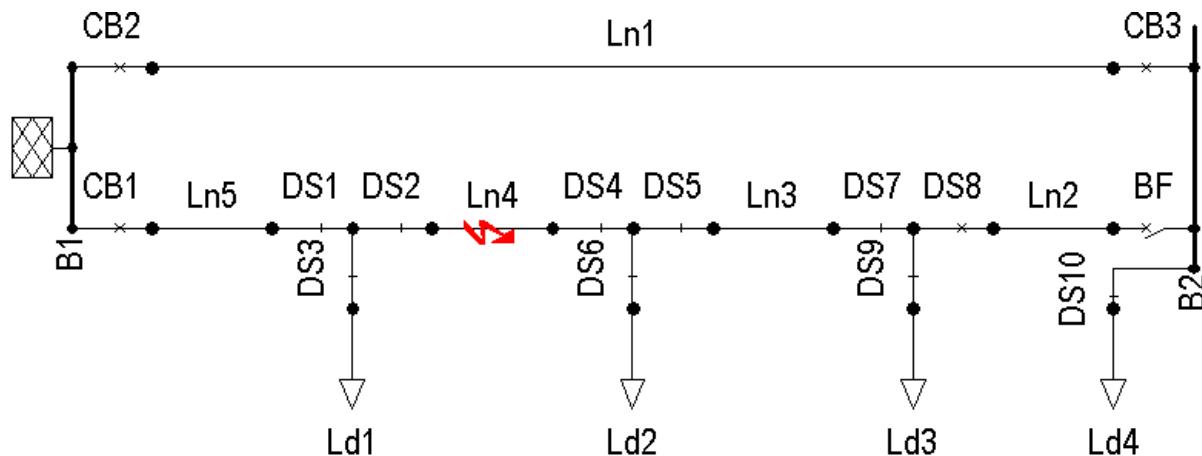


Figure 45.1.1: Short-Circuit on Ln4

### Fault clearance

The area isolated by the fault clearance procedure is called the 'protected area'. Figure 45.1.2 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.

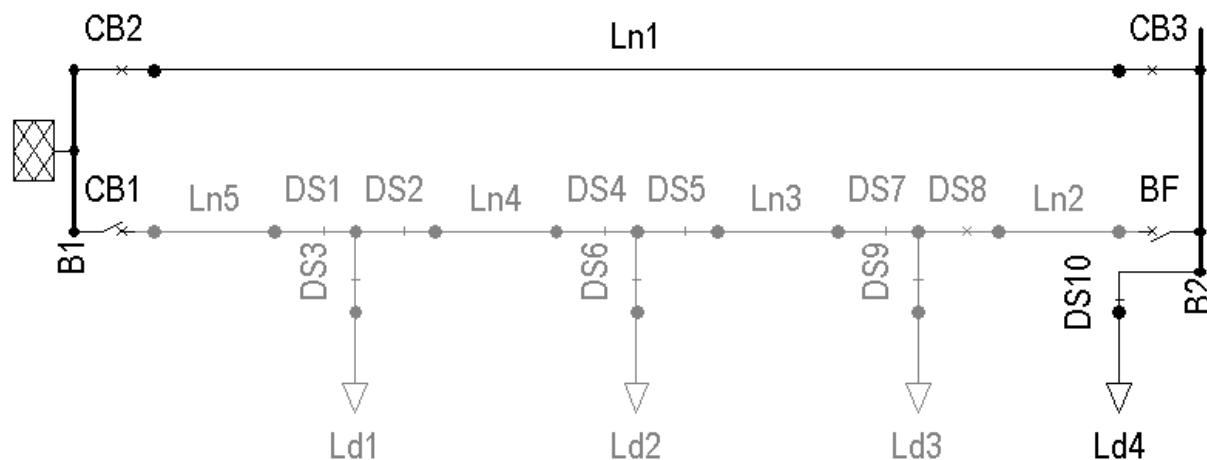


Figure 45.1.2: Protected Area

### Fault Isolation

Figure 45.1.3 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.

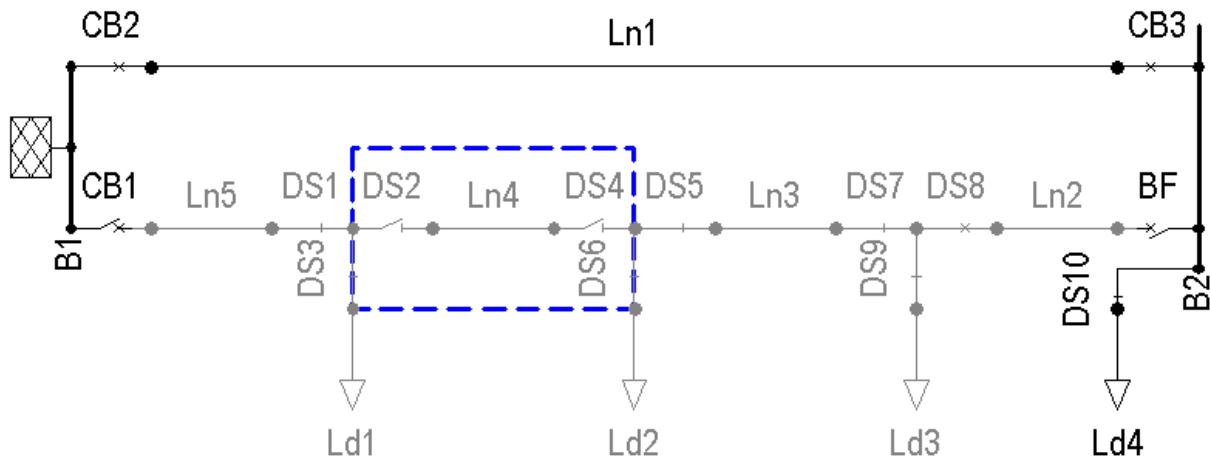


Figure 45.1.3: Separated Area Highlighted

### **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 45.1.4.

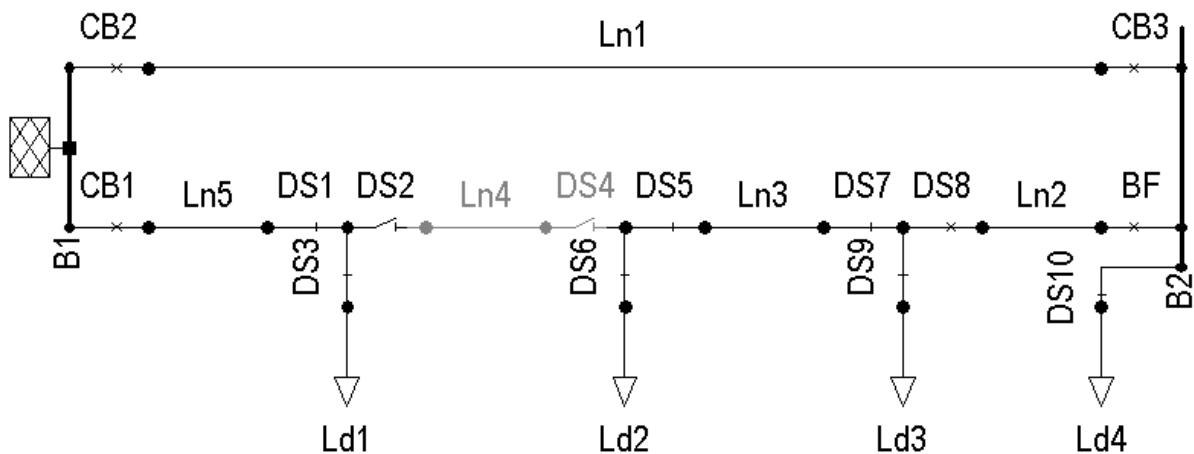


Figure 45.1.4: Power Restoration by Back-Feed Switch BF1 and CB1

### **Overload Alleviation and Load Shedding**

Figure 45.1.5 shows a line overload in the post-fault condition in the example network: line 'Ln1' is loaded to 113%.

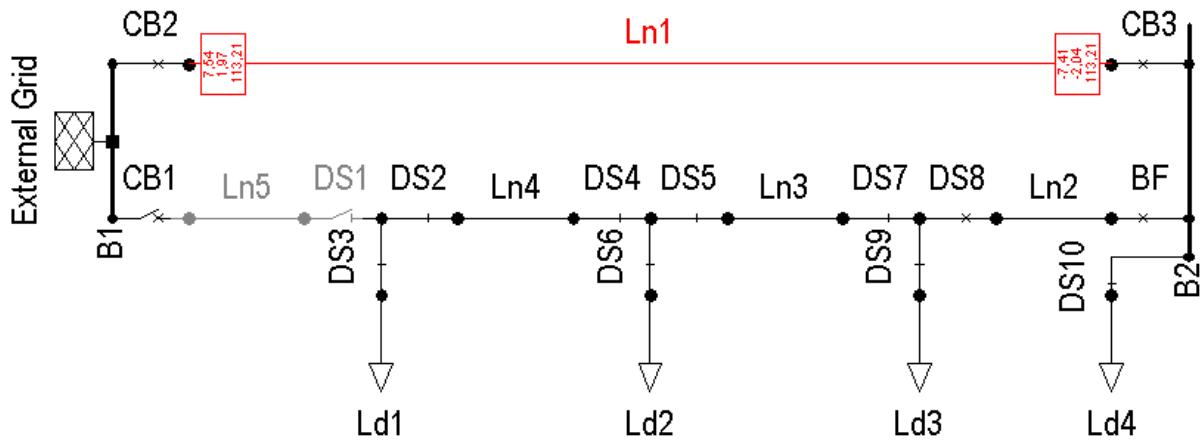


Figure 45.1.5: Overloaded Post-Fault Condition

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.

## 45.2 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.

## 45.3 Optimal RCS Placement

Following a Backbone Calculation (see Section 41.5), an Optimal Remote Control Switch (RCS) Placement can be performed to optimise placement of remote control 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 44: Reliability Assessment, Section 44.3 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 45.0.1 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.

---

### 45.3.1 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.

### 45.3.2 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 control switches and their costs, and depending on the selected objective function, energy not supplied or interruption costs results.

### 45.3.3 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).

### 45.3.4 Example Optimal RCS Calculation

Consider the simple example shown in Figure 45.3.1 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 Control Switch (RCS) on each feeder to minimise the ENS.

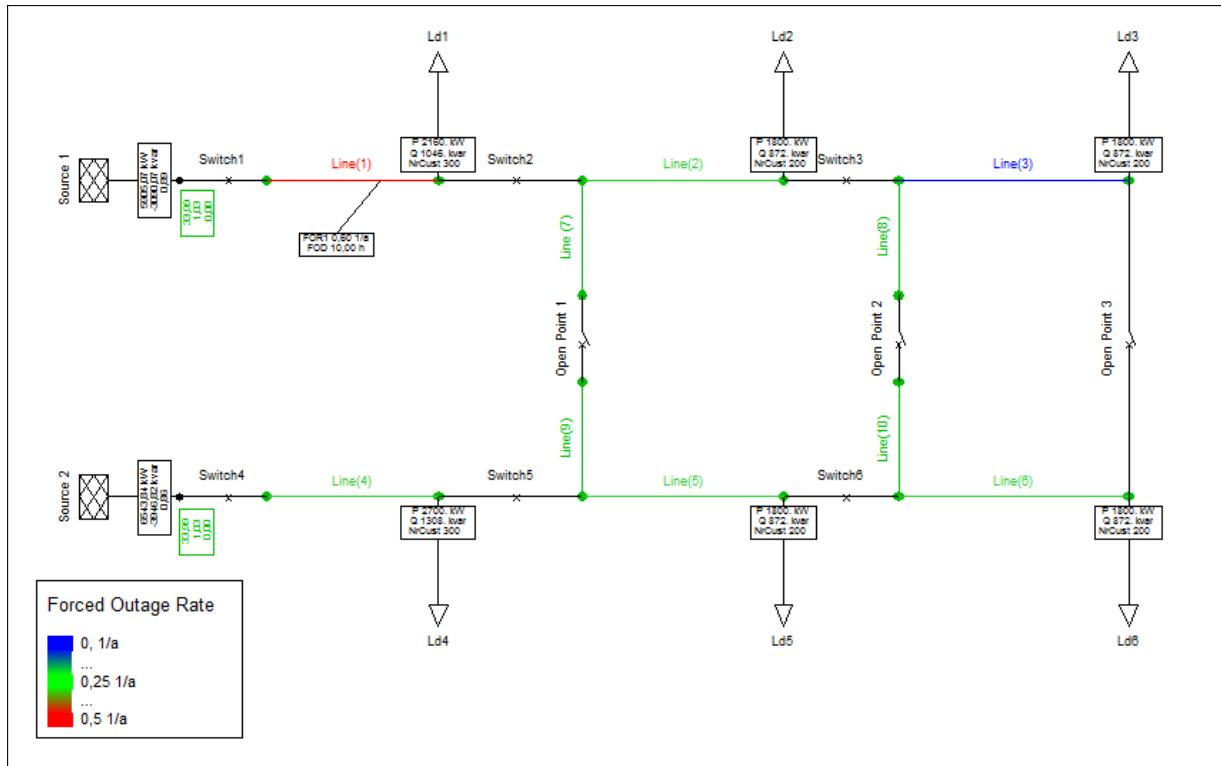


Figure 45.3.1: Example Optimal RCS Model

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 41.5 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 control switches at locations "Switch2" and "Switch5" to minimise the ENS.

## 45.4 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 control 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 control 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 control switches and open manual switches. Level II pockets are areas enclosed by remote control switches, open manual switches and OMR level I switches. Similarly, Level 3 pockets are areas enclosed by remote control switches, open manual switches and OMR switches of level 1 and 2.

#### 45.4.1 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 41.5).
- There must be at least one remote control switch in the network.
- It is recommended to build the network using terminals or secondary substation layouts (*ElmTrfstat*).

#### 45.4.2 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.

### 45.4.3 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 control 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 control 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 control 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).

#### 45.4.4 Definition of the objective function

The aim of the OMR calculation is to minimise the following objective function:

$$\Delta^x = |P_{upReg}^x \cdot F_{upReg}^x - P_{downReg}^x \cdot F_{downReg}^x| \quad (45.1)$$

The members of the above objective function are defined based on the following equalities:

$$P_{upReg}^x = P_{up}^x - P_{up}^{startReg} - \sum P_{down}^{upNStart} \quad (45.2)$$

$$F_{upReg}^x = F_{up}^x - F_{up}^{startReg} - \sum F_{down}^{upNStart} \quad (45.3)$$

$$P_{downReg}^x = P_{down}^x - \sum P_{down}^{downNStart} \quad (45.4)$$

$$F_{downReg}^x = F_{down}^x - \sum F_{down}^{downNStart} \quad (45.5)$$

$$(45.6)$$

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
- $F_{down}^{downNStart}$  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.

#### 45.4.5 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 45.4.1. 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.

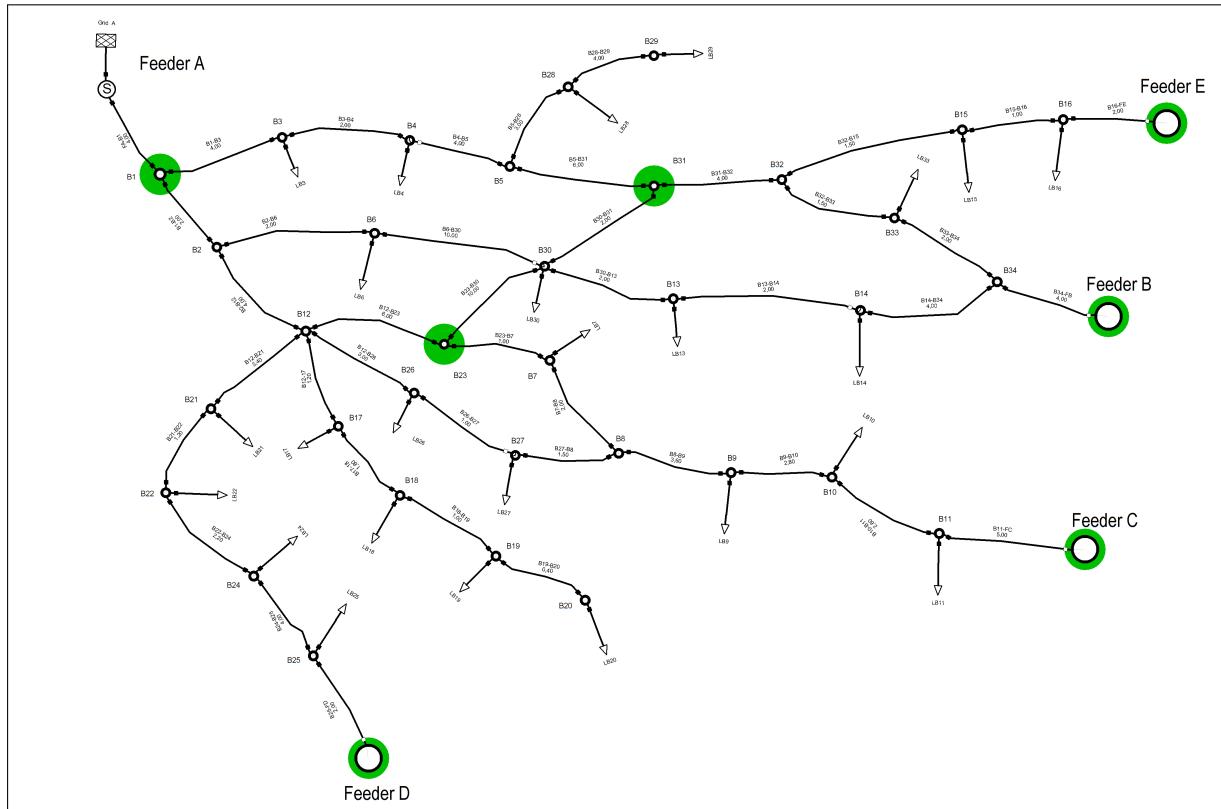


Figure 45.4.1: Generic MV Distribution Network

A substation layout similar to the one shown in Figure 45.4.2 is used for all substations.

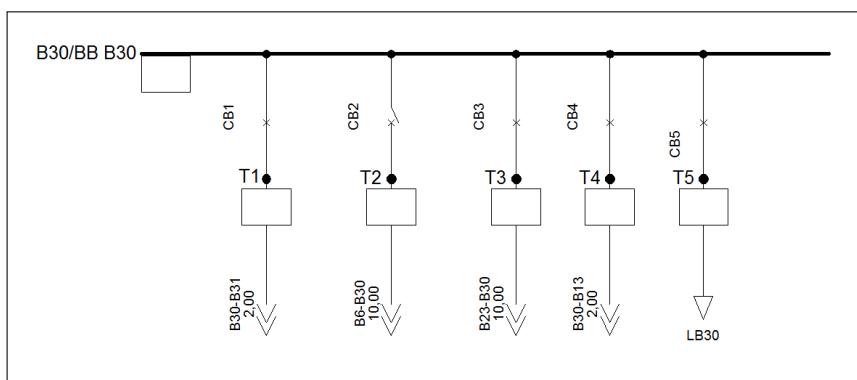


Figure 45.4.2: Generic Substation Single Line Diagram

A backbone calculation (*ComBbone*) for *Feeder A* is performed on this network based on *path load* (see Section 41.5 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 45.4.3.

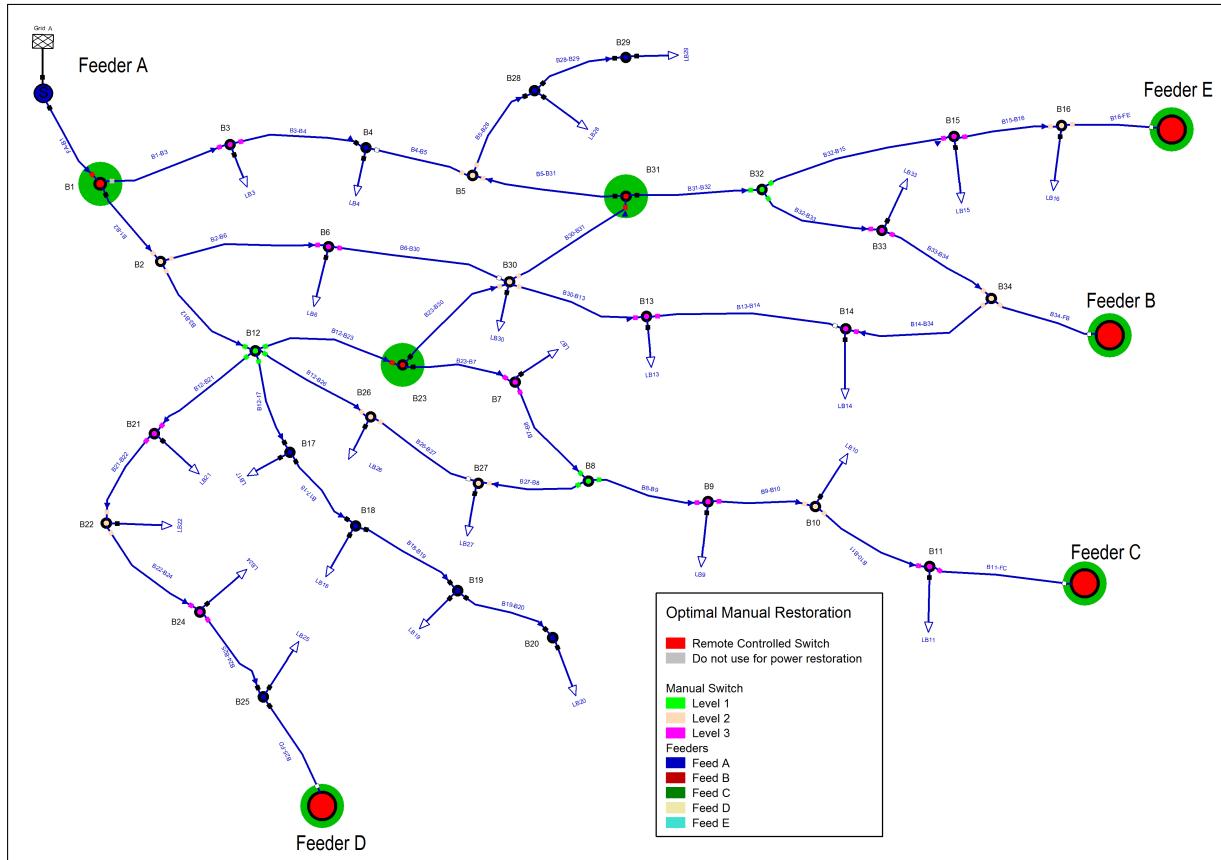


Figure 45.4.3: OMR Calculation Results Shown in the Single Line Diagram using Different Colours

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.

# Chapter 46

## Generation Adequacy Analysis

### 46.1 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)

---

### 46.2 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 46.2.1.

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.

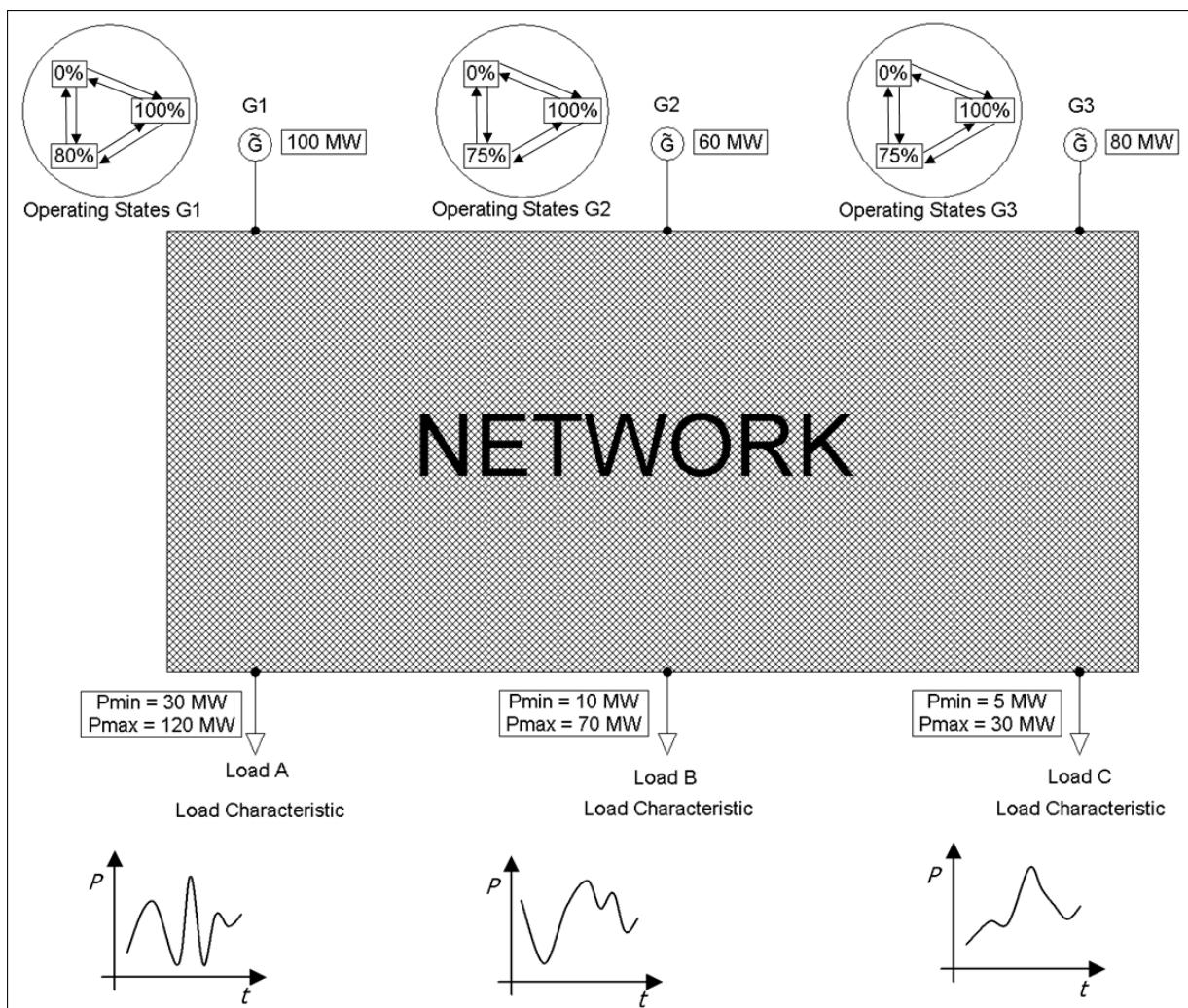


Figure 46.2.1: Generation Adequacy Assessment Illustration

### 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 46.2.1 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 46.1.

$$DNS = \sum Demand - \sum Generation \quad (46.1)$$

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 46.2.1:

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 46.2.1: 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:

$$LOLP = \frac{N_{DNS}}{N} \cdot 100\% \quad (46.2)$$

$$EDNS = \frac{\sum DNS}{N} \quad (46.3)$$

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:

$$LOLP = \frac{2}{6} \cdot 100 = 33,33\% \quad (46.4)$$

$$EDNS = \frac{30}{6} = 5MW \quad (46.5)$$

## 46.3 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.

### 46.3.1 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 46.3.1. An unlimited number of states is possible with each state divided into:

**State** : name of the state

**Availability [% ]**: percentage of the nominal 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 44.3.1.5.

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 %.

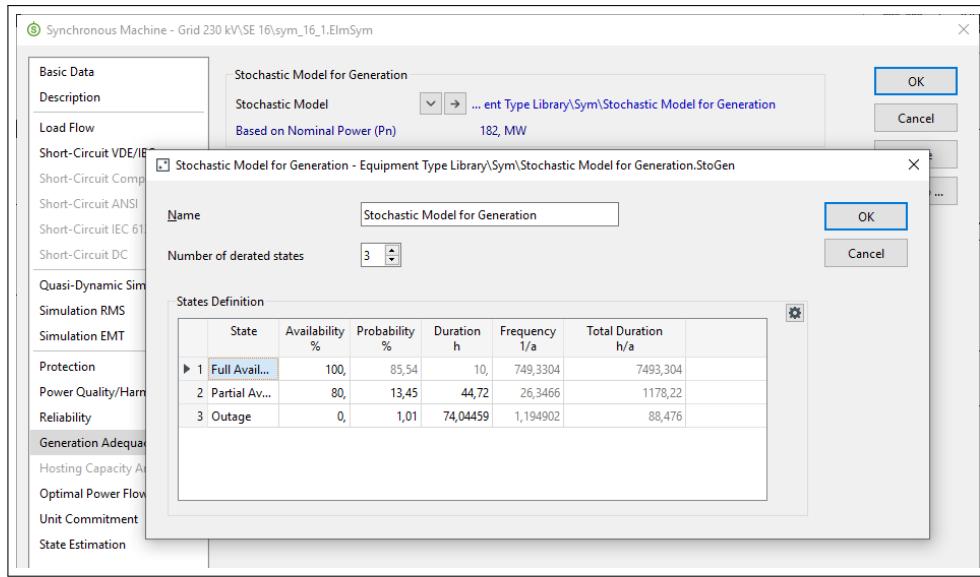


Figure 46.3.1: Stochastic Model for Generation dialog

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 nominal apparent power and  $\cos \theta$  is the nominal power factor.

### 46.3.2 Power Curve Type

The Power Curve Type (*TypPowercurve*) object is used to specify the wind speed (in m/s) vs nominal 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.

### 46.3.3 Meteorological station

If a group of wind generators have a wind speed characteristic that is correlated, it can be represented through the *Meteo Station* object (described in Section 15.6).

Note that when two wind generators are correlated as members of the same *Meteo 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 *Meteo Station*. This wind speed is then

applied to every wind generator in that *Meteo 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.

## 46.4 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.

### 46.4.1 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 [46.3.1](#).

- 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.

#### 46.4.2 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.

#### 46.4.3 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 46.4.1.

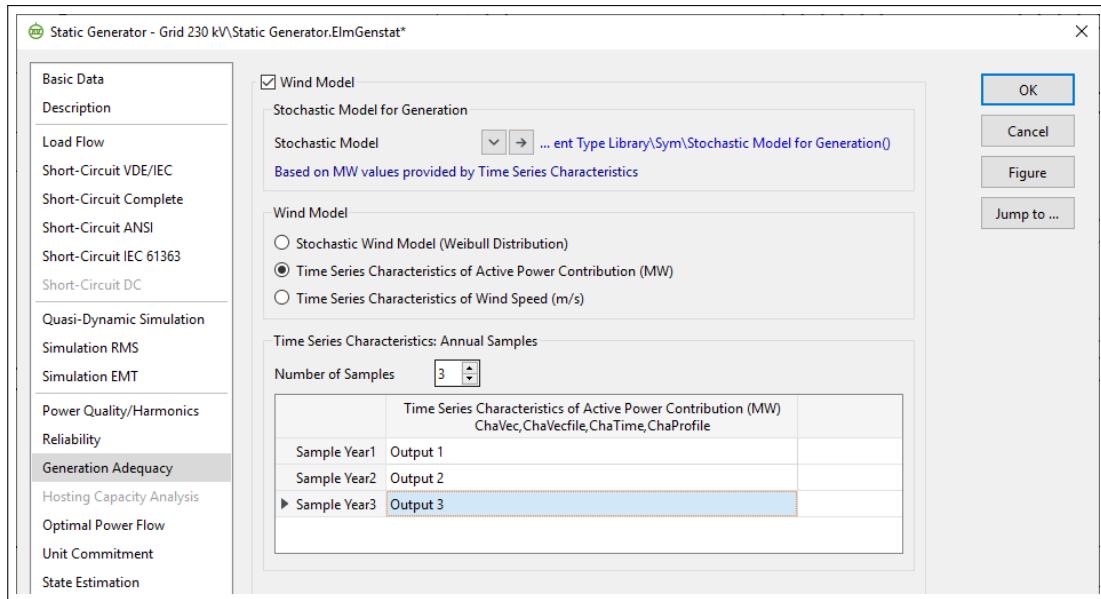


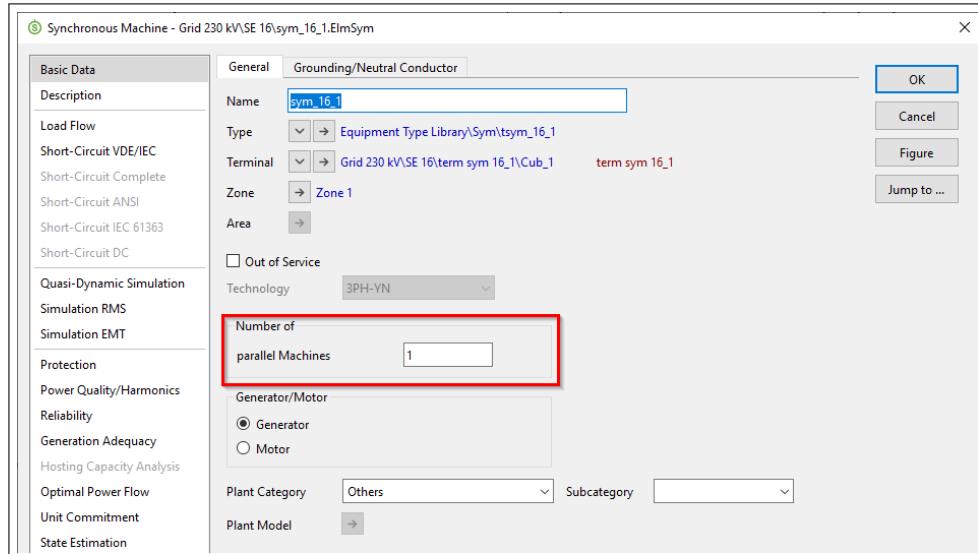
Figure 46.4.1: Wind Model using Wind Output Data

### 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 (*ngeum*), as shown in Figure 46.4.2. 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 46.4.2: Synchronous machine element with the parameter *ngeum* (number of parallel machines highlighted).

#### 46.4.4 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.

## 46.5 Generation Adequacy Analysis toolbar

Once the Generation Adequacy toolbar is selected, the available buttons are shown in Figure 46.5.1.

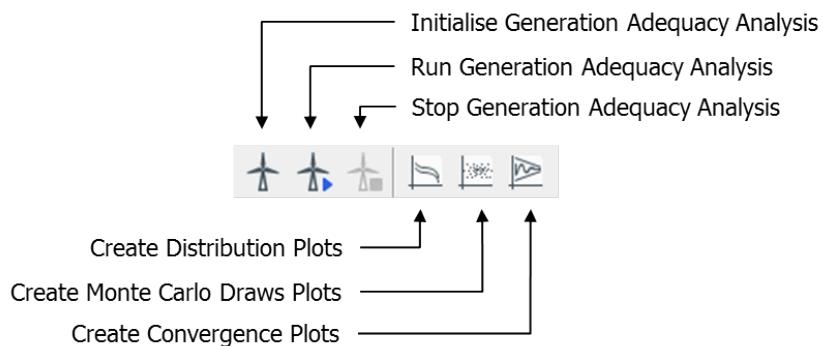


Figure 46.5.1: Generation Adequacy Analysis toolbar buttons

### 46.5.1 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.

#### 46.5.1.1 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 Define → Planned Outage For more information on Planned Outages refer to Chapter 14: Project Library, Section 14.3.7 (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 46.4.3 (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.

#### 46.5.1.2 Output page

- MC Draws: if this option is checked, then *PowerFactory* will automatically create Monte-Carlo Draw plots after the simulation finishes. See Section 46.6 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.

#### 46.5.1.3 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.

### 46.5.2 Run Generation Adequacy command

The *Run Generation Adequacy Analysis* command (*ComGenreI*) 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 46.5.2. On the other hand, if some iterations are already complete, then the calculation can be continued and the dialog appears as shown in Figure 46.5.3.

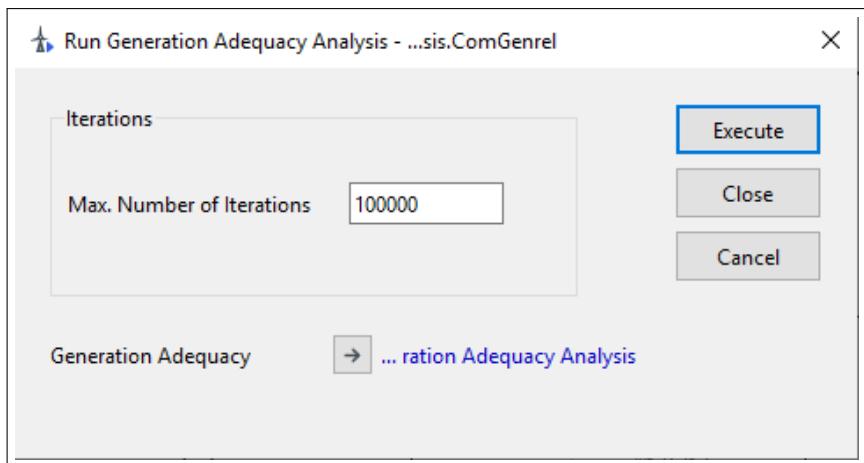


Figure 46.5.2: Run Generation Adequacy command dialog (new simulation)

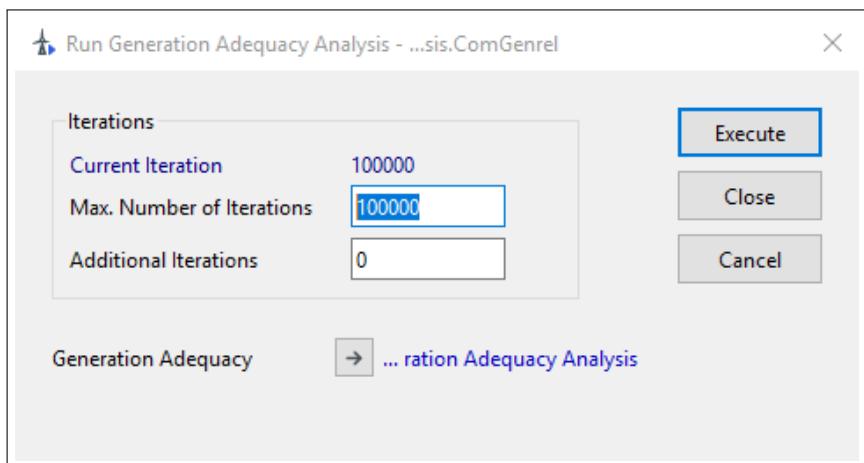


Figure 46.5.3: Run Generation Adequacy command dialog (post simulation)

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.

## 46.6 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.

### 46.6.1 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 46.6.1.

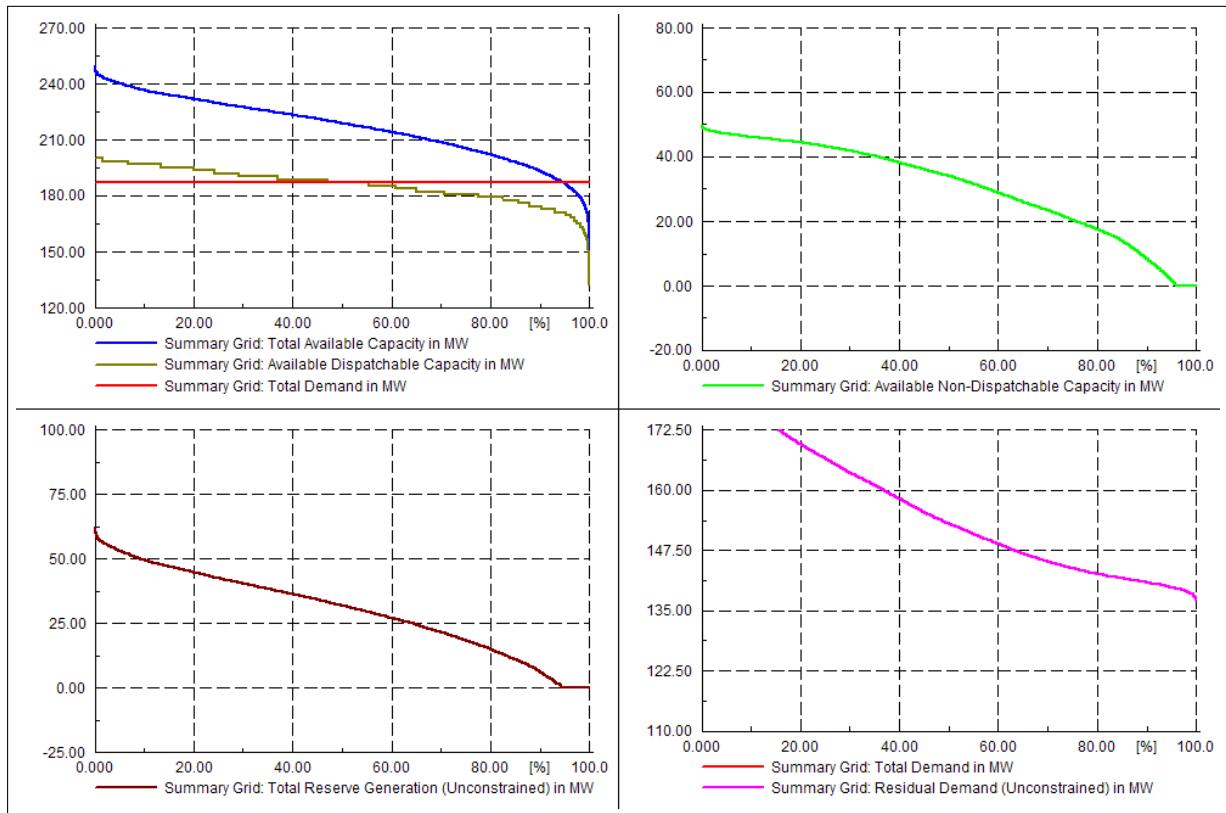


Figure 46.6.1: Distribution (Cumulative Probability) Plots

### 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 46.6.2.

When the demand is variable, then the LOLP index cannot be inferred from the above diagram. Figure 46.6.3 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 46.6.4, the intersection of this curve with the x-axis gives the LOLP index.

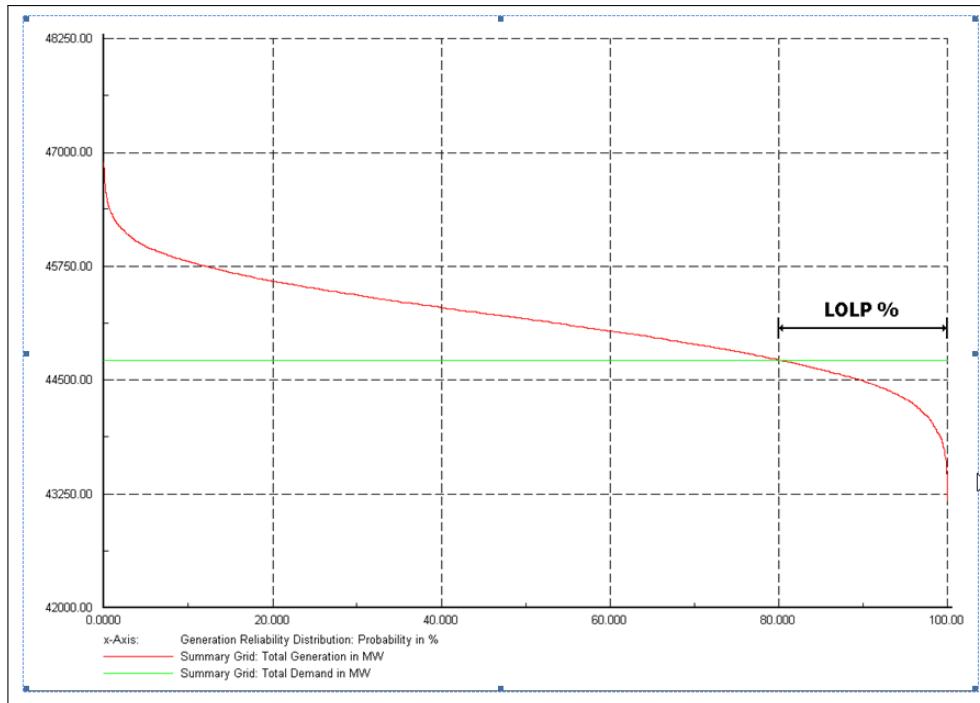


Figure 46.6.2: Inferring the LOLP index directly from the intersection of the Total Generation and Total Demand

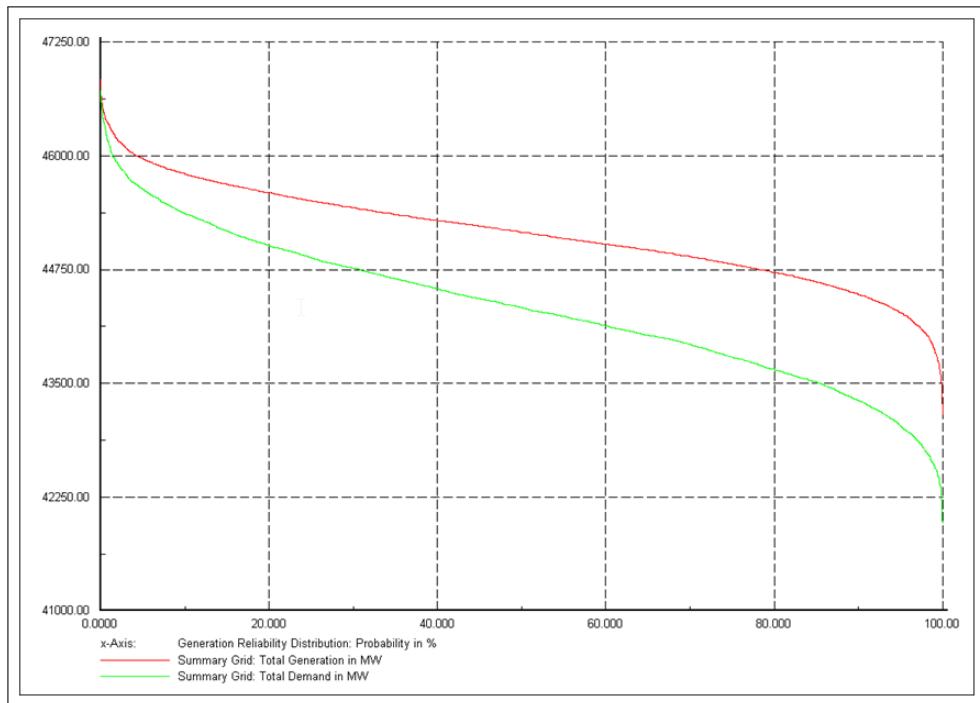


Figure 46.6.3: Variable Demand - distribution of Total Generation and Total Demand

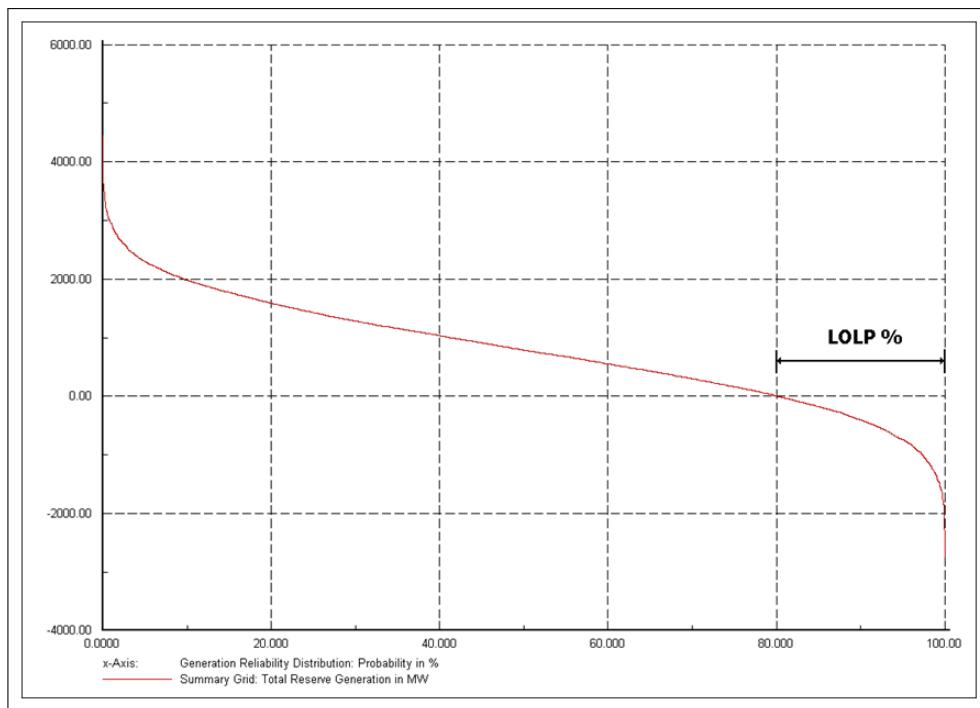


Figure 46.6.4: Total Reserve Generation

## 46.6.2 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 (掣) can be used. This button draws by default four figures as shown in Figure 46.6.5. Each of the data points on the plots represents a single Monte Carlo simulation.

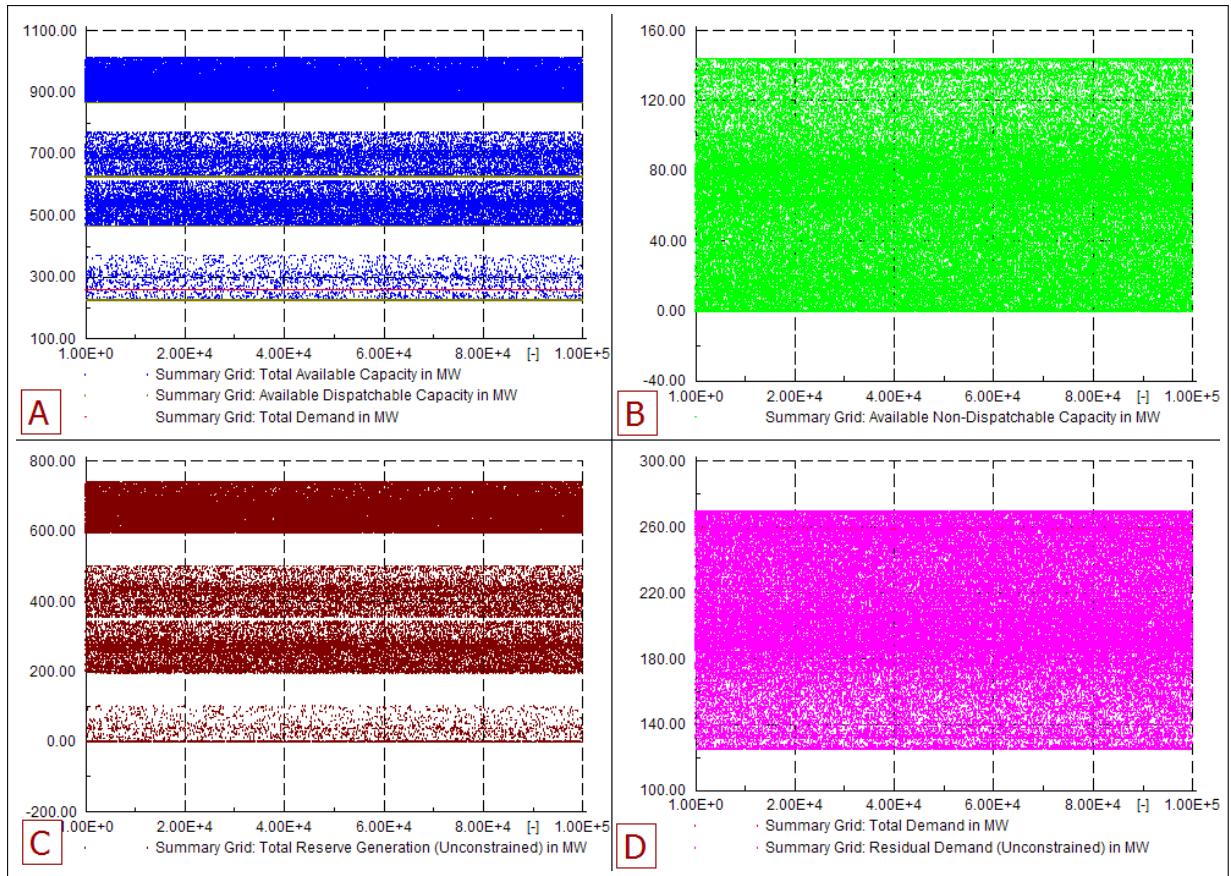


Figure 46.6.5: Monte-Carlo Draws (Iterations) Plots

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;

### 46.6.3 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 46.6.6.

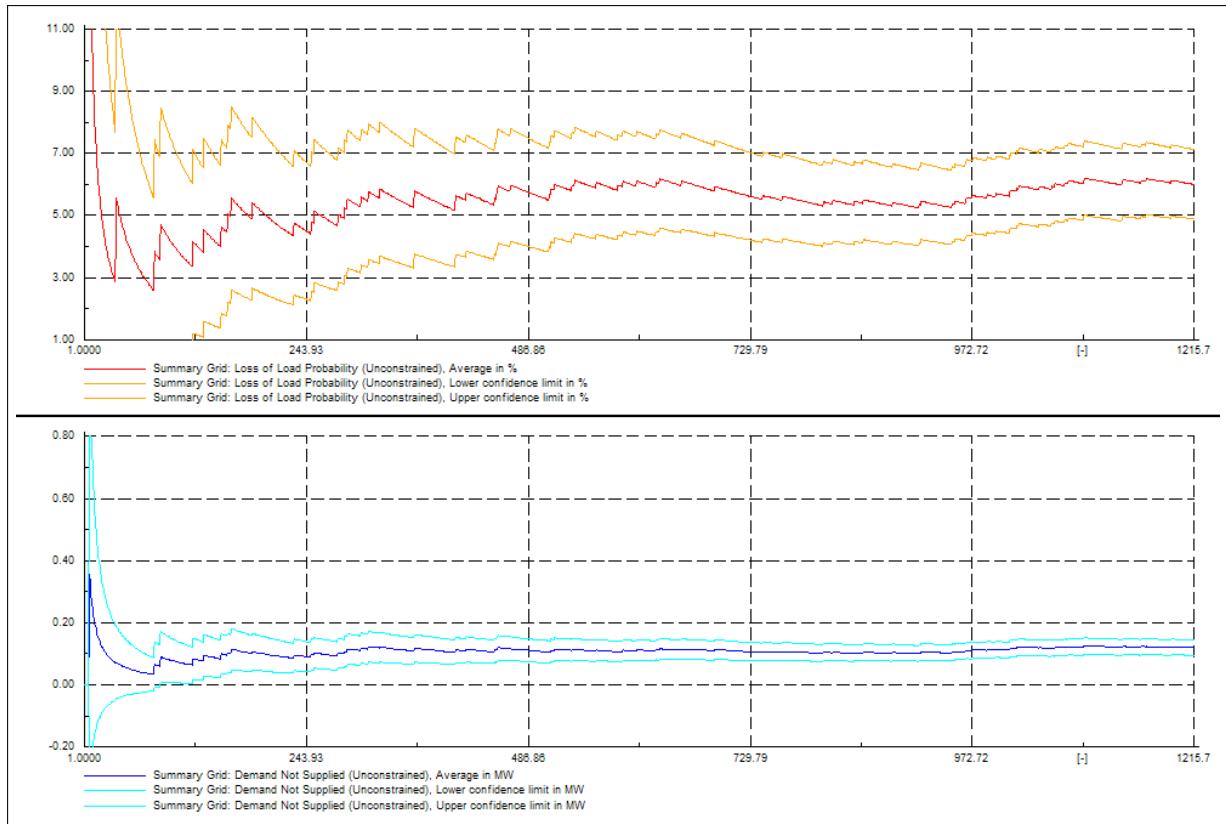


Figure 46.6.6: Example Convergence Plot

**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:

$$\sigma^2 = \frac{1}{n-1} \cdot \sum_{i=1}^n (y_i - \bar{y})^2 \quad (46.6)$$

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:

$$CL = \bar{y} \pm \frac{\sigma}{\sqrt{n}} \cdot z \quad (46.7)$$

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.

#### 46.6.4 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:NDG	$\min(C:AvailNDCap, DemTot * (1 + Losses\% / 100))$
Used Dispatchable Generation	c:DG	$\min(C:AvailDCap, DemTot * (1 + Losses\% / 100) - c:NDG)$
Total Used generation	c:TotGen	$c:Dgen + c:NDG$
Residual Demand	c:ResidDem	$c:DemTot * (1 + Losses\% / 100) - c:NDG$

Table 46.6.1: Generation Adequacy Calculated Variables

# Chapter 47

## 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 *Calculate → 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.

### 47.1 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 the Network Model Manager; reporting scripts can be used to access the complete set of results in the result file.

#### 47.1.1 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 change in power flow on a circuit as a result of the (fault) outage of another 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 40.4) 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.

---

### 47.1.2 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**:  $dv/dP$ ,  $dv/dQ$ ,  $d\phi/dP$  and  $d\phi/dQ$ .
- **At terminals for a transformer-tap sensitivity**:  $dv/dtap$ , and  $d\phi/dtap$ .

## 47.2 Sensitivities / Distribution Factors Options

### 47.2.1 Basic Options

#### 47.2.1.1 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.

---

#### 47.2.1.2 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.

### 47.2.1.3 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 47.2.3. 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 47.2.3.

#### 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.

## 47.2.2 Results

### 47.2.2.1 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.

### 47.2.2.2 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.

---

#### 47.2.2.3 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.

#### 47.2.2.4 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  $dphi/dtap$ .

### 47.2.3 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.

#### 47.2.3.1 Diagonal Elements Only

This is a rather specific option which is only visible when just *Busbar* is selected on the Basic Options page. Its purpose is to provide compatibility with the earlier Load Flow Sensitivity option “Diagonal Elements Only”. For most purposes it can be ignored.

#### 47.2.3.2 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.

#### 47.2.3.3 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.

#### 47.2.3.4 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.

#### 47.2.3.5 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.

#### 47.2.3.6 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.

### 47.2.4 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.

### 47.2.5 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.

## 47.3 Reporting

The results of the *Sensitivities / Distribution Factors* calculation can be reported using the  icon. Below, the reporting command options are described, and section [47.3.4](#) gives some information about the expected output.

### 47.3.1 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 ( $dP/dP$ )
  - $dP/dP$ ,  $dQ/dP$ ,  $dv/dP$ ,  $d\phi/dP$ ,  $dP_{loss}/dP$ ,  $dQ_{loss}/dP$
  - $dP/dQ$ ,  $dQ/dQ$ ,  $dv/dQ$ ,  $d\phi/dQ$ ,  $dP_{loss}/dQ$ ,  $dQ_{loss}/dQ$
- Transformer sensitivities
  - PSDF/TCDF ( $dP/dtap$ )
  - $dP/dtap$ ,  $dQ/dtap$ ,  $d\phi/dtap$ ,  $dv/dtap$ ,  $dP_{loss}/dtap$ ,  $dQ_{loss}/dtap$
- LODF values

### 47.3.2 Thresholds

This tab allows the user to set the thresholds for the branch and terminals sensitivities to be included in the reports.

### 47.3.3 Used Format

This tab gives read access to the reporting formats, held in the System folder of the database.

### 47.3.4 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.

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.

## 47.4 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 48

## Network Reduction

### 48.1 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 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.

The above methods both 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.

The chapter is separated into five parts. Firstly, the technical background of the *PowerFactory* Network Reduction algorithms are explained. Section 48.3 then discusses the steps needed to run a Network Reduction and Section 48.4 explains in detail each of the options of the *PowerFactory* Network Reduction tool. The penultimate part, Section 48.5, presents a simple example and the final section provides some *tips and tricks* to consider when working with the Network Reduction tool.

### 48.2 Technical Background

Some additional technical background on the Network Reduction tool is provided in the following sections.

### 48.2.1 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).

### 48.2.2 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).

### 48.2.3 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.

### 48.2.4 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.

## 48.3 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.

This process is described in detail in Chapter 15 Grouping Objects, Section 15.3 (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 48.4: Network Reduction Command.

### 48.3.1 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 48.4.5: 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 8.1.4: 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).

### 48.3.2 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 15.3 (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 15.3 (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 48.4 (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.

### 48.3.3 Expected Output of the Network Reduction

This sub-section describes the expected output of the network reduction tool after successfully executing it. The output varies depending on whether the reduced project was created in V13.2 or earlier and contains system stages, or if it was created in V14.0 or higher. Both output scenarios are explained in the following sections. Also, the additional objects that the Network Reduction tool creates are explained.

#### Changes to the network model for projects created in V14.0 or higher

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 48.3.1 shows an example of a network data model after a successful Network Reduction.

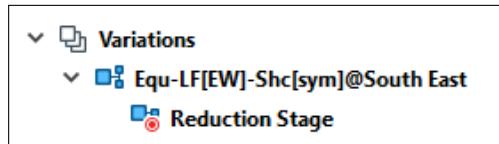


Figure 48.3.1: Variation and Reduction Stage after a successful Network Reduction using the default options.

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*)

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 48.4.6). 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 11.6). 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.

## 48.4 Network Reduction Command

In this section, the *Network Reduction* command options are explained.

### 48.4.1 Basic Options

#### 48.4.1.1 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 or the REI equivalent. Dependent on this selection the options on the following pages are offered accordingly.

#### 48.4.1.2 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 [15.3](#) (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.

#### 48.4.1.3 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.

#### 48.4.2 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.

### 48.4.3 REI Equivalent

#### 48.4.3.1 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.

#### 48.4.3.2 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.

#### 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

## 48.4.4 Dynamic Equivalent

### 48.4.4.1 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.

#### 48.4.4.2 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.

## 48.4.5 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.

---

### 48.4.5.1 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.

### 48.4.5.2 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.

### 48.4.5.3 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 [48.3.1](#) (How to Backup the Project (optional)).

### 48.4.5.4 Clean up empty substations

If this option is selected the empty substations 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.

### 48.4.5.5 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.

## 48.4.6 Advanced Options

This section describes the Advanced Options for the Network Reduction command.

### 48.4.6.1 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).

#### 48.4.6.2 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.

### 48.4.7 Verification

#### 48.4.7.1 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.

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.

#### 48.4.7.2 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),  $I'_k$  (Ikss),  $i_p$  (ip),  $i_b$  (ib),  $I_b$  (lb),  $X_b/R_b$  ( $X_{toR_b}$ ), and  $X/R$  ( $X_{toR}$ ).

## 48.5 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 48.5.1. 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*):

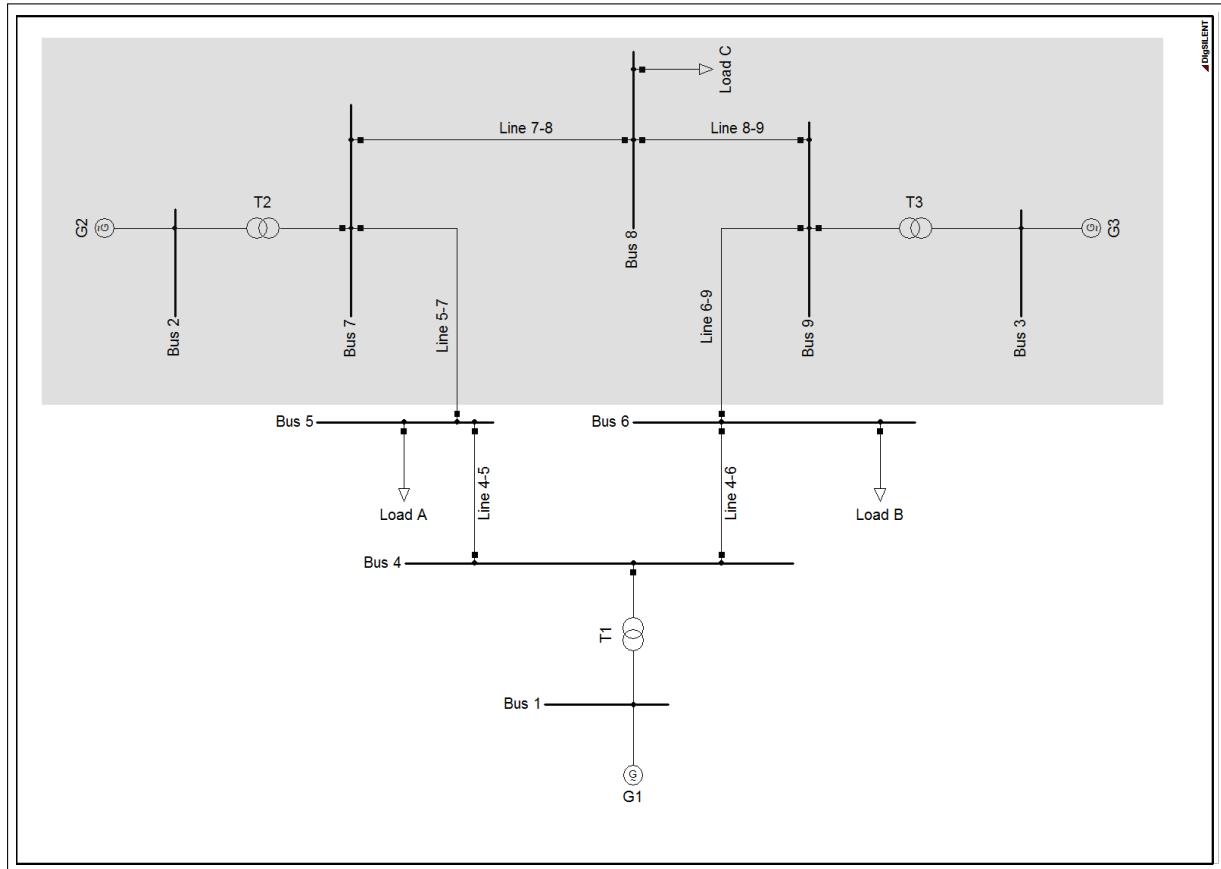


Figure 48.5.1: Example System with Original Network

1. Select the lines “Line 5-7” and “Line 6-9”.
2. Right-click on the selected lines and choose the option *Define → Boundary ....* The boundary dialog will appears.
3. Click on the **Mark Interior Region** button and verify that the region marked corresponds with the region showed grayout in Figure 48.5.1.

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 48.5.2.

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.

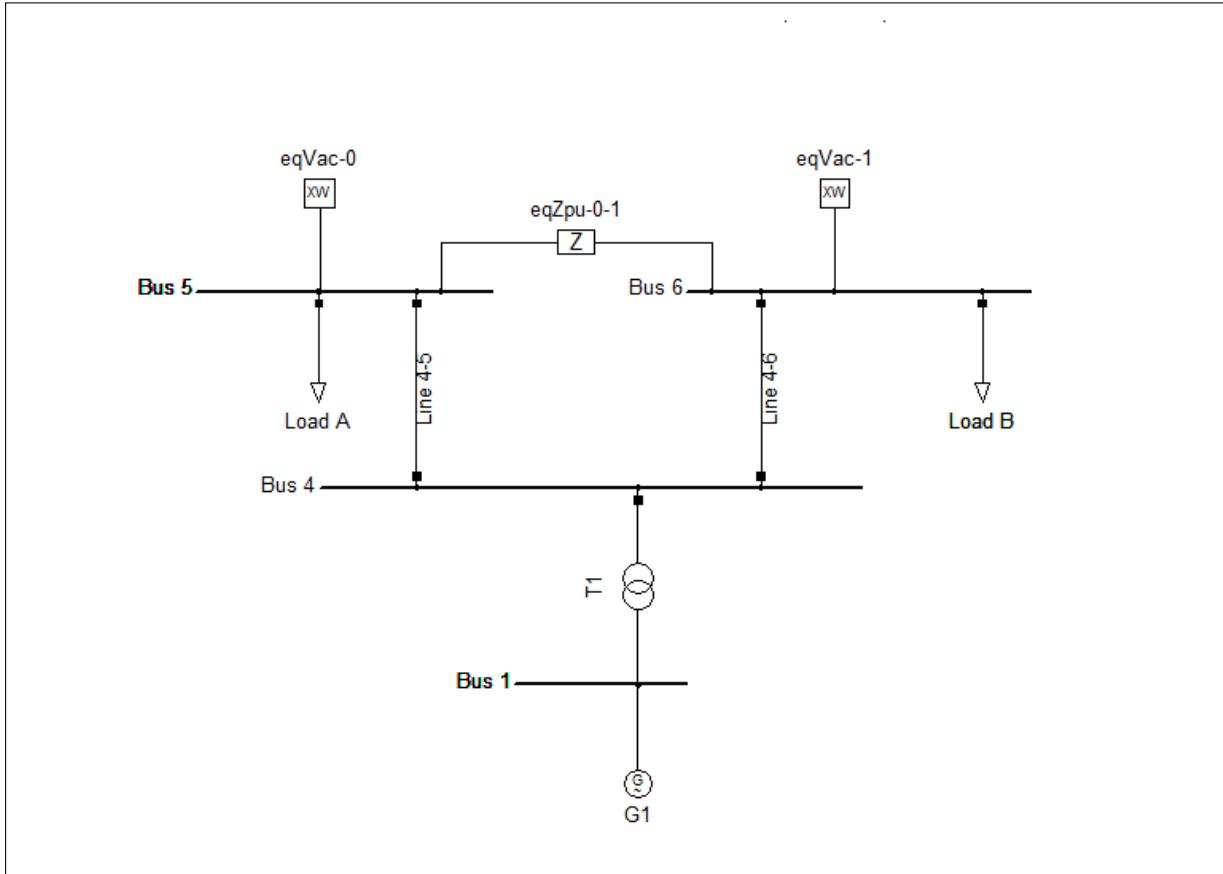


Figure 48.5.2: Example System with Reduced Network

## 48.6 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.

### 48.6.1 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.

### 48.6.2 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 49

## Techno-Economical Calculation

### 49.1 Introduction

This chapter presents the tools available to perform Techno-Economical Calculations in *PowerFactory*. It provides a general description, technical background, description of the command dialogs, and an example calculation. The Techno-Economical Calculation (*ComTececo*) can be accessed from the Additional Functions toolbar as shown in figure 49.1.1

Techno-economical calculations are used to perform an economic assessment and comparison of network expansions (projects) through an analysis of:

- The cost of electrical losses.
- The economic impact of failure rates (reliability).
- Investment costs, including initial costs, initial value, scrap value, and expected life span.
- Project timing.

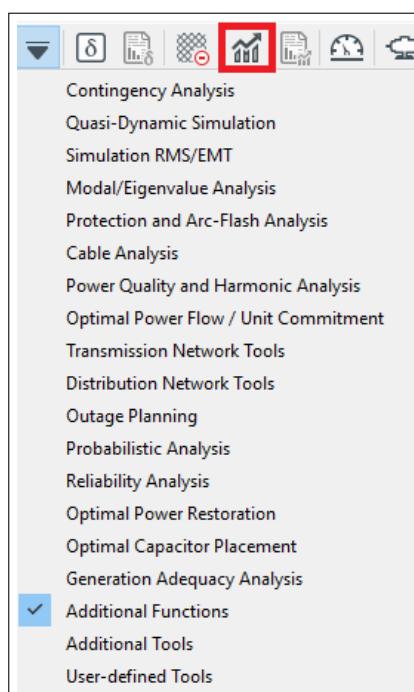


Figure 49.1.1: How to access the Techno-Economical Calculation

## 49.2 Requirements for Calculation

Prior to conducting a Techno-Economical Calculation, economic data should be defined within each Expansion Stage (*IntStage*). To define economic data, right-click on the Expansion Stage, click *Edit*, and select the *Economical Data* tab. Parameters to be defined are as follows:

### Costs for expansion

Define the *Investment* costs in “k\$”, and *Additional* costs in “k\$/a”.

### Commercial equipment value

Define the *Original value* in “k\$”, *Scrap value* in “k\$”, and *Expected life span* in years “a”. Note that the Expected life span is used in the economic calculation, it does not take the Variation out of service at the end of the expected life span.

## 49.3 Calculation Options

### 49.3.1 Basic Options Page

#### Calculation Points

Select to either *Calculate*:

- *once per year*. Calculations are executed once per year from the 1st day of the *Calculation Period Start* (01.01.XXXX, 00:00:00) to the last day of the year at the calculation period *End* (31.12.YYYY, 23:59:59).
- *for every expansion stage*. Calculations are executed on the 1st day of the *Calculation Period Start*, at the Activation Time of each Expansion Stage.
- *for user-defined dates*. Calculations are executed on the 1st day of the *Calculation Period Start*, at each user-defined date. To define dates, *Insert* rows to the *Calculation Points* table and specify the required dates. To automatically populate the table of calculation points with *once per year* dates and *for every expansion stage* dates, select **Get keyAll keyCalculation keyPoints**. The dates can then be edited as required (note that it is not possible to append rows beyond the end date).

---

**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.

---

#### Strategy

Click **Show Activated Variations** to show Activated Variations. Only Expansion Stages within Activated Variations, and an Activation Time within the Calculation Period will be considered by the calculation.

#### Calculatory Interest Rate

Specify the *Calculatory Interest Rate* to be used in the Net Present Value (NPV) calculations.

#### Tolerance

Specify a “Tolerance for calculation points (in days)” for 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.

## Load growth

Optionally *Incorporate load growth* in the calculation, to consider load growth within each calculation interval. In contrast to the case where no load growth is incorporated and costs for a calculation period are calculated at the beginning of that period, enabling this flag will lead to a second cost calculation at the end of the current calculation period. Corresponding costs are then calculated 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).

### 49.3.2 Costs Page

Optionally consider *Losses*, *Interruption Costs*, *User-defined Costs*, and *Annual additional costs*, and select whether to *Optimise Tie Open Points*.

#### Losses

- Optionally modify the *Load Flow Calculation* options via the pointer to the Load Flow Calculation command.
- Select whether to consider losses for the *whole system*, or for a *user-defined set of substations/feeders*. If more than one feeder or substation is selected, PowerFactory automatically creates a Set within the active Study Case, by default named “Techno-eco. Calc. - Substations/Feeder Set”.
- Define the *Costs for Losses (Load)* in “\$/kWh”, relating to line losses.
- Define the *Costs for Losses (no Load)* in “\$/kWh”, relating to transformer no-load losses.

#### Interruption Costs

Modify the Reliability Assessment options. By default, a new Reliability Assessment command object is created within the Techno-Economical command. See Chapter 44 for details of how to configure the Reliability Command options. For a Techno-Economical calculation, it is generally recommended that the following options are selected in the Reliability Assessment Command:

- *Basic Options* → *Load flow analysis*
- *Basic Options* → *Distribution (Sectionalising, Switching actions)*
- *Advanced Options* → *Automatic Power Restoration*
- *Advanced Options* → *Do not save corresponding events*

#### User-defined costs

Optionally select a user-defined DPL *Cost Assessment Script*. This functionality may be required for detailed analysis where factors besides losses and outage costs are to be considered in the calculation.

#### Annual additional costs

Optionally define Annual additional costs in “k\$/a”. These are costs that are to be applied irrespective of the network development strategy.

#### Optimise Tie Open Points

Optionally select to calculate losses from the output of the TOPO calculation. The network open point(s) will be re-configured during the Techno-Economical Calculation in order to minimise losses, in accordance with the options selected in the TOPO command. By default, a new TOPO command object is created within the Techno-Economical command. See Section 41.4: Tie Open Point Optimisation, for details on how to configure the TOPO command.

---

**Note:** If the costs of losses are not considered by the Techno-Economical command directly, *Optimise Tie Open Points* may still be selected so that the impact of network switching configuration is considered by the calculation, where either *Interruption Costs* or *Additional Costs* is selected.

---

### 49.3.3 Output Page

#### Results

A reference (pointer) to the results object.

#### Report

(Optionally) select the format of results printed to the output window. The report includes a summary of selected calculation options, and annual costs, total costs, and Net Present Value (NPV).

### 49.3.4 Parallel Computing Page

Parallel calculation of the *Techno-Economical Calculation* is possible and can be activated on the Parallel Computing page of the *Techno-Economical Calculation* command dialog. The options provided on this page are described below.

**Parallel computation:** if the checkbox is ticked, the *Techno-Economical Calculation* is executed in parallel. Otherwise, the calculation is run sequentially.

**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 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 [22.4](#).

### 49.3.5 Results Reports

After a *Techno-Economical Calculation* reports can be created in the output window. The corresponding button  is located to the right of the Techno-Economical Calculation icon on the *Additional Functions* Toolbox. Depending on the options three reports can be created.

**Settings of the Calculation:** if the checkbox is ticked, a report is created reflecting the settings used for the *Techno-Economical Calculation*

**Variation Input Data:** If this checkbox is selected, the configurations of the active expansion stages containing economic data are reported.

**Costs of indiv. Calculation Periods:** if this option is selected, a report of the cash flows including a summary of the annual costs, total costs, and Net Present Value (NPV) is created.

## 49.4 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 44 (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 49.4.1).

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 49.4.2).

Based on this, it has been determined that a new substation is required in 2015 to off-load the existing line. Figure 49.4.3 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.

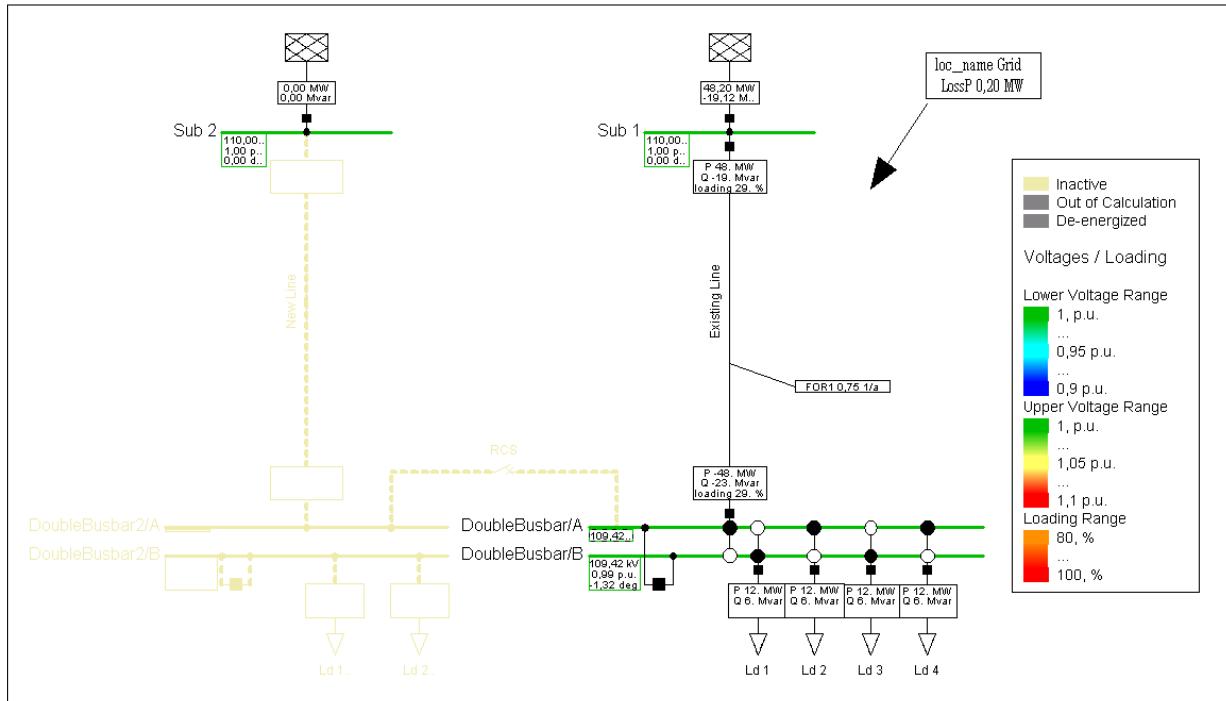


Figure 49.4.1: Example case, study time “2010”

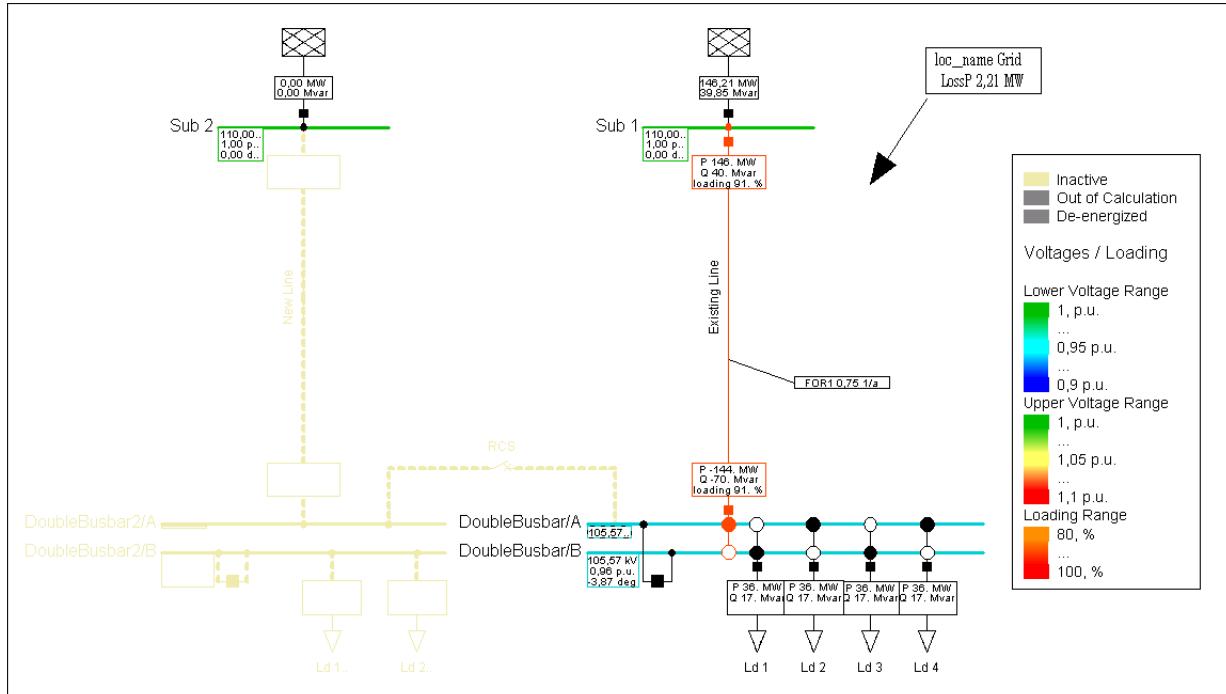


Figure 49.4.2: Example case,study time "2014"

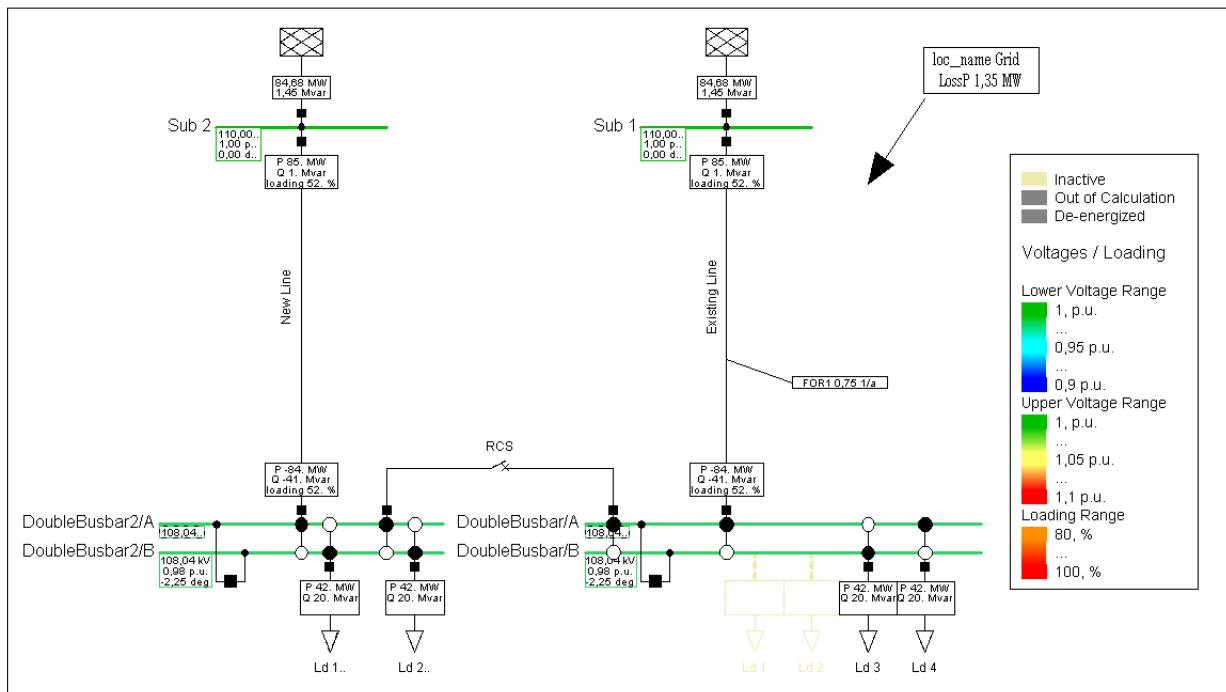


Figure 49.4.3: Example case,study time "2015"

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 49.4.1 below summarises the results of the Techno-Economical calculations.

T(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 49.4.1: Summary of Calculation Results

---

**Note:** To automatically calculate the optimal Activation Time for an Expansion Stage, in the Data Manager, right-click on the Expansion Stage, select “Execute DPL Scripts” and run the “Efficiency ratio calculation” script.

---

# Chapter 50

## State Estimation

### 50.1 Introduction

The State Estimation (SE) function of *PowerFactory* provides consistent load flow results for an entire power system, based on real time measurements, manually entered data and the network model. Before any further analysis, such as contingency analysis, security checks etc. can be carried out, the present state of a power system must be estimated from available 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 are usually more data available than absolutely necessary and it is possible to profit by redundant measurements for improving the accuracy of the estimated network state.

The states that can be estimated by the State Estimation on the base of the given measurements vary for different elements in the network:

- Loads
  - Active Power, and/or
  - Reactive Power, or
  - Scaling Factor, as an alternative
- Synchronous Machines
  - Active Power, and/or
  - Reactive Power
- Asynchronous Machines
  - Active Power
- Static var System
  - Reactive Power
- 2- and 3-winding transformers
  - Tap Positions (for all but one taps).

## 50.2 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:

$$f(\vec{x}) = \sum_{i=1}^n \left( \sigma_i^{-1} \cdot \frac{calVal_i - meaVal_i}{rating} \right)^2 \quad (50.1)$$

where:

$$\sigma_i = (accuracy_i / 100)$$

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 a weighting factor  $w_i$  to the standard deviation of the corresponding measurement device (+transmission channels, etc.).

In this setting, the goal of a State Estimation is to minimise the above given function  $f$  under the side constraints that all load flow equations are fulfilled.

## 50.3 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. State Estimation (Non-Linear Optimisation)

Figure 50.3.1 illustrates the algorithmic interaction of the different components. The first *Preprocessing* phase adjusts all breaker and tap positions according to their measured signals.

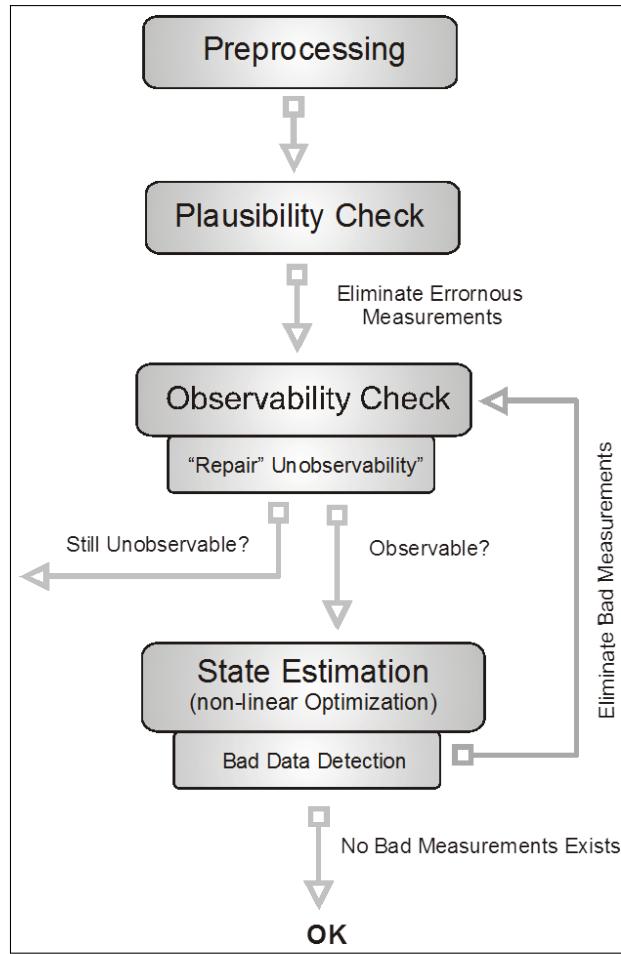


Figure 50.3.1: Variation of the *PowerFactory* State Estimation algorithm

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.

### 50.3.1 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 [50.6](#)).

### 50.3.2 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 [50.6.4](#)).

#### Observability of individual states

The Observability Analysis identifies not only, 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 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.

### 50.3.3 State Estimation (Non-Linear Optimisation)

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.

In order to come up with a solution for a non-observable system, various strategies can be followed: One option is to reset all non-observable states, such that some manually entered values or historic data is used for these states. An alternative option is to use so-called pseudo-measurements for non-observable states. A pseudo-measurement basically is a measurement with a very poor accuracy. These pseudo-measurements force the algorithm to converge. At the same time, the resulting estimated states will be of correct proportions within each equivalence-class.

In the remaining sections of this guide of use, the instructions related to Data Entry, Options and Constraints, and Visualisation of Results are presented.

## 50.4 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)
  - 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*.

### 50.4.1 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 50.4.1). Then, the new object dialog pops up with a predefined list of external measurements. Please select the desired measurement device among this list.

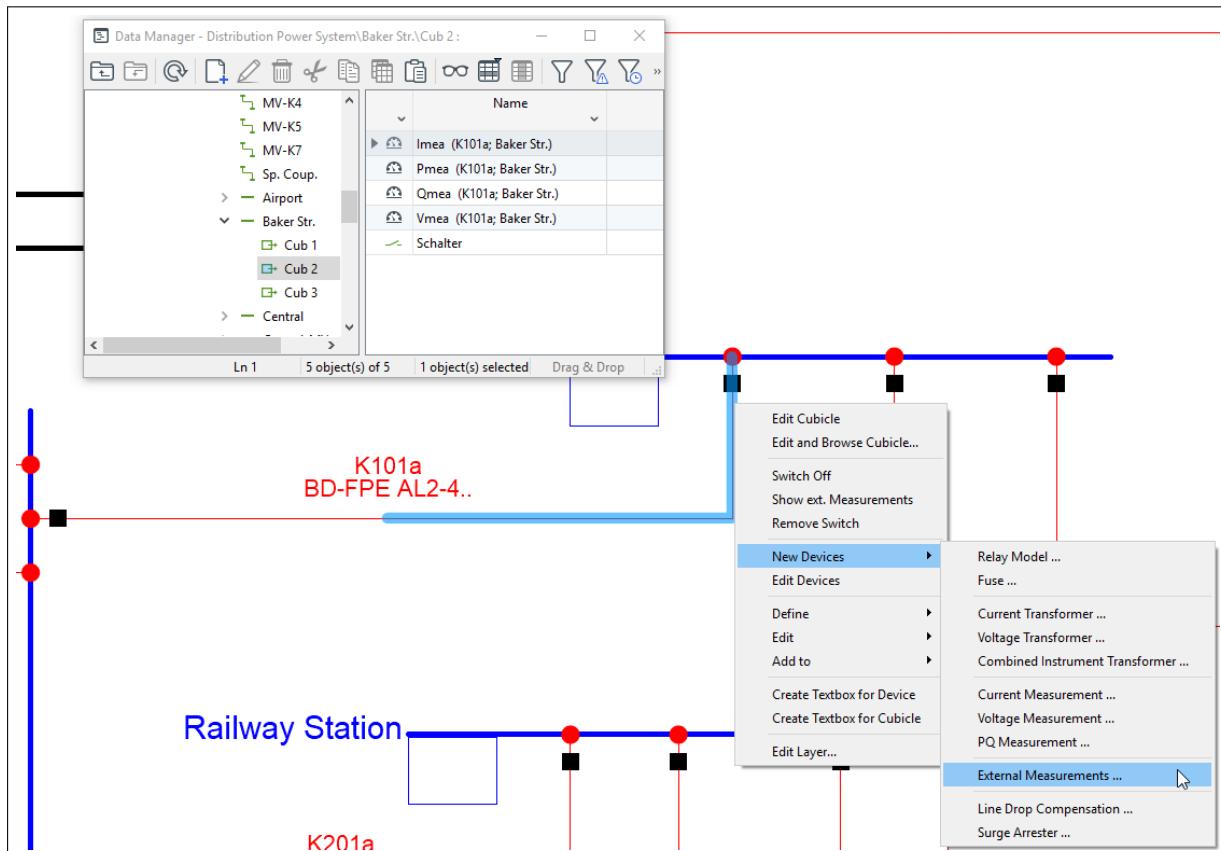


Figure 50.4.1: External Measurements that are located in a cubicle

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.

#### Branch Flow Measurements

Any branch flow measurement (*StaExpmea*, *StaExtqmea*) is defined by the following values (see figures 50.4.2 and 50.4.3):

- Measured value (e:Pmea or e:Qmea, respectively)

- Multiplicator ( $e:\text{Multip}$ )
- Orientation ( $e:\text{i\_gen}$ )
- Accuracy class and rating ( $e:\text{Snom}$  and  $e:\text{accuracy}$ )
- Input status (to be found on the second page of the edit object, see Figure 50.4.3):  
E.g., tele-measured, manually entered, read/write protected,... ( $e:\text{iStatus}$ ). 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.

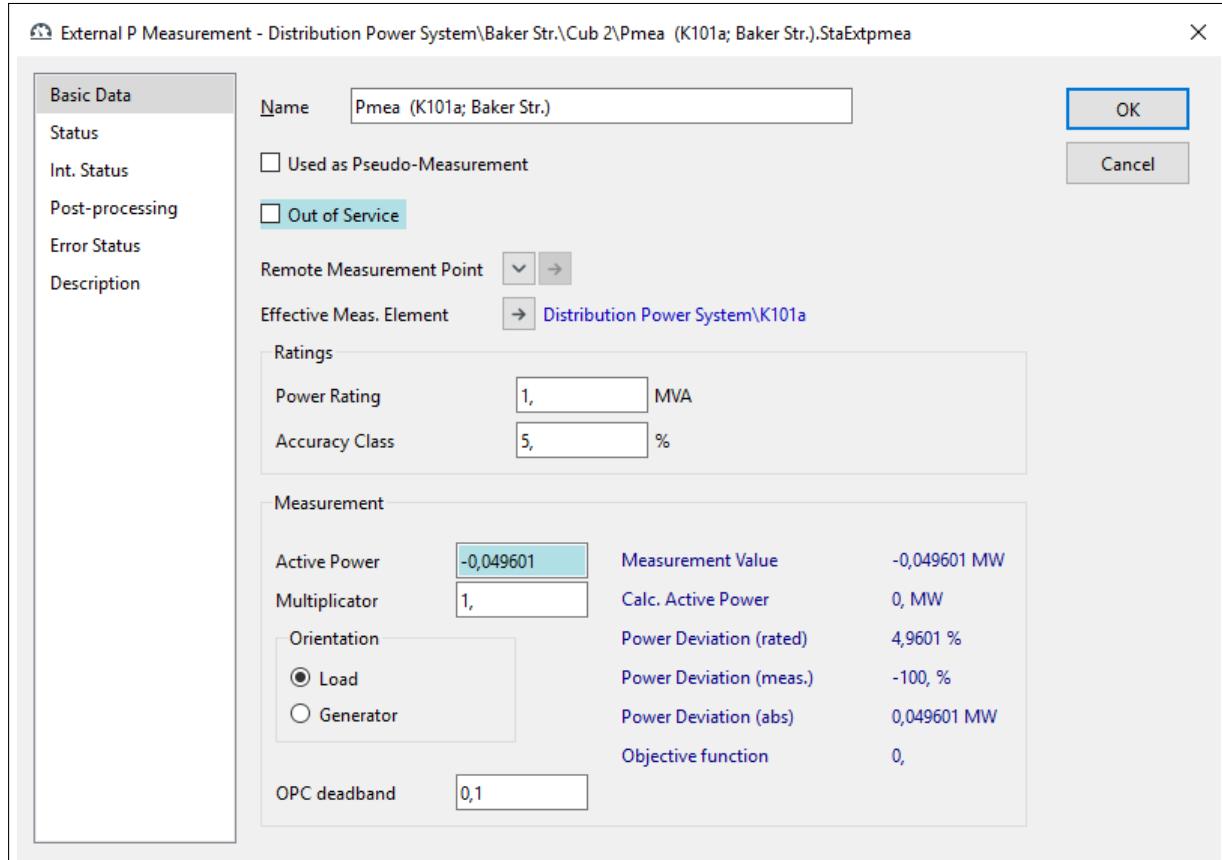


Figure 50.4.2: Dialog for an external P-measurement

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 ( $e:\text{pseudo}$ ). 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 50.5.1: Basic Setup Options for details).

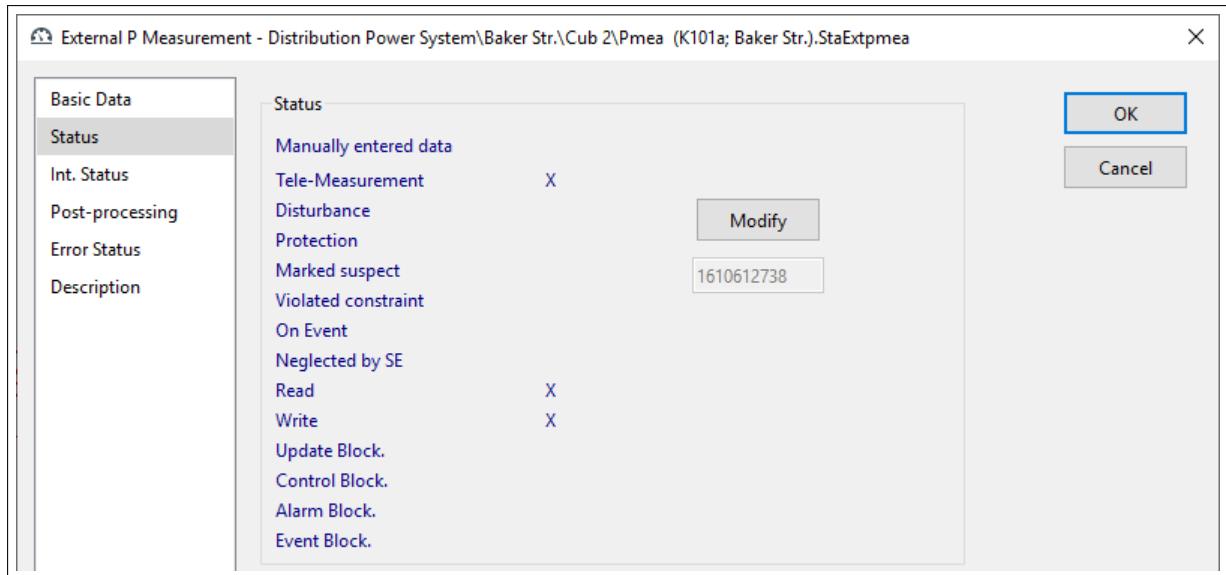


Figure 50.4.3: Status page of the dialog for an external P-measurement

### Current Measurements

The External I-measurement (*StaExtimea*) plays a special role and slightly differs from the External P- and Q-measurements (see Figure 50.4.4): 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*).

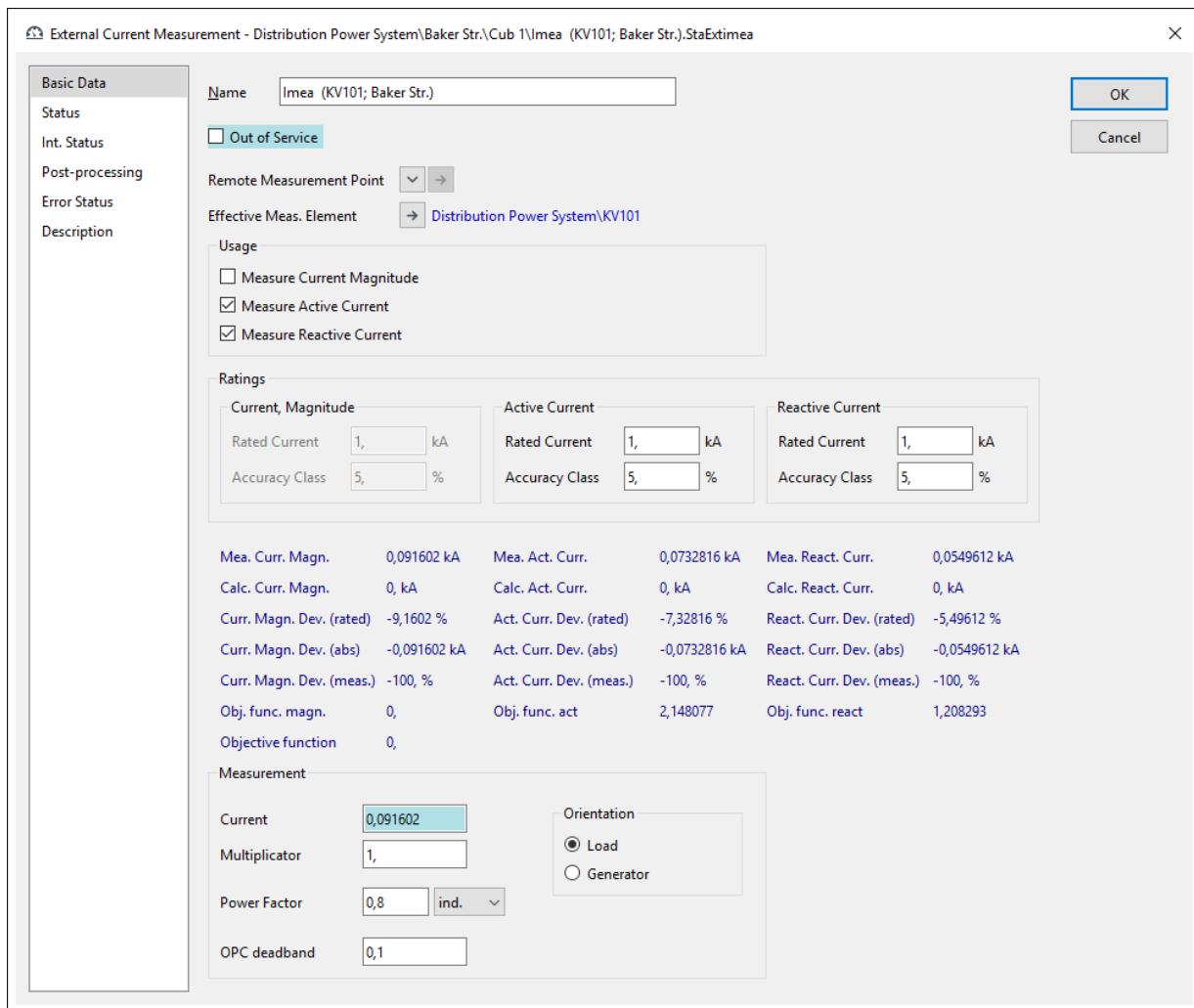


Figure 50.4.4: Dialog for an external I-measurement

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 is 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).

### Voltage Measurements

Voltage measurements (`StaExvmea`) need to be placed in cubicles as well. The measurement point then is the adjacent terminal.

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.

### **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).

## **50.4.2 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. 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 and all tap positions.

### **Loads**

For each load (*ElmLod*), the user can specify whether its active and/or reactive power shall be estimated by the State Estimation. Alternatively, the State Estimation is able to estimate the scaling factor (for a given P and Q injection). The specification which parameter shall be estimated, is done by checking corresponding boxes on the *State Estimation* page of the load. When these options are disabled, the load is treated as in the conventional load flow calculation during the execution of the SE.

### **Synchronous Machines**

Similarly, for synchronous machines (*ElmSym*), the active and reactive power can be selected as a control variable for being estimated by the State Estimation. Again, the user will find corresponding check boxes on the *State Estimation* page of the element.

If the corresponding check box(es) are disabled, the synchronous machine behaves as in the conventional load flow calculation.

### **Asynchronous Machines**

For asynchronous machines (*ElmAsm*), the active power may serve as a state to be estimated. Once again, the corresponding box has to be checked on the *State Estimation* page.

If the corresponding check box is disabled, the asynchronous machine behaves as in the conventional load flow calculation.

### **Static var Systems**

For static var systems (*ElmSvs*), the reactive power may serve as a state to be estimated. Again, the corresponding box has to be checked on the *State Estimation* page.

If the corresponding check box is disabled, the static var system behaves as in the conventional load flow calculation.

### **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.

The corresponding check boxes are found on the *State Estimation* page of the transformers. If the check box is disabled the State Estimation will treat the tap position of the transformers as in the conventional load flow calculation.

## 50.5 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**.

### 50.5.1 Basic Setup Options

Recall that the State Estimation in *PowerFactory* consists of three different parts (Plausibility Check, Observability Analysis, State Estimation (non-linear optimisation)) and an additional precedent Preprocessing step. This variation is reflected in the Basic Options dialog.

#### 50.5.1.1 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 or the State Estimation.

##### **Adapt breaker measurements**

If this check box is marked, all measured breakers statuses will be set to the corresponding measured signal values.

##### **Adapt tap position measurements**

If this check box is marked, all measured tap positions will be set to the corresponding measured values.

#### 50.5.1.2 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 50.6).

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  $\sigma \cdot rating$ , where  $\sigma$  and *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  $P_{mea_1}, \dots, P_{mea_r}$  in each of its r connection devices. The threshold  $\varepsilon$ , by which the nominal loss shall not be exceeded, is given by:  $\varepsilon = \sum_{i=1}^r \sigma_i \cdot rating_i$ , where  $\sigma_i$  and  $rating_i$  are the accuracy and the rating, respectively, of measurement  $P_{mea_i}$ .

### 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  $P_{mea_1}, \dots, P_{mea_r}$  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  $(-\sum_{i=1}^r \sigma_i \cdot rating_i)$ .

### 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  $\sigma \cdot rating$ .

### 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  $P_{mea_1}, \dots, P_{mea_r}$  has magnitude below the threshold  $\sum_{i=1}^r \sigma_i \cdot rating$  (similarly for Q-measurements).

#### 50.5.1.3 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.

##### Check for observability regions

If the corresponding check box is marked by the user, the execution of the State Estimation will run the Observability Analysis (prior to the state Estimation optimisation).

##### 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 50.6 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. 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 P- (Q-)pseudo-measurement in the cubicle of the element carrying the non-observable state. 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. 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.

#### 50.5.1.4 State Estimation (Non-Linear Optimisation)

The non-linear optimisation is the central component of the State Estimation. The underlying numerical algorithm to minimise the measurements’ overall error is the iterative Lagrange-Newton method.

##### Run state estimation algorithm

Check this box to enable the non-linear optimisation. Note that after convergence of the method, upon user settings on the advanced state estimation option page PowerFactory performs a bad data check which eliminates the worst P-,Q-,V-, and I-measurements among all bad data. Observability Analysis and State Estimation are run in a loop until no further bad measurements exist (recall the algorithm variation as shown in Figure 50.3.1).

## 50.5.2 Advanced Setup Options for the Plausibility Check

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  $\epsilon = \sum_{i=1}^r \sigma_i \cdot \text{rating}$ . The user has the possibility to influence the strictness of this threshold. Therefore, the settings provide to enter so-called “exceeding factors”  $fac > 0$  such that the new threshold is  $fac \cdot \epsilon$  instead of  $\epsilon$ . 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.

### 50.5.3 Advanced Setup Options for the Observability Check

#### 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.

#### 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.

### 50.5.4 Advanced Setup Options for 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 50.3.1). 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

$$\frac{\text{calcVal} - \text{meaVal}}{\text{rating}} \geq \frac{\text{accuracy}}{100} \quad (50.2)$$

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 (50.2) exceeds  $\text{facErr} \cdot \text{accuracy}/100$ . 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).

### 50.5.5 Advanced Setup Options for Iteration Control

#### Initialisation

The non-linear optimisation requires an initialisation step to generate an initial starting configuration.

## Initialisation of non-linear optimisation

The user may specify whether the initialisation shall be performed by a load flow calculation or by some 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.

### Load Flow

Specifies the settings of the load flow command which is taken for initialisation in case no flat start is used.

### Stopping criteria for the non-linear optimisation

The non-linear optimisation is implemented using an iterative Newton-Lagrange method. 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

$$\min f(\vec{x}) \quad (50.3)$$

under the constraint that

$$g(\vec{x}) = 0 \quad (50.4)$$

where  $g$  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

$$L(\vec{x}, \vec{\lambda}) = f(\vec{x}) + \vec{\lambda}^T \cdot \vec{g}(\vec{x}) \quad (50.5)$$

with the Lagrange multipliers  $\vec{\lambda}$ .

The following parameters can be used to adapt the stopping criteria for this iterative process. The algorithm stops successfully if the following three issues are fulfilled:

1. The maximum number of iterations has not yet been reached.
2. All load flow constraint equations  $g(\vec{x}) = 0$  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 parameters serve to adjust these stopping criteria. The user unfamiliar with the underlying optimisation algorithm is urged to use the default settings here.

### Iteration Control of non-linear optimisation

The user is asked to enter the maximum number of iterations.

### Convergence of Load Flow Constraint Equations

The user should enter a maximal 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.

### Convergence of Objective Function

The user is asked 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.

In the first case, the user is asked to enter an absolute maximum change in value of the objective function. If the change in value between two consecutive iterations falls below this value, the Lagrangian is assumed to be converged.

In the latter case, the user is asked to enter an absolute maximum value for the gradient of the Lagrangian. If the gradient falls below this value, the Lagrangian is assumed to be converged.

It is strongly recommended due to mathematical precisioness to use the criterion on the gradient. The other option might only be of advantage if the underlying Jacobian matrix behaves numerically instable which then typically results in a “toggling” of the convergence process in the last iterations.

### Output

Two different levels of output during the iterative process can be selected.

## 50.6 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.

### 50.6.1 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`).

### 50.6.2 External Measurements

## 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 *StaExt\*mea*
- e:cMeaVal: measured value (including multiplier)
- 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.

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).

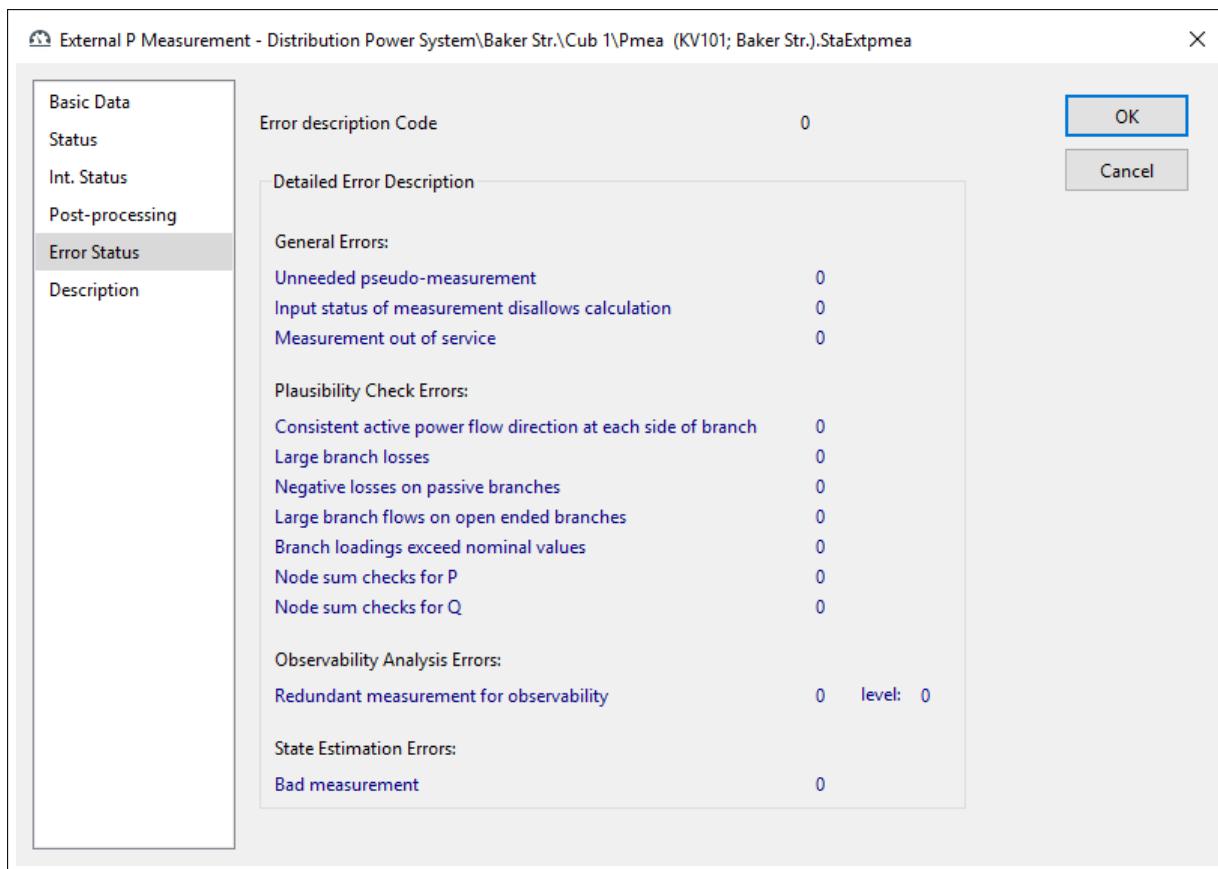


Figure 50.6.1: Description page for external measurements (*StaExtVmea*, *StaExtQmea*, *StaExtPmea*).

## 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 figures 50.6.1) 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).

### 50.6.3 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 50.4.2: Editing the Element Data) 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:iPSetp*, *c:iQSetp*, *c:iScaleSetp*, *c:iTapSetp* 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*, *ElmAsm*, *ElmSym*, *ElmSvs*, *ElmTr2*, and *ElmTr3*.

#### 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  $c:iPobsFlg$ ,  $c:iQobsFlg$ ,  $c:iScaleobsFlg$ ,  $c:iTapobsFlg$  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  $i$ .

#### 50.6.4 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 50.6.2). The colouring can be accessed by clicking the icon  on the task bar.

The colour representation paints the location of measurements (of a specific type) and the location of states (of a specific type) simultaneously.

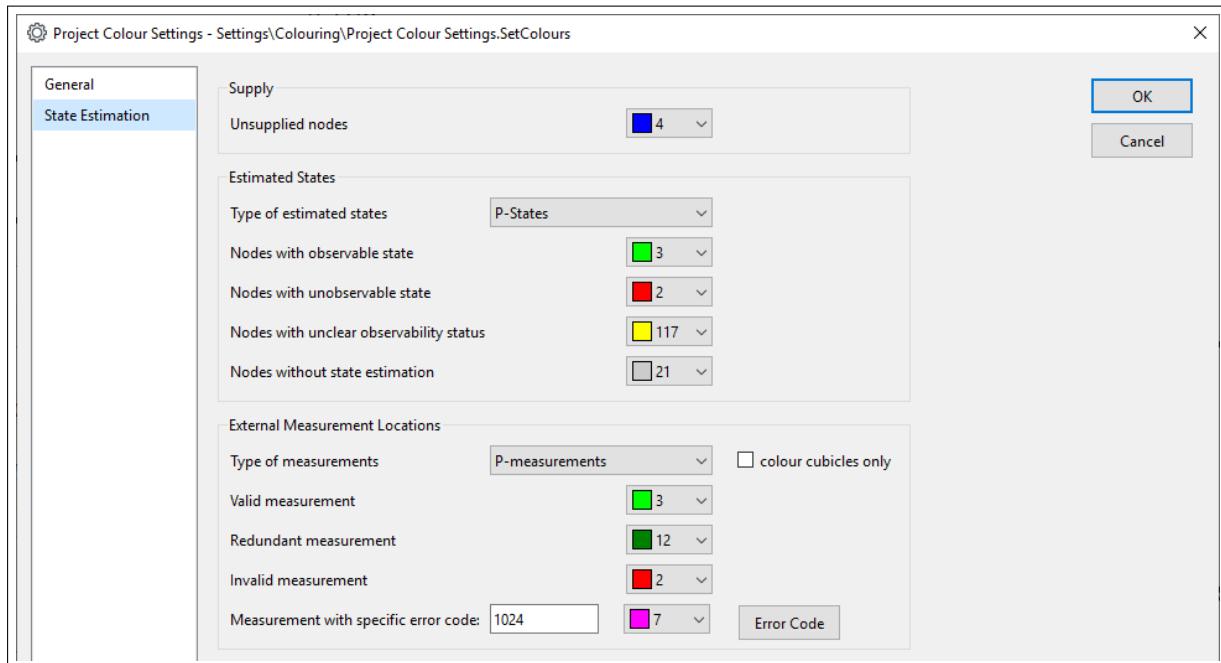


Figure 50.6.2: Colouring of measurement error statuses and estimated states.

#### 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 51

## Motor Starting

### 51.1 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.

### 51.2 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 a motor and (optionally) define a motor driven machine (mdm).

#### 51.2.1 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 *PowerFactory* Global 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 >=*.

### 51.2.2 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 *PowerFactory* Global 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 *Define... → New Motor Driven (mdm) machine*.
  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 *Define... → Motor Driven (mdm) machine from library*.
  2. Select an existing motor driven machine from the project library, or global library *Database → Library → Motor Driven Machine*.

---

**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](#).

---

## 51.3 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 *Calculate → Motor Starting*.
3. Enter the command options (see following subsections for a description of the command options).

### 51.3.1 Basic Options Page

#### 51.3.1.1 Motor(s)

The motors selected for the Motor Starting command.

### 51.3.1.2 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.

---

### 51.3.1.3 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 sets are merged with the Motor Starting command default variables.
  4. Modify Simulation command (*ComSim*) settings as required.

### 51.3.1.4 Monitoring

Click *Select* (▼) and select the *Additional Terminals* to be monitored for the Motor Starting simulation.

### 51.3.1.5 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 51.1:

$$I_{thrTs} = \sqrt{\frac{I_{thr1s}^2}{T_{start}}} \quad (51.1)$$

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:

$$I_{2t} = \int_0^{T_{start}} I^2 dt \approx \sum_0^{T_{start}} I^2 \Delta t \quad (51.2)$$

Where:

$\Delta t$  = is the integration step size of the simulation.

The calculated thermal current ( $I_{thrcalc}$ ) is then calculated as follows:

$$I_{thrcalc} = \sqrt{\frac{I_{2t}}{T_{start}}} \quad (51.3)$$

Finally, the thermal loading is calculated as the relation between rated thermal current and calculated thermal current at starting time:

$$\text{ThermalLoading} = \frac{I_{thrcalc}}{I_{thrTs}} \quad (51.4)$$

#### • 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 51.5 :

$$I_{thrTs} = \sqrt{\frac{I_{thr1s}^2}{T_{start}}} \quad (51.5)$$

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

$$I_{thrcalc} = I_{start} \quad (51.6)$$

Finally, the thermal loading is calculated as the relation between rated thermal current and calculated thermal current at starting time:

$$\text{ThermalLoading} = \frac{I_{thrcalc}}{I_{thrTs}} \quad (51.7)$$

### 51.3.2 Output Page

#### 51.3.2.1 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.

### 51.3.2.2 Static Simulation

## Report

Optionally modify report *Settings* and *Results*. Figure 51.3.1 shows an example of a Static Simulation Report with the option “Check Thermal Limits of Cables and Transformers” selected.

			DIGSILENT	Project:
			PowerFactory	
			15.0.1	Date:
Motor Starting				
Study Case:	Study Case		Annex:	/ 1
Simulation Type:	Static Simulation			
Motors				
Detailed Models				
Motor Name	Terminal Name	Terminal Voltage	Starting Current	Starting P.F.
		Voltage Minimum on After Starting	Current	Start?
		Before Starting Starting		Starting Time
		(p.u.) (p.u.) (p.u.)	(kA) (p.u.)	(s)
ASMiline	BB	1.000 0.447	0.984 1.523	15.826 0.706 Yes 7.221
Thermal Limits Violations				
Cables				
Cable Name	Starting Time	Rated Thermal Current @ 1 s	Calculated Thermal Current @ Start. Time	
	(s)	(kA)	(kA)	(%)
LineBBx1	7.221	2.000	0.744 1.523	204.601

Figure 51.3.1: 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.

### 51.3.3 Motor Starting simulation results

#### 51.3.3.1 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.

#### 51.3.3.2 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.

### 51.3.4 Motor Starting Example

Consider the following dynamic motor starting example for a single 6.6kV asynchronous motor shown in Figure 51.3.2.

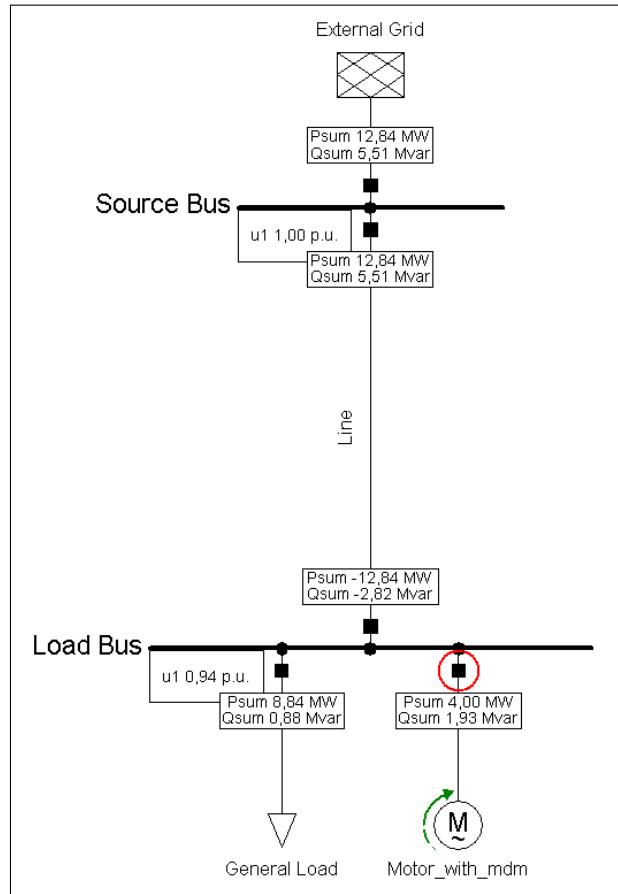


Figure 51.3.2: Motor Starting example Single Line Graphic

The *Variable Rotor Resistance* starting method has been selected, with three values of time-dependent resistance, as shown in Figure 51.3.3.

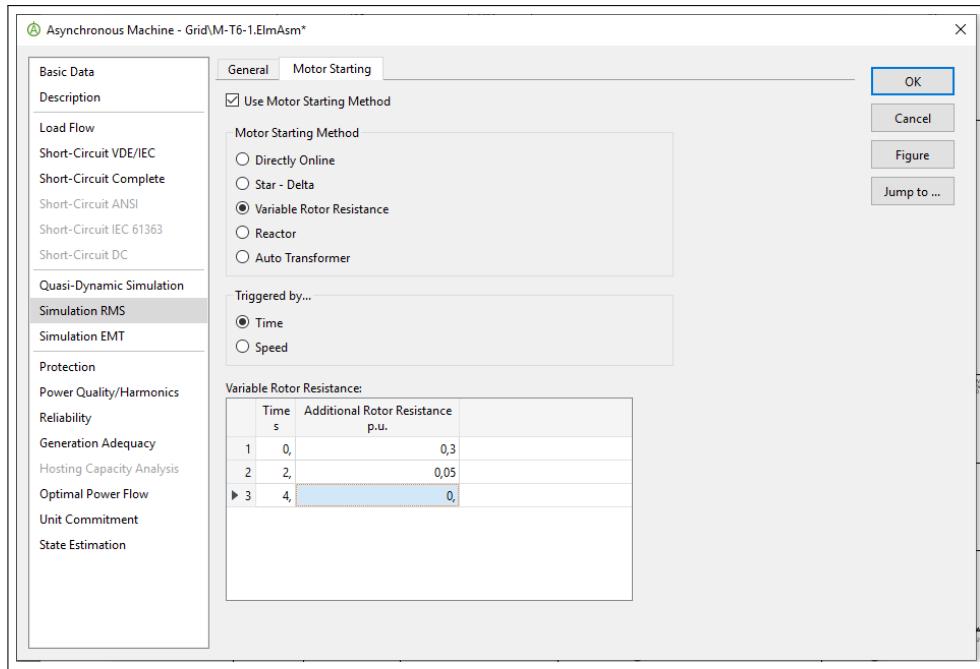


Figure 51.3.3: Motor starting methodology options

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 51.3.4.

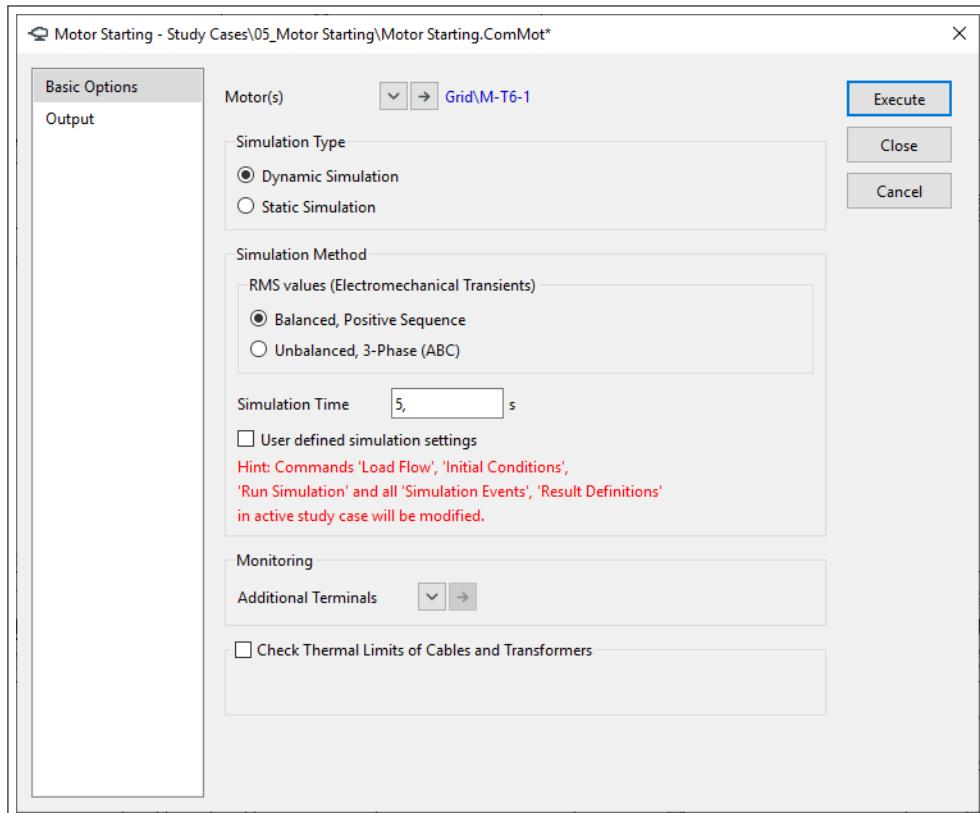


Figure 51.3.4: Motor starting Basic Options

Following execution of the command, *PowerFactory* automatically produces plots showing motor quantities of interest (as described in Section 51.3.3.1) and monitored voltage results as shown in Fig-

ure 51.3.5 and Figure 51.3.6.

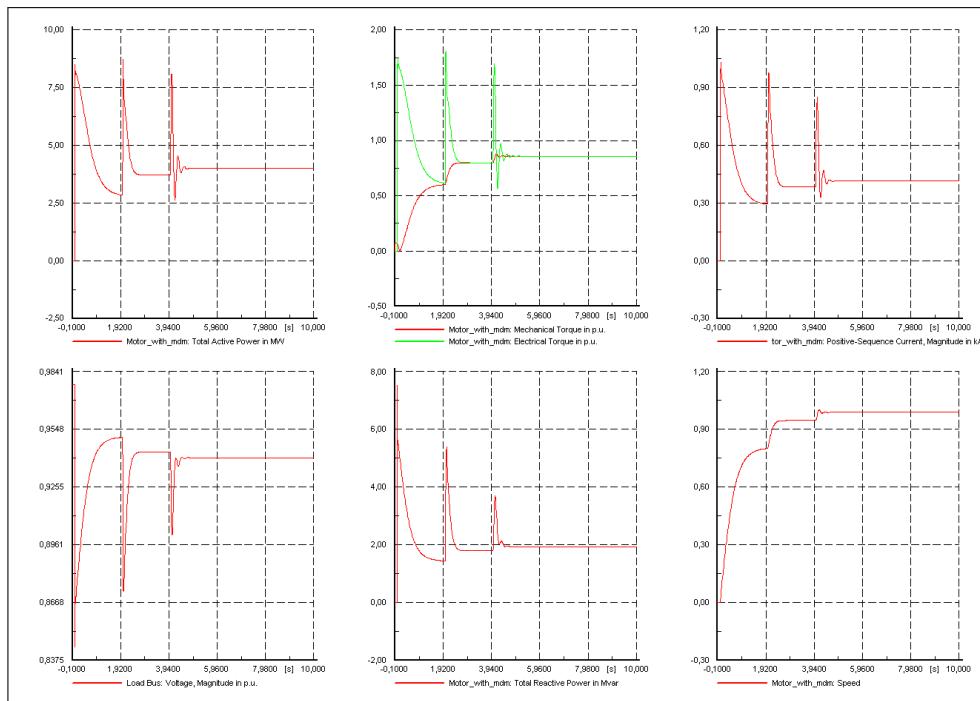


Figure 51.3.5: Motor starting example motor results

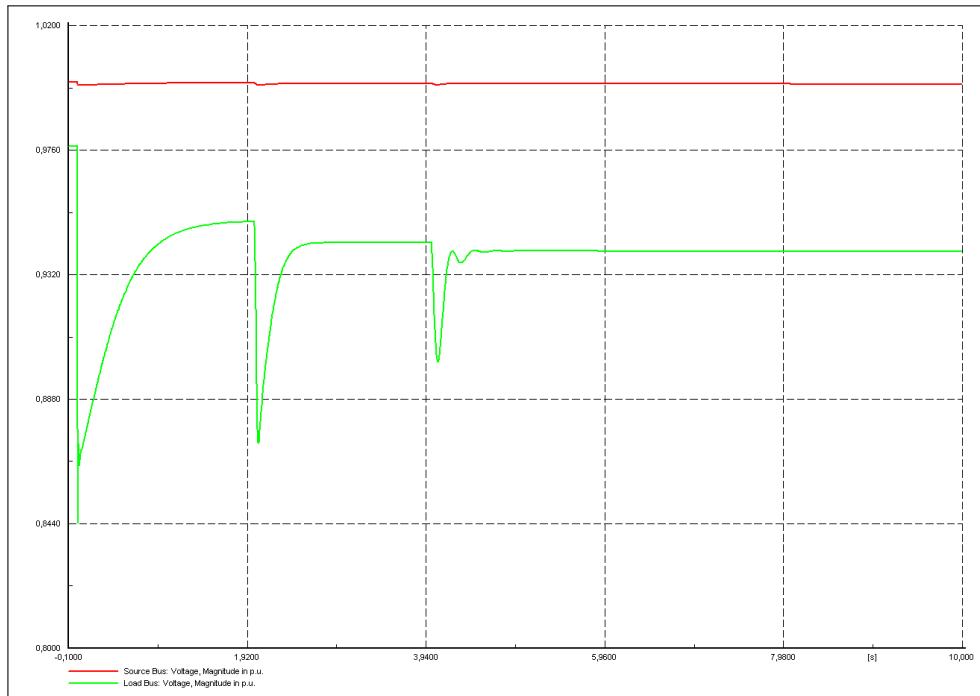


Figure 51.3.6: Motor starting example voltage results

**Part V**

## **Appendix**

# Appendix A

## Hotkeys Reference

### A.1 Calculation Hotkeys

Combination	Description
<b>F10</b>	Perform Load Flow calculation
<b>F11</b>	Perform Short-Circuit calculation
<b>Ctrl + F10</b>	Edit Load Flow calculation options
<b>Ctrl + F11</b>	Edit Short-Circuit calculation options
<b>F12</b>	Reset Calculation

Table A.1.1: Calculation Hotkeys

### A.2 Graphic Windows Hotkeys

**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, Vi's	Zoom out
<b>Ctrl + +</b>	Single Line Graphic, Block Diagrams, Vi's	Zoom in
<b>Ctrl + Scrolling</b>	Single Line Graphic, Block Diagrams, Vi's	Zoom in/out
<b>Ctrl + Double-click</b>	Busbar system	Open detailed graphic of substation
<b>Press Mouse Scroll Wheel + Moving</b>	Single Line Graphic, Block Diagrams, Vi's	Panning, Moving the visi- ble part of the graphic
<b>Alt + Area selec- tion</b>		Only textboxes inside the rubber band are marked, no parent objects
<b>Alt + Left-click</b>	Textbox	Textbox and Parent- Object are marked

Combination	Where/When	Description
<b>Ctrl + A</b>		All elements 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
<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

Combination	Where/When	Description
<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 A.2.1: Graphic Window Hotkeys

## A.3 Data Manager Hotkeys

Combination	Where/When	Description
<b>Alt + F4</b>		Close Data Manager
<b>Alt + Return</b>	Right; Link	Open the edit dialog of the element
<b>Backspace</b>		Jump one directory up
<b>Pag (arrow: up)</b>	Right	Scroll a page up
<b>Pag (arrow: down)</b>	Right	Scroll a page down
<b>Ctrl + (arrow: up)</b>	Edit dialog open	Call the edit dialog of the next object from the list and closes the current dialog
<b>Ctrl + (arrow: down)</b>	Edit dialog open	Call the edit dialog of the previous object from the list and closes the current dialog
<b>Ctrl + A</b>	Right	Mark all
<b>Ctrl + B</b>	Detail-Mode	Change to next tab
<b>Ctrl + C</b>	Marked object, marked symbol	Copy marked object
<b>Ctrl + C</b>	Marked cell	Copy the value of the marked cell
<b>Ctrl + D</b>		Change between normal and detail mode
<b>Ctrl + F</b>		Call the Filter dialog
<b>Ctrl + G</b>	Right	Go to line
<b>Ctrl + I</b>	Right	Call the dialog Select Element, in order to insert a new object. The object class depends on the current position
<b>Ctrl + Left-click</b>		Select the object
<b>Ctrl + M</b>		Mark element in diagram(s)

Combination	Where/When	Description
<b>Ctrl + O</b>		Change between the display of out of service and no relevant objects for calculation
<b>Ctrl + Q</b>	Right; station, Busbar or element with a connection	Open the station graphic
<b>Ctrl + Q</b>	Right, element with more than one connection	Call the dialog Select Station, which lists all the connected stations
<b>Ctrl + R</b>	Project	Activate the project
<b>Ctrl + R</b>	Study case	Activate study case
<b>Ctrl + R</b>	Grid	Add the grid to the study case
<b>Ctrl + R</b>	Variant	Insert the variant to the current study case, if the corresponding grid is not in the study case
<b>Ctrl + Tab</b>	Detail-Modus	Change to next tab
<b>Ctrl + V</b>		Insert the content of the clipboard
<b>Ctrl + W</b>		Change the focus between right and left side
<b>Ctrl + X</b>	Marked object, marked symbol	Cut object
<b>End</b>	Right	Jump to the last column of the current row
<b>Del</b>	Right, symbol	Delete marked object
<b>Del</b>	Right, cell	Delete the content of the cell
<b>Esc</b>	Right; after change in the line	Undo the change
<b>F2</b>	Right; cell	Change to edit mode
<b>F4</b>		Activate/Deactivate Drag&Drop-Mode
<b>F5</b>		Update
<b>F8</b>	Right, Graphic	Open the graphic
<b>Pos1</b>	Right	Jump to the first column of the current row
<b>Return</b>	Right	Call the edit dialog of the marked object
<b>Return</b>	Right; after change in the line	Confirm changes
<b>Shift + Left-click</b>		Select all the objects between the last marked object and the clicked row

Table A.3.1: Data Manager Hotkeys

## A.4 Dialog Hotkeys

Combination	Where/When	Description
<b>Ctrl + A</b>	Input field	Mark the content
<b>F1</b>		Online help

Table A.4.1: Dialog Hotkeys

## A.5 Output Window Hotkeys

Combination	Where/When	Description
<b>Pag (arrow: up)</b>		Page up
<b>Pag (arrow: down)</b>		Page down
<b>Ctrl + A</b>		Mark the content of the output window
<b>Ctrl + Pag (arrow: up)</b>		Like Ctrl + Pos1
<b>Ctrl + Pag (arrow: down)</b>		Like Ctrl + End
<b>Ctrl + C</b>		Copy the market report to the clipboard
<b>Ctrl + E</b>		Open a new empty editor
<b>Ctrl + End</b>		Set the cursor in the last position of the last row
<b>Ctrl + F</b>		Open the Search and Replace dialog
<b>Ctrl + O</b>		Call the Open dialog
<b>Ctrl + Arrow (up)</b>		Page up
<b>Ctrl + Arrow (down)</b>		Page down
<b>Ctrl + Pos1</b>		Set the cursor in the first position of first row
<b>Ctrl + Shift + End</b>		Set the cursor in the last position and marks the report in between
<b>Ctrl + Shift + home</b>		Set the cursor in the first position and marks the report in between
<b>End</b>		Set the cursor in the last position of the row
<b>F3</b>	Cursor in a Word	Jump to next same word of the current searched string
<b>Shift + F3</b>	Cursor in a Word	Jump to previous same word of the current searched string

Combination	Where/When	Description
<b>Ctrl + F3</b>	Cursor in a Word	Jump to next same word; New searched string becomes the word on which the cursor is currently positioned
<b>Arrow (up)</b>		Set the cursor one line above
<b>Arrow (right)</b>		Set the cursor one position after
<b>Arrow (down)</b>		Set the cursor one line below
<b>Arrow (left)</b>		Set the cursor one position before
<b>home</b>		Set the cursor to the first position of the row
<b>Shift + Pag (arrow: up)</b>		Set the cursor one page up and select the in between content
<b>Shift + Pag (arrow: down)</b>		Set the cursor one page down and select the in between content

Table A.5.1: Output Window Hotkeys

## A.6 Editor Hotkeys

Combination	Where/When	Description
<b>Ctrl + O</b>		Open file
<b>Ctrl + S</b>		Save
<b>Ctrl + P</b>		Print
<b>Ctrl + Z</b>		Undo
<b>Ctrl + C</b>		Copy
<b>Ctrl + V</b>		Paste
<b>Ctrl + X</b>		Cut
<b>Ctrl + A</b>		Select all
<b>Ctrl + R</b>		Comment selected lines
<b>Ctrl + T</b>		Uncomment selected lines
<b>Ctrl + F2</b>		Set bookmark / Remove bookmark
<b>Del</b>		Delete
<b>F2</b>		Go to next bookmark
<b>Shift + F2</b>		Go to previous bookmark
<b>F3</b>	Cursor in a word	Jump to next same word of the current searched string

Combination	Where/When	Description
<b>Shift + F3</b>	Cursor in a Word	Jump to previous same word of the current searched string
<b>Ctrl + F3</b>	Cursor in a Word	Jump to next same word; New searched string becomes the word on which the cursor is currently positioned
<b>Ctrl + F</b>		Open 'Find' dialog
<b>Ctrl + G</b>		Open 'Go to' dialog
<b>Ctrl + H</b>		Open 'Find and Replace' dialog
<b>Ctrl + Alt + T</b>		Show / Hide tabs and blanks
<b>Alt + Return</b>		Open user settings dialog on <i>Editor</i> page
<b>Backspace</b>		Delete character in front of cursor
<b>Insert</b>		Switch between insert and replace mode
<b>Arrow (right)</b>		One char right
<b>Shift + Arrow (right)</b>		Extend selection to next char right
<b>Ctrl + Arrow (right)</b>		Set cursor to beginning of next word
<b>Ctrl + Shift + Arrow (right)</b>		Extend selection to beginning of next word
<b>Arrow (left)</b>		One char left
<b>Shift + Arrow (left)</b>		Extend selection to next char left
<b>Ctrl + Arrow (left)</b>		Set cursor to beginning of previous word
<b>Ctrl + Shift + Arrow (left)</b>		Extend selection to beginning of previous word
<b>Arrow (down)</b>		One line down
<b>Shift + Arrow (down)</b>		Extend selection one line down
<b>Arrow (up)</b>		One line up
<b>Shift + Arrow (up)</b>		Extend selection one line up
<b>home</b>		Set cursor to first pos. in line
<b>Ctrl + home</b>		Set cursor to beginning of text
<b>Shift + home</b>		Extend selection to beginning of line
<b>Ctrl + Shift + home</b>		Extend selection to start of text

Combination	Where/When	Description
<b>end</b>		Set cursor to last pos. in line
<b>Ctrl + end</b>		Set cursor to end of text
<b>Shift + end</b>		Extend selection to end of line
<b>Ctrl + Shift + end</b>		Extend selection to end of text
<b>Pag (arrow: down)</b>		Set cursor one page down
<b>Shift + Pag (arrow: down)</b>		Extend selection to one page down
<b>Pag (arrow: up)</b>		Set cursor one page up
<b>Shift + Pag (arrow: up)</b>		Extend selection to one page up
<b>F1</b>		Open manual and search for word in which cursor is placed
<b>Ctrl + M</b>	Cursor at a (matched) bracket	Set cursor to matching bracket
<b>Ctrl + Shift + M</b>	Cursor at a (matched) bracket	Extend selection to matching bracket
<b>Alt + Arrow (up)</b>		Move selected line(s) up
<b>Alt + Arrow (down)</b>		Move selected line(s) down
<b>Ctrl + Space</b>	In DPL/QD-SL/Python/DSL scripts	Request Auto-completion for the word at the cursor position
<b>Ctrl Mousewheel (up)</b> +		Zoom in (increase zoom level)
<b>Ctrl Mousewheel (down)</b> +		Zoom out (decrease zoom level)
<b>Ctrl + 0</b>		Reset zoom level

Table A.6.1: Editor Hotkeys

## Appendix B

# Standard Models in *PowerFactory*

### B.1 AVR Models

Model Name	Full Name	Description
avr_AC7B	IEEE 421.5 2005 AC7B excitation system	This model is an improved version of an ac alternator with either stationary or rotating rectifiers, in which the controls have been replaced. If a PSS control is supplied, the Type PSS2B or PSS3B models are appropriate.
avr_AC8B	IEEE 421.5 2005 AC8B excitation system	This model consists of PID control, with a thorough selection of the proportional, integral, and derivative gains to guarantee the best performance for each particular generator excitation system.
avr_AC8BnoPIDlimits	IEEE 421.5 2005 AC8B excitation system	This model is similar to avr_AC8B differing only in that it includes a PID control without saturation.
avr_BBSEX1	Brown-Boveri static excitation system	Brown-Boveri model including stabilisation feedback.
avr_BUDGET	Czech proportional/integral excitation system	Manufacturer-specific model with PI control.
avr_CELIN	ELIN excitation system and PSS	Manufacturer-specific model which includes a power system stabiliser model.
avr_CELIN no pss	ELIN excitation system without PSS	Manufacturer-specific model which does not include a power system stabiliser model.
avr_DC3A	IEEE 421.5 2005 DC3A excitation system	Model used to represent old dc commutator exciters with non-continuously acting regulators.
avr_EMAC1T	AEP Rockport excitation system	Modified IEEE type AC1 excitation system which includes a second lead-lag block but excludes access to the excitation system stabiliser block output.

Model Name	Full Name	Description
avr_ESAC1A	1992 IEEE type AC1A excitation system	These models are applicable for simulating the performance of Westinghouse brushless excitation systems. They model a field-controlled, alternator-rectifier excitation system consisting of an alternator main exciter with non-controlled rectifiers. The exciter does not employ self-excitation and the voltage regulator power is taken from a source not affected by external transients. The diode characteristic in the exciter output imposes a lower limit of zero on the exciter output voltage.
avr_ESAC2A	1992 IEEE type AC2A excitation system	Models a high initial response field-controlled, alternator-rectifier excitation system in which the alternator main exciter is used with a non-controlled rectifier. The model includes two additional exciter field current feedback loops simulating exciter time constant compensation and exciter field current limiting elements, respectively. These models are applicable for simulating the performance of Westinghouse high initial response brushless excitation systems.
avr_ESAC3A	1992 IEEE type AC3A excitation system	Models a field-controlled, alternator-rectifier excitation system, which includes an alternator main exciter with non-controlled rectifiers. The model includes self-excitation and derives the voltage regulator power from the exciter output voltage.
avr_ESAC4A	1992 IEEE type AC4A excitation system	Represents an alternator-supplied rectifier excitation system with high initial response. The independent voltage regulator used by this system is not modelled, but transient loading effects on the exciter alternator are included.
avr_ESAC5A	1992 IEEE type AC5A excitation system	This is a reduced model of a brushless excitation system where the regulator is supplied by a permanent magnet generator. This model is intended to represent simplified systems with rotating rectifiers.
avr_ESAC6A	1992 IEEE type AC6A excitation system	The model is used to represent field-controlled, alternator-rectifier excitation systems using electronic voltage regulators with system-supplied. The regulator maximum output depends on the terminal voltage; the model includes an exciter field current limiter. It is particularly suitable for representation of stationary diode systems.

Model Name	Full Name	Description
avr_ESAC8B	Basler DECS model	Represents the Basler digital excitation control system voltage regulator as applied to a brushless exciter. The automatic voltage regulator consists of a PID controller implemented in a microprocessor. Because it uses a high processor rate, the design is implemented as if the controller were continuous even though it is digital.
avr_ESDC1A	1992 IEEE type DC1A excitation system	This model is widely used to represent systems with shunt dc excitors as well as systems with alternator excitors and uncontrolled shaft-mounted rectifier bridges.
avr_ESDC2A	1992 IEEE type DC2A excitation system	These models are used to represent systems with shunt dc excitors as well as systems with alternator excitors and uncontrolled shaft-mounted rectifier bridges. The voltage regulator's source of supply is the generator or auxiliary bus voltage. As a result, the regulator output limits are proportional to $u$ .
avr_ESST1A	1992 IEEE type ST1A excitation system	Models an excitation system where the supply of power to the controlled-bridge rectifier is provided by the generator terminals through a transformer to lower the voltage to an appropriate level.
avr_ESST2A	1992 IEEE type ST2A excitation system	These models of a potential source-controlled, rectifier-exciter excitation system are intended to represent systems in which excitation power is supplied through a transformer from the generator terminals (or the unit's auxiliary bus) and is regulated by a controlled rectifier. The maximum exciter voltage available from such systems is directly related to the generator terminal voltage.
avr_ESST3A	1992 IEEE type ST3A excitation system	Some static systems utilise internal quantities within the generator to form the source of excitation power. This model represents these kinds of compound source-controlled, rectifier-excitation systems which employ controlled rectifiers in the exciter output circuit.
avr_ESST4B	IEEE type ST4B potential or compounded source-controlled rectifier-exciter	This model is a variation of the ST3A model, including a PI regulator block replacing the lag-lead regulator property. Includes potential- and compound source rectifier excitation systems, and PI regulators with non-windup limits.
avr_ESURRY	Modified IEEE type AC1A excitation system	This is a modified version of the IEEE Type AC1A model including a second lead-lag block. High value gate and low value gate are neglected.

Model Name	Full Name	Description
avr_EX2000	EX2000 excitation system	Example of IEEE type AC7B alternator-rectifier excitation system applied to ac/dc rotating exciters.
avr_EXAC1	1981 IEEE type AC1 excitation system	These models are applicable for simulating the performance of Westinghouse brushless excitation systems. They model a field-controlled, alternator-rectifier excitation system consisting of an alternator main exciter with non-controlled rectifiers. The exciter does not employ self-excitation and the voltage regulator power is taken from a source not affected by external transients. The diode characteristic in the exciter output imposes a lower limit of zero on the exciter output voltage.
avr_EXAC1A	Modified type AC1 excitation system	These models are applicable for simulating the performance of Westinghouse brushless excitation systems. These model a field-controlled, alternator-rectifier excitation system consisting of an alternator main exciter with non-controlled rectifiers. Self-excitation is not applied and the voltage regulator power source is not affected by external transients.
avr_EXAC2	1981 IEEE type AC2 excitation system	These model a high initial response field-controlled, alternator-rectifier excitation system in which the alternator main exciter is used with a non-controlled rectifier. The model includes two additional exciter field current feedback loops simulating exciter time-constant compensation and exciter field current limiting elements, respectively. These models are applicable for simulating the performance of Westinghouse high initial response brushless excitation systems.
avr_EXAC3	1981 IEEE type AC3 excitation system	These model a field-controlled, alternator-rectifier excitation system, which includes an alternator main exciter with non-controlled rectifiers. The exciter employs self-excitation and the voltage regulator power is derived from the exciter output voltage. Therefore, this system has an additional non-linearity, simulated by the use of a multiplier whose inputs are the voltage regulator command signal and the exciter output voltage.

Model Name	Full Name	Description
avr_EXAC4	1981 IEEE type AC4 excitation system	These model an alternator-supplied, rectifier excitation system. This high initial response excitation system utilises a full thyristor bridge in the exciter output circuit, and the voltage regulator operates directly on these elements. The exciter alternator uses an independent voltage regulator to control its output voltage to a constant value. These effects are not modelled, but transient loading effects on the exciter alternator are included. Exciter loading effects can be accounted for by using the exciter load current and commutating reactance to modify excitation limits.
avr_EXBAS	Basler static voltage regulator feeding dc or ac rotating exciter	The model is used to represent Basler static voltage regulators whose output provides the entire field current of a shaft-mounted dc or ac exciter. The model should be used mainly for separately excited exciters where the regulator is the sole source of dc excitation to the excitation field, which may be either shaft-mounted dc or ac. The regulator power supply of this model is an auxiliary bus fed independently from the generator or is a shaft-driven permanent magnet generator, which yields an essentially constant ac voltage.
avr_EXDC2	1981 IEEE type DC2 excitation system	This model is widely used to represent systems with shunt dc exciters as well as systems with alternator exciters and uncontrolled shaft-mounted rectifier bridges. The voltage regulator's source of supply is the generator or auxiliary bus voltage.
avr_EXELI	Static PI transformer fed excitation system	This model is used to represent an all-static PI, transformer-fed excitation system. The model combines a stabiliser and an exciter unit. A PI voltage controller establishes a desired field current setpoint for a proportional current controller. The integrator of the PI controller has a follow-up input to match its signal to the present field current.
avr_EXELI no pss	Static PI transformer fed excitation system	The models simplifies the PSS function included in EXELI and takes the PSS signal coming directly from the power system stabiliser.
avr_EXNEBB	Bus- or solid-fed SCR bridge excitation type NEBB	In this model the excitation power is supplied through a transformer from the generator terminals or the station auxiliary bus, and is regulated by a controlled rectifier.
avr_EXNI	Bus- or solid-fed SCR bridge static exciter type NI	Modified version of the EXNEBB model, including a negative current logic for the bus or solid fed selection.

Model Name	Full Name	Description
avr_EXPIC1	Proportional/Integral excitation system	This is recommended to model excitation systems whose voltage regulator control element is a proportional plus integral type. Careful choice of constants can allow this model to be used to model a variety of manufacturers implementation of a PI type exciter.
avr_EXPIC1 v1	Proportional/Integral excitation system	Newer version of EXPIC1 recommended for modelling excitation systems whose voltage regulator control element is a proportional plus integral type. Careful choice of constants can allow this model to be used to model a variety of manufacturers implementation of a PI type exciter.
avr_EXST1	1981 IEEE type ST1 excitation system	These models of a potential source-controlled rectifier-exciter excitation system are intended to represent systems in which excitation power is supplied through a transformer from the generator terminals (or the unit's auxiliary bus) and is regulated by a controlled rectifier. The maximum exciter voltage available from such systems is directly related to the generator terminal voltage.
avr_EXST2	1981 IEEE type ST2 excitation system	These models of a potential source-controlled rectifier-exciter excitation system are intended to represent systems in which excitation power is supplied through a transformer from the generator terminals (or the unit's auxiliary bus) and is regulated by a controlled rectifier. The maximum exciter voltage available from such systems is directly related to the generator terminal voltage. The regulator output is added to the compounded transformer output.
avr_EXST2A	Modified 1981 IEEE ST2 excitation system	These models of a potential source-controlled, rectifier-exciter excitation system are intended to represent systems in which excitation power is supplied through a transformer from the generator terminals (or the unit's auxiliary bus) and is regulated by a controlled rectifier. The maximum exciter voltage available from such systems is directly related to the generator terminal voltage. The regulator output is multiplied with the compounded transformer output.

Model Name	Full Name	Description
avr_EXST3	1981 IEEE type ST3 excitation system	Some static systems utilise internal quantities within the generator to form the source of excitation power. Such compound source-controlled rectifier-excitation systems employing controlled rectifiers in the exciter output circuit are represented by this model. The excitation system stabiliser for these systems is provided by a series lag-lead element, represented by time constants. An inner loop field-voltage regulator is characterised by two gains and a time constants.
avr_IEEET1	1968 IEEE type 1 excitation system	This model is widely used to represent systems with shunt dc exciters as well as systems with alternator exciters and uncontrolled shaft-mounted rectifier bridges.
avr_IEEET1S	Modified IEEE type 1 excitation system	This modified version of the T1 excitation system includes a voltage-dependent maximum output limit instead of a fixed limit.
avr_IEEET2	1968 IEEE type 2 excitation system	This model is widely used to represent systems with shunt dc exciters as well as systems with alternator exciters and uncontrolled shaft-mounted rectifier bridges.
avr_IEEET3	1968 IEEE type 3 excitation system	This model is widely used to represent systems with shunt dc exciters as well as systems with alternator exciters and uncontrolled shaft-mounted rectifier bridges. The key parameter of this model is the current compounding gain which specifies the relative strength of the voltage and current inputs to the excitation power source.
avr_IEEET4	1968 IEEE type 4 excitation system	This model represents older excitation systems where the exciter is a dc machine and the voltage regulator is one of a wide variety of electromechanical devices.
avr_IEEET5	Modified 1981 IEEE type 4 excitation system	This model represents older excitation systems where the exciter is a dc machine and the voltage regulator is one of a wide variety of electromechanical devices. It has no deadband in its slow reset path.
avr_IEEEEX1	1979 IEEE type 1 excitation system and 1981 IEEE type DC1	This model is widely used to represent systems with shunt dc exciters as well as systems with alternator exciters and uncontrolled shaft-mounted rectifier bridges.
avr_IEEEEX2	1979 IEEE type 2 excitation system	This model is widely used to represent systems with shunt dc exciters as well as systems with alternator exciters and uncontrolled shaft-mounted rectifier bridges. IEEEEX2 takes the source used for the excitation system stabilising feedback as proportional to the control element output.

Model Name	Full Name	Description
avr_IEEX3	1979 IEEE type 3 excitation system	This model is widely used to represent systems with shunt dc excitors as well as systems with alternator excitors and uncontrolled shaft-mounted rectifier bridges. The key parameter of this model is the current compounding gain, $K_i$ , which specifies the relative strength of the voltage and current inputs to the excitation power source.
avr_IEEX4	1979 IEEE type 4 excitation system, 1981 IEEE type DC3 and 1992 IEEE type DC3A	This model represents older excitation systems where the exciter is a dc machine and the voltage regulator is one of a wide variety of electromechanical devices.
avr_IET1A	Modified 1968 IEEE type 1 excitation system	This model is widely used to represent systems with shunt dc excitors as well as systems with alternator excitors and uncontrolled shaft-mounted rectifier bridges.
avr_IET1B	Modified 1968 IEEE type 1 excitation system	Models of exciters developed for Consolidated Edison Company units; Ravenswood 1, 2, and 3; Poletti; and Arthur Kill.
avr_IET1S	IEEE type 1S excitation system	A simplified model of T1 which does not include under- and over-excitation voltage signals and saturation.
avr_IET5A	Modified 1968 IEEE type 4 excitation system	This model represents older excitation systems where the exciter is a dc machine and the voltage regulator is one of a wide variety of electromechanical devices.
avr_IEEX2A	1979 IEEE type 2A excitation system	This model is widely used to represent systems with shunt dc excitors as well as systems with alternator excitors and uncontrolled shaft-mounted rectifier bridges. IEEX2 takes the source used for the excitation system stabilising feedback as proportional to the exciter field current.
avrIVOEX	IVO excitation system	Manufacturer-specific model.
avr_OEX12T	Ontario Hydro IEEE type ST1 excitation system	This adaptation of the ST1 excitation system is used to increase the voltage at the terminals for generators accelerating following a fault. The maximum voltage is limited by a special fast-acting, bang-bang and continuous voltage limiter system.
avr_OEX3T	Ontario Hydro IEEE type ST1 excitation system	This is an adaptation of the ST1 excitation system including a semi-continuous and acting terminal voltage limiter.
avr_REXSY1	General-purpose rotating excitation system	Model to be used for general application in a brushless excitation system.
avr_REXSYS	General-purpose rotating excitation system	Model to be used for general application in a brushless excitation system.

Model Name	Full Name	Description
avr_SCRX	Bus or solid fed SCR bridge excitation system	This model can represent a rectifier bridge system fed either from an independent source, such as an externally supplied plant auxiliary bus, or from a transformer connected directly at the generator terminals. SCRX distinguishes between excitation rectifier systems having bidirectional current capability and those that can carry current only in the positive direction.
avr_SEXS	Simplified excitation system	This model represents no specific type of excitation system, but rather the general characteristics of a wide variety of properly-tuned excitation systems. It is particularly useful in cases where an excitation system must be represented and its detailed design is not known.
avr_ST5B	IEEE 421.5 2005 ST5B excitation system	This excitation system is a modification of the Type ST1A model, with alternative over-excitation and under-excitation inputs and supplementary limits.
avr_ST6B	IEEE 421.5 2005 ST6B excitation system	This model includes a PI voltage regulator with an inner loop field voltage regulator and pre-control which together with the delay in the feedback improve the dynamic response.
avr_ST7B	IEEE 421.5 2005 ST7B excitation system	The model represents a static potential-source excitation system which includes a PI voltage regulator. By the addition of a series phase lead-lag filter a derivative function is introduced, which is typically used with brushless excitation systems.
avr_URHIDT	Excitation system	High dam excitation system.
avr_URST5T	Excitation system	IEEE proposed type ST5B excitation system.

Table B.1.1: Global library standard AVR models [29]

## B.2 Turbine-Governor Models

Model Name	Full Name	Description
gov_BBGOV1	Brown-Boveri governor	Brown-Boveri governor model based on a PI controller and includes both speed and power feedback.
gov_BBGOV1 v1	Brown-Boveri governor	Modified version of the BBGOV1 model in which integral and proportional gain in the PI controller are equal to $K_p/T_n$ , and 1, respectively.
gov_BBGOV1B	Brown-Boveri governor	Modified version of the BBGOV1 model in which the controller consists of a proportional gain plus a first-order delay function.

Model Name	Full Name	Description
gov_CRCMVG	Cross compound turbine-governor	Represents turbines consisting of two shafts, each connected to a generator and driven by one or more turbine sections, although operated as a single unit.
gov_CRCMVGwd	Cross compound turbine-governor CRCMVG with mechanical damping	Modified version of the CRCMVG in which the mechanical damping is modelled.
gov_DEGOV	Woodward diesel governor	This is a model of an isochronous governor for a diesel engine. This model is based on a Woodward governor consisting of an electric speed sensor, a hydro-mechanical actuator, and the diesel engine. The output of the actuator is a valve position of the fuel supply. The use of this model should be restricted to diesel generators operating isolated from other synchronous sources.
gov_DEGOV1	Woodward diesel governor	This is a model of a governor for a diesel engine where droop control is used with either throttle or electric power feedback. This model is based on a Woodward governor consisting of an electric speed sensor, a hydro-mechanical actuator, and the diesel engine. The output of the actuator is a valve position of the fuel supply. The use of this model should be restricted to diesel generators operating isolated from other synchronous sources.
gov_GAST	Gas turbine-governor	Represents the principal dynamic characteristics of industrial gas turbines driving generators connected to electric power systems. The model consists of a forward path with governor time constant and a combustion chamber time constant, together with a load-limiting feedback path. The load limit is sensitive to turbine exhaust temperature, and the time constant to represent the exhaust gas measuring system is considered.
gov_GAST2A	Gas turbine-governor	This model has a more detailed representation of gas turbine dynamics than GAST. As with GAST, it is intended for operation near rated speed. The speed governor can be configured for droop or isochronous modes of operation. The temperature controller assumes control of turbine power when exhaust gas temperature exceeds its rated value.
gov_GASTWD	Gas turbine-governor	This model has the same level of detail in its representation of gas turbine dynamics as does the GAST2A. The governor system for this model is based on a Woodward governor consisting of an electric speed sensor with proportional, integral, and derivative control.

Model Name	Full Name	Description
gov_GGOV1	General Electric turbine-governor	General purpose turbine-governor model.
gov_HYGOV	Hydro turbine-governor	Represents a straightforward hydro-electric plant governor, with a simple hydraulic representation of the penstock with unrestricted head race and tail race, and no surge tank.
gov_HYGOV2	Hydro turbine-governor	This non-standard hydro turbine-governor includes the same basic permanent and temporary droop elements as HYGOV, but includes a slightly different representation of the lags within the governor hydraulic servo system and of the speed signal filtering. The penstock turbine model is very simplified and is valid only for small deviations of gate position from their initial conditions.
gov_HYGOVM	Hydro turbine-governor lumped parameter model	The lumped parameters hydraulic system model is designed to allow detailed representation of the surge tank system, including penstock dynamics, surge tank chamber dynamics and tunnel dynamics.
gov_HYGOVMmodif	Hydro turbine-governor lumped parameter model	This is a modified version of the hydro turbine-governor lumped parameter model. The model has the same characteristics as HYGOVM, however the implementation is different and each of the different elements is specifically identified.
gov_IIEEEG1	1981 IEEE type 1 turbine-governor	This is a generic model applicable to all common steam turbine configurations. Using the correct settings and neglecting time constants, the model can be used to represent a wide number of turbine configurations.
gov_IIEEEG2	1981 IEEE type 2 speed-governing model	IIEEEG2 is an alternative representation of hydro turbine speed governing systems.
gov_IIEEEG3	1981 IEEE type 3 speed-governing model	IIEEEG3 is an alternative representation of hydro turbine speed governing systems.
gov_IESGO	1973 IEEE standard turbine-governor	The general-purpose turbine-governor model is included for its compatibility with other widely-used stability programs. With careful selection of the time constants and gains, this model yields either a good representation of a reheat steam turbine or an approximate representation of a hydro plant of simple configuration.
gov_IVOGO	IVO turbine-governor	Manufacturer-specific model.

Model Name	Full Name	Description
gov_PIDGOV	Hydro turbine-governor	This hydro turbine governor model represents hydro power plants with straightforward penstock configurations and three-term electro-hydraulic governors (i.e. Woodard electronic). The model uses a simplified turbine-governor and penstock model that does not account for the variation of the water inertia effect with the gate opening.
gov_TGOV1	Steam turbine-governor	This is a simple model representing governor action and the reheater time constant effect for a steam turbine. The ratio, $T_2/T_3$ , equals the fraction of turbine power that is developed by the high-pressure turbine.
gov_TGOV2	Steam turbine-governor with fast valving	This is a fast valving model of a steam turbine which represents governor action, a reheat time constant, and the effects of fast valve closing to reduce mechanical power.
gov_TGOV3	Modified IEEE type 1 turbine-governor with fast valving	Modified IEEE type 1 turbine-governor model with fast valving. This model recognises the non-linearity between flow and valve position.
gov_TGOV4	Modified IEEE type 1 speed-governor with PLU and EVA	This is a model of a steam turbine and boiler that represents governor action, explicit valve action for both control and intercept valves, main, reheat and low-pressure steam effects, and boiler effects. Also incorporated into the model are a power load unbalance (PLU) relay and an early valve actuation (EVA) relay, which, when triggered will cause fast valving of the control or intercept valves.
gov_TGOV5	IEEE type 1 speed-governor modified to include boiler controls	Modified IEEE type 1 turbine-governor model with boiler control.
gov_TGOV5ntd	IEEE type 1 speed-governor modified to include boiler controls	This is a modified version of the TGOV5 model; the difference being the delay on the boiler control output which is not considered in this representation.
gov_TURCZT	Czech hydro or steam turbine-governor	This is a general purpose hydro and thermal turbine-governor model. A hydro turbine or steam turbine dynamic model can be selected by setting a flag to 1 for a steam turbine and to 0 for a hydro turbine. Penstock dynamic is not included in the model.
gov_TWDM1T	Tail water depression hydro governor model 1	The hydro turbine-governor model has the same basic permanent and transient droop elements as the HYGOV model, but additionally includes a representation of a tail water depression protection system.

Model Name	Full Name	Description
gov_TWDM1Tdgl	Tail water depression hydro governor model 1	In this model a different implementation in the gate control loop of the TWDM1T model is considered.
gov_TWDM2T	Hydro turbine-governor	The hydro turbine-governor model has the same basic turbine/penstock elements as the HYGOV model but additionally includes a representation for a tail water depression protection system and uses a governor PID controller.
gov_URGS3T	WSCC gas turbine	Specific gas turbine model.
gov_WEHGOV	Woodward electronic hydro governor	This model represents a Woodward Electronic hydro-governor with proportional, integral, and derivative control. The turbine is represented by a non-linear model for the penstock dynamics in a similar fashion to HYGOV, but the model includes look-up tables to allow the user to represent non-linearities in the flow versus gate position and mechanical power versus flow during steady-state operation. The model allows for the use of three feedback signals for droop: Electrical power, Gate position, and PID output.
gov_WESGOV	Westinghouse digital governor for gas turbine	Represents a model of a Woodward Electronic hydro-governor with proportional, integral, and derivative control. The turbine is represented by a nonlinear model for the penstock dynamics in a similar fashion toas HYGOV, but the model includes look-up tables to allow the user to represent non-linearities in the flow versus gate position and mechanical power versus flow during steady-state operation. The model allows for the use of three feedback signals for droop: Electrical power, Gate position, and PID output.
gov_WPIDHY	Woodward PID hydro governor	This model represents a Woodward PID hydro governor with proportional, integral and derivative control. This electric governor has advantages over the hydraulic governor with temporary droop which can provide a faster response.
gov_WPIDHYnwu	Woodward PID hydro governor	This is a modified version of the WPIDHY model in which a different implementation for the gate output limits is introduced.
gov_WSHYDD	WSCC double-derivate hydro governor	This model includes two deadbands, and also a non-linear gate/power relationship and a linearised turbine/penstock model.
gov_WSHYGP	WSCC GP hydro governor plus turbine	This is a hydro turbine-governor model with a PID controller. The penstock dynamics are similar to those of the WSHYDD hydro turbine-governor model.

Model Name	Full Name	Description
gov_WSIEG1	WECC modified 1981 IEEE type 1 turbine-governor	This modified version of the generic model IEEE type 1, includes speed and servo deadband, and a more detailed valve characteristic representation.

Table B.2.1: Global library standard turbine-governor models

## B.3 PSS Models

Model Name	Full Name	Description
pss_BEPSST	Transient excitation boosting stabiliser	Manufacturer-specific model.
pss_IEE2ST	Dual-input signal power system stabiliser	The model implements the general behaviour of a supplementary stabiliser. It allows two input signals to be summed to generate a signal which is then to be processed by the phase-lead blocks.
pss_IEEEST	1981 IEEE power system stabiliser	The basic function of this model is to make the phase of the supplementary signal lead that of the input signal. This function is handled by the two lead-lag blocks and is specified by the time constants.
pss_IVOST	IVO stabiliser	Manufacturer-specific model.
pss_OSTB2T	Ontario Hydro delta-omega power system stabiliser	Manufacturer-specific model based on shaft speed signal.
pss_OSTB5T	Ontario Hydro delta-omega power system stabiliser	Manufacturer-specific model.
pss_PSS1A	IEEE type PSS1A - P-Input power system stabiliser	The model represents a generalised form of a PSS with a reduced number of inputs.
pss_PSS1A_old	IEEE type PSS1A - P-Input power system stabiliser	Older version of model IEEE type PSS1A.
pss_PSS2A	1992 IEEE type PSS2A dual-input signal stabiliser	This model can represent a variety of stabilisers with inputs of power, speed, or frequency. For each of the two inputs, two washouts can be represented along with a transducer time constant. The indices N and M allow a "ramp-tracking" or simpler filter characteristic to be represented. Phase compensation is provided by the two lead-lag or lag-lead blocks.
pss_PSS2B	IEEE type PSS2B - dual-input stabiliser	Is a slight variation of the PSS2A model. Two additional parameters, lag time constant Ts11 and lead time constant Ts10, are included in an additional block to model stabilisers which incorporate a third lead-lag function.

Model Name	Full Name	Description
pss_PSS3B	IEEE type PSS3B - dual-input stabiliser	This model uses two input selectors to choose the input signals which will be used to derive an equivalent mechanical power signal. The maximum and minimum output from the controller may be adapted with the limit parameters VSTmax and VSTmin.
pss_PSS4B	IEEE type PSS4B - P-W input power system stabiliser	Three dedicated loops for the low-, intermediate- and high-frequency oscillations modes, are used in this delta-omega PSS to represent a multiple frequency band structure.
pss_PSSIEEE2B	IEEE type PSS2B - dual-input stabiliser	This model is a variation of the PSS2B model. A second proportional block is considered, adding an additional washout coupling factor.
pss_PTIST1	PTI microprocessor-based stabiliser	Models the microprocessor-based power system stabiliser as built by PTI. It models the inputs as derived from potential and current transformers. The algorithms convert these sampled values into a stabilising signal.
pss_PTIST3	PTI microprocessor-based stabiliser	This model is an extension of the PTIST1 model for the PTI microprocessor-based stabiliser.
pss_ST2CUT	Dual-input signal power system stabiliser	The model is identical to IEE2ST except for the way in which blocking the output is achieved. It assumes the stabiliser is either directly wired to the exciter setpoint or that an operator adjusts the input voltage setpoint signal to the stabiliser.
pss_STAB1	Speed sensitive stabiliser	This model is a subset of IEEEST responding to the shaft speed of its designated generator as its only input.
pss_STAB2A	ASEA power sensitive stabiliser	This model is a special representation of specific types of supplementary stabilising units. It produces a supplementary signal by introducing phase-lead into a signal proportional to electrical power output measured at the generator terminals.
pss_STAB3	Power sensitive stabiliser	Produces a supplementary signal by introducing phase-lead into a signal proportional to electrical power output measured at the generator terminals.
pss_STAB4	Power sensitive stabiliser	This model is a special representation of specific types of supplementary stabilising units. It produces a supplementary signal by introducing phase-lead into a signal proportional to electrical power output measured at the generator terminals.
pss_STABNI	Power sensitive stabiliser type NI (NVE)	This model is a simpler representation of specific types of supplementary stabilising units.

Model Name	Full Name	Description
ssc_STBSVC	WSCC supplementary signal for static var system	This model provides a supplementary signal for the WSCC static var compensation model CSVGN5. It receives one or two signals as input; for the first signal the user may choose electrical power, bus frequency, or accelerating power. An optional second signal is derived from a remote bus voltage, vars from the SVC into the system and the SVC current into the system.

Table B.3.1: Global library standard PSS models [29]

## B.4 Excitation Limiter Models

Model Name	Full Name	Description
uel_MNLEX1	Minimum excitation limiter	Minimum excitation limiters are provided in excitation systems to increase excitation voltage during high voltage to maintain steady-state stability.
uel_MNLEX2	Minimum excitation limiter	Minimum excitation limiters are provided in excitation systems to increase excitation voltage during high voltage to maintain steady-state stability.
uel_MNLEX3	Minimum excitation limiter	Minimum excitation limiters are provided in excitation systems to increase excitation voltage during high voltage to maintain steady-state stability.
uel_UEL1	Circular characteristic UEL	This under excitation model prevents loss of excitation using a circular characteristic limit in terms of machine reactive vs. real output power. The phasor inputs of I and V are synchronous machine terminal output current and voltage with both magnitude and phase angle of these ac quantities sensed.
uel_UEL2	Piecewise linear UEL	In this model, the UEL limit has either a straight line or multi-segment characteristic represented in terms of machine reactive vs. real power output.
oel_MAXEX1	Maximum excitation limiter	This model acts through the regulator to correct an over-excitation problem, to protect the generator field of an ac machine with automatic excitation control from overheating. Over-excitation can be caused either by failure of a component of the voltage regulator or an abnormal system condition.

Model Name	Full Name	Description
oel_MAXEX2	Maximum excitation limiter	This model allows limiting of either field voltage or field current. Its function is similar to MAXEX1, with the difference being the more accurate representation of the physics of actual maximum excitation limiters.

Table B.4.1: Global library standard excitation models [29]

## B.5 Static Compensator Models

Model Name	Full Name	Description
svc_CHESVC		SVC control model.
svc_CHESVC RB		SVC control model.
svc_CSTATT	Static compensator/condenser (STATCOM)	STATCOM control model.
svc_CSTNCT	Static compensator (STATCOM)	This static compensator model requires the base power as an input parameter.
svc_CSTNCT noSTB	Static compensator (STATCOM)	In contrast to CSTNCT, this model representation of a static compensator does not need the base to be defined as it is automatically defined.
svc_CSVGN1	SCR controlled static var source	This model represents an SCR-controlled shunt reactor and a parallel connected capacitor, if present.
svc_CSVGN3	SCR controlled static var source	The model is intended to control an SVC with TCR and fixed capacitors (no TSC) but it can be made to control a regular SVC with switchable capacitors. For this model, if the voltage magnitude deviates from nominal by "Override Voltage" per unit, the reactor will be gated either all the way on or off.
svc_CSVGN4	SCR controlled static var source	This models an SCR-controlled shunt reactor and a parallel connected capacitor, if present. This model allows the static var source to try to regulate the voltage on a remote bus.
svc_CSVGN5	WSCC controlled static var source	The features of this model include a fast override and remote bus voltage control. The fast override is activated when the voltage error exceeds a threshold value during major disturbances, such as faults near the SVC or switching.
svc_CSVGN6	WSCC controlled static var source	The features of this model include a fast override and remote bus voltage control. The fast override is activated when the voltage error exceeds a threshold value during major disturbances, such as faults near the SVC or switching. The controller includes non-windup limits.

Model Name	Full Name	Description
drp_COMP	Voltage regulator compensating model	This model compensates generator terminal voltage in general accordance with $uc = eterm - jXeIgen$ . A positive value of Xe causes the compensated voltage to correspond to that at a point outside the generator.
drp_IEEEVC	IEEE standard voltage compensating model	This model compensates the terminal voltage according to $uc = eterm + ZcIgen$ . The current, igen, is positive when leaving the generator. Accordingly, positive values of Rc and Xc in Zc ( $Rc + jXc$ ) cause the compensated voltage to appear to be the voltage at a point inside the generator.

Table B.5.1: Global library standard static compensator models

## B.6 Frames for Dynamic Models

Frame Name	Description
Cross Compound Frame HP	Frame to be used for the HP turbine-generator set including droop.
Cross Compound Frame HP no droop	Frame to be used for the HP turbine-generator set without droop.
Cross Compound Frame HP/LP	Frame to be used for the HP and LP turbine-generators set including droop.
Cross Compound Frame HP/LP 2	Frame to be used for the HP and LP turbine-generators set including droop.
Cross Compound Frame HP/LP no droop	Frame to be used for the HP and LP turbine-generators set without droop.
Cross Compound Frame HP/LP no droop 2	Frame to be used for the HP and LP turbine-generators set without droop.
Cross Compound Frame LP	Frame to be used for the LP turbine-generator set including droop.
Cross Compound Frame LP no droop	Frame to be used for the LP turbine-generator set without droop.
CrossCompound Frame General	“Master” frame combining the Cross Compound Frame HP and Cross Compound Frame LP.
SVS Frame	Standard IEEE frame using a voltage droop.
SVS Frame_no droop	Standard IEEE frame without droop.
SYM Frame	Standard IEEE frame using a voltage droop as well as under- and over-excitation limiter slots.
SYM Frame 1	Standard IEEE frame using a voltage droop as well as under- and over-excitation limiter slots.
SYM Frame 12	Standard IEEE frame using a voltage droop as well as under- and over-excitation limiter slots.
SYM Frame 2	Standard IEEE frame using a voltage droop as well as under- and over-excitation limiter slots.
SYM Frame_no droop	Simplified frame without droop (only VCO, PSS and PCU slots) plus frequency measurement.

Frame Name	Description
SYM Frame_no droop 2	Simplified frame without droop (only VCO, PSS and PCU slots) plus frequency measurement.

Table B.6.1: Global library standard composite model frames

All available frames in the standard library work with any standard control model from the library. The droop block/function refers to “voltage droop”, not to “frequency droop” which can be adjusted separately in the governor model.

## B.7 Typical Arrangements

In case no specific information about the controllers is available, the following typical configurations for the most common applications may be used.

**Steam Plant Model** In the standard library there are several models representing steam units turbine-/governors, all TGOVs, IEEEG1, IEEESGO, and others.

The SYM Frame\_no droop can be used to model thermal power plants (no combined cycle). For combined cycle there is no standard frame, so it has to be created from information relating to each specific unit. If no detailed information is available, the standard thermal model IEEG1 may be used for the governor, enabling as many stages (HP, MP and/or LP) as required; for a more generic and simplified model, TGOV1 can be also used. The AVR for this kind of application strongly depends on the excitation system of the machine, predominantly AC or static systems, except for old installations, where DC systems may be still in operation.

**Hydro Turbine Model** The most commonly-used hydro models are the HYGOV, IEEEG3, HYG3, GP-WSCC, and HYGOV4 models, all of which are used to represent the principal dynamic effects of the energy source and prime mover. Static excitation systems are commonly used for modern hydro units such as ESST1A. Any of the standard SYM Frame or SYM frame\_no droop can be used.

**Internal Combustion Model** A diesel or gas with synchronous generator is no different to any other type of synchronous generator unit, only their total inertia is usually lower, and they are usually salient pole machines. For the AVR, the ESAC8B, AC8B or EXAC1 and EXAC1A types from the library may be used. The only diesel governor available in the standard library is the DEGOV1.

## Appendix C

# ENTSO-E Dynamic Models in *PowerFactory*

### C.1 Excitation Models

Model Name	Description
Exc_AC2A	Modified IEEE AC2A alternator-supplied rectifier excitation system with different field current limit.
Exc_AC3A	Modified IEEE AC3A alternator-supplied rectifier excitation system with different field current limit.
Exc_AC4A	Modified IEEE AC4A alternator-supplied rectifier excitation system with different minimum controller output.
Exc_AC5A	Modified IEEE AC5A alternator-supplied rectifier excitation system with different minimum controller output.
Exc_AC6A	Modified IEEE AC6A alternator-supplied rectifier excitation system with speed input.
Exc_AC8B	Modified IEEE AC8B alternator-supplied rectifier excitation system with speed input and input limiter.
Exc_ANS	Italian excitation system. It represents static field voltage or excitation current feedback excitation system.
Exc_AVR1	Italian excitation system corresponding to IEEE (1968) Type 1 Model. It represents exciter dynamo and electromechanical regulator.
Exc_AVR2	Italian excitation system corresponding to IEEE (1968) Type 2 Model. It represents alternator and rotating diodes and electromechanical voltage regulators.
Exc_AVR3	Italian excitation system. It represents exciter dynamo and electric regulator.
Exc_AVR4	Italian excitation system. It represents static exciter and electric voltage regulator.
Exc_AVR5	Manual excitation control with field circuit resistance. This model can be used as a very simple representation of manual voltage control.
Exc_AVR7	IVO excitation system.

Model Name	Description
Exc_BBC	Transformer fed static excitation system (static with ABB regulator). This model represents a static excitation system in which a gated thyristor bridge fed by a transformer at the main generator terminals feeds the main generator directly.
Exc_CZ	Czech proportion/integral exciter.
Exc_DC1A	Modified IEEE DC1A direct current commutator exciter with speed input and without under-excitation limiters inputs.
Exc_DC2A	Modified IEEE DC2A direct current commutator excitors with added speed multiplier, added lead-lag, and voltage-dependent limits.
Exc_DC3A1	Modified old IEEE Type 3 excitation system.
Exc_ELIN1	Static PI transformer fed excitation system. This model represents an all-static excitation system.
Exc_ELIN2	Detailed excitation system model. This model represents an all-static excitation system.
Exc_HU	Hungarian excitation system model, with built-in voltage transducer.
Exc_IEEE_AC1A	Represents the IEEE Std 421.5-2005 field-controlled alternator-rectifier excitation systems designated Type AC1A.
Exc_IEEE_AC2A	Represents the IEEE Std 421.5-2005 high initial response field-controlled alternator-rectifier excitation system Type AC2A model. Similar to that of Type AC1A except for the inclusion of exciter time constant compensation and exciter field current limiting elements.
Exc_IEEE_AC3A	Represents the IEEE Std 421.5-2005 field-controlled alternator-rectifier excitation systems designated Type AC3A model. This model is applicable to excitation systems employing static voltage regulators.
Exc_IEEE_AC4A	Represents the IEEE Std 421.5-2005 Type AC4A alternator-supplied controlled-rectifier excitation system which is quite different from the other type ac systems.
Exc_IEEE_AC5A	Represents the IEEE Std 421.5-2005 simplified model for brushless excitation systems Type AC5A model.
Exc_IEEE_AC6A	Represents the IEEE Std 421.5-2005 field-controlled alternator-rectifier excitation systems Type AC6A model.
Exc_IEEE_AC7B	Represents the IEEE Std 421.5-2005 excitation systems Type AC7B model which consist of an ac alternator with either stationary or rotating rectifiers to produce the dc field requirements.
Exc_IEEE_AC8B	Represents the IEEE Std 421.5-2005 Type AC8B model. A PID voltage regulator with either a brushless exciter or dc exciter. The representation of the brushless exciter is similar to the model Type AC2A. This model can be used to represent static voltage regulators applied to brushless excitation systems.
Exc_IEEE_DC1A	Represents the IEEE Std 421.5-2005 field-controlled dc commutator excitors with continuously acting voltage regulators Type DC1A model.
Exc_IEEE_DC2A	Represents the IEEE Std 421.5-2005 field-controlled dc commutator excitors with continuously acting voltage regulators Type DC2A model. It differs from the Type DC1A model only in the voltage regulator output limits, which are now proportional to terminal voltage.

Model Name	Description
Exc_IEEE_DC3A	Represents the IEEE Std 421.5-2005 Type DC3A model. This model represents older systems, in particular those dc commutator excitors with non-continuously acting regulators that were commonly used before the development of the continuously acting varieties.
Exc_IEEE_DC4B	Represents the IEEE Std 421.5-2005 Type DC4B model. These excitation systems utilise a field-controlled dc commutator exciter with a continuously acting voltage regulator having supplies obtained from the generator or auxiliary bus.
Exc_IEEE_ST1A	Represents the IEEE Std 421.5-2005 Type ST1A model. This model represents systems in which excitation power is supplied through a transformer from the generator terminals (or the units auxiliary bus) and is regulated by a controlled rectifier. The maximum exciter voltage available from such systems is directly related to the generator terminal voltage.
Exc_IEEE_ST2A	Represents the IEEE Std 421.5-2005 Type ST2A model.
Exc_IEEE_ST3A	Represents the IEEE Std 421.5-2005 Type ST3A model.
Exc_IEEE_ST4B	Represents the IEEE Std 421.5-2005 Type ST4B model. This model is a variation of the Type ST3A model, with a proportional plus integral (PI) regulator block replacing the lag-lead regulator characteristic that is in the ST3A model.
Exc_IEEE_ST5B	Represents the IEEE Std 421.5-2005 Type ST5B model. This excitation system is a variation of the Type ST1A model, with alternative over-excitation and under-excitation inputs and additional limits.
Exc_IEEE_ST6B	Represents the IEEE Std 421.5-2005 Type ST6B model.
Exc_IEEE_ST7B	Represents the IEEE Std 421.5-2005 Type ST7B model. This model is representative of static potential-source excitation systems.
Exc_OEX3T	Modified IEEE Type ST1 excitation system with semi-continuous and acting terminal voltage limiter.
Exc_PIC	Proportional/integral regulator excitation system model. This model can be used to represent excitation systems with a proportional-integral (PI) voltage regulator controller.
Exc_SCRX	Simple excitation system model representing generic characteristics of many excitation systems; intended for use where negative field current may be a problem.
Exc_SEXS	General purpose rotating excitation system model. This model can be used to represent a wide range of excitation systems whose DC power source is an AC or DC generator.
Exc_ST1A	Modification of an old IEEE ST1A static excitation system without over-excitation limiter and under-excitation limiter.
Exc_ST2A	Modified IEEE ST2A static excitation system with additional lead-lag block.
Exc_ST3A	Modified IEEE ST3A static excitation system with added speed multiplier.
Exc_ST4B	Modified IEEE ST4B static excitation system with maximum inner loop feedback gain.
Exc_ST6B	Modified IEEE ST6B static excitation system with PID controller and optional inner feedbacks loop.

Model Name	Description
Exc_ST7B	Modified IEEE ST7B static excitation system without stator current limiter and current compensator inputs.
Exc_REXS	General purpose rotating excitation system model. This model can be used to represent a wide range of excitation systems whose DC power source is an AC or DC generator.
Exc_SK	Slovakian excitation system model. UEL and secondary voltage control are included in this model. When this model is used, there cannot be a separate under-excitation limiter or VAr controller model.

Table C.1.1: ENTSO-E standard excitation models

## C.2 Governor Models

Model Name	Description
Gov_CT1	General model for any prime mover with a PID governor, used primarily for combustion turbine and combined cycle units. Note: The implementation model uses <code>minstepsize</code> function, since as specified by ENTSO-E working group the model is dependent on the step size.
Gov_CT2	General governor model with frequency-dependent fuel flow limit. This model is a modification of the Gov_CT1 model in order to represent the frequency-dependent fuel flow limit of a specific gas turbine manufacturer. Note: The implementation model uses <code>minstepsize</code> function, since as specified by ENTSO-E working group the model is dependent on the step size.
Gov_GAST	Single shaft gas turbine.
Gov_GAST1	Modified single shaft gas turbine.
Gov_GAST2	Gas turbine model.
Gov_GAST3	Generic turbogas with acceleration and temperature controller.
Gov_GAST4	Generic turbogas.
Gov_GASTWD	Woodward gas turbine governor model.
Gov_Hydro1	Basic hydro turbine governor model.
Gov_Hydro2	IEEE hydro turbine governor model represents plants with straightforward penstock configurations and hydraulic-dashpot governors.
Gov_Hydro3	Modified IEEE hydro governor-turbine model. This model differs from that defined in the IEEE modelling guideline paper in that the limits on gate position and velocity do not permit wind up of the upstream signals.
Gov_Hydro4	Hydro turbine and governor. Represents plants with straightforward penstock configurations and hydraulic governors of traditional dashpot type. This model can be used to represent simple, Francis, Pelton or Kaplan turbines.
Gov_HydroDD	Double derivative hydro governor and turbine.
Gov_HydroFrancis	Detailed hydro unit - Francis model. This model can be used to represent three types of governors.

Model Name	Description
Gov_HydroIEEE_0	IEEE simplified hydro governor-turbine model. Used for mechanical-hydraulic and electro-hydraulic turbine governors, with or without steam feedback.
Gov_HydroIEEE_2	IEEE hydro turbine governor model represents plants with straightforward penstock configurations and hydraulic-dashpot governors.
Gov_HydroPelton	Detailed hydro unit - Pelton model. This model can be used to represent the dynamic related to water tunnel and surge chamber.
Gov_HydroPID	PID governor and turbine.
Gov_HydroPID2	Hydro turbine and governor. Represents plants with straightforward penstock configurations and three term electro-hydraulic governors.
Gov_HydroR	Fourth order lead-lag governor and hydro turbine.
Gov_Steam0	A simplified steam turbine governor model.
Gov_Steam1	Steam turbine governor model, based on the Gov_SteamIEEE_1 model (with optional deadband and nonlinear valve gain added).
Gov_Steam2	Simplified governor model.
Gov_SteamEU	Simplified model of boiler and steam turbine with PID governor.
Gov_SteamFV2	Steam turbine governor with reheat time constants and modelling of the effects of fast valve closing to reduce mechanical power.
Gov_SteamIEEE_1	IEEE steam turbine governor model.
Gov_SteamSGO	Simplified steam turbine governor model.
Gov_Hydro_WEH	Woodward electric hydro governor model.

Table C.2.1: ENTSO-E standard governor models

## C.3 Power System Stabiliser Models

Model Name	Description
Pss_ELIN2	Power system stabiliser typically associated with Exc_ELIN2.
Pss_IEEE_1A	Represents the IEEE Std 421.5-2005 Type PSS1A power system stabiliser model. PSS1A is the generalised form of a PSS with a single input.
Pss_IEEE_2B	Represents the IEEE Std 421.5-2005 Type PSS2B power system stabiliser model. This stabiliser model is designed to represent a variety of dual-input stabilisers, which normally use combinations of power and speed or frequency to derive the stabilising signal.
Pss_IEEE_3B	Represents the IEEE Std 421.5-2005 Type PSS3B power system stabiliser model. The PSS model PSS3B has dual inputs of electrical power and rotor angular frequency deviation. The signals are used to derive an equivalent mechanical power signal.
Pss_PSS1	Italian PSS.
Pss_PSS1A	Single input power system stabiliser. It is a modified version in order to allow representation of various vendors implementations on PSS Type 1A.

Model Name	Description
Pss_PSS2B	Modified IEEE PSS2B model. Extra lead/lag (or rate) block added at end.
Pss_PSS2ST	PTI microprocessor-based stabiliser Type 1.
Pss_PSS5	Italian PSS - detailed PSS.
Pss_PSSSB4	Power sensitive stabiliser model.
Pss_PSSSH	Model for Siemens “H infinity” power system stabiliser with generator electrical power input.
Pss_PSSSK	Slovakian power stabiliser type.

Table C.3.1: ENTSO-E standard PSS models

## C.4 Voltage Compensator Models

Model Name	Description
Vcmp_IEEE_1	Represents the terminal voltage transducer and the load compensator as defined in the IEEE Std 421.5-2005. This model is common to all excitation system models described in the IEEE Standard.
Vcmp_IEEE_2	Represents the terminal voltage transducer and the load compensator as defined in the IEEE Std 421.5-2005. This model is designed to cover reactive droop, transformer-drop or line-drop and cross-current compensation.

Table C.4.1: ENTSO-E standard voltage compensator models

## C.5 Over-Excitation Limiter Models

Model Name	Description
Oel_OEL2	Different from LimIEEEOEL, this model has a fixed pickup threshold and reduces the excitation set-point by mean of non-windup integral regulator.
Oel_X1	Field voltage over-excitation limiter.
Oel_X2	Field Voltage or Current over-excitation limiter designed to protect the generator field of an AC machine with automatic excitation control from overheating due to prolonged over-excitation.

Table C.5.1: ENTSO-E standard over-excitation limiter models

## C.6 Under-Excitation Limiter Models

Model Name	Description
Uel_IEEE1	Represents the Type UEL1 model which has a circular limit boundary when plotted in terms of machine reactive power vs. real power output.
Uel_IEEE2	The class represents the Type UEL2 which has either a straight-line or multi-segment characteristic when plotted in terms of machine reactive power output vs. real power output.
Uel_X1	Represents the Allis-Chalmers minimum excitation limiter.

Table C.6.1: ENTSO-E standard under-excitation limiter models

## C.7 Frames for ENTSO-E Dynamic Models

Frame Name	Description
Standard Synchronous Machine	Standard interconnection synchronous machine.

Table C.7.1: ENTSO-E composite model frames

## Appendix D

# The *DigSILENT* 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 *DigSILENT* Output Language can be used.

By selecting the *Format Editor* input mode, the editor is activated (see Figure D.0.1).

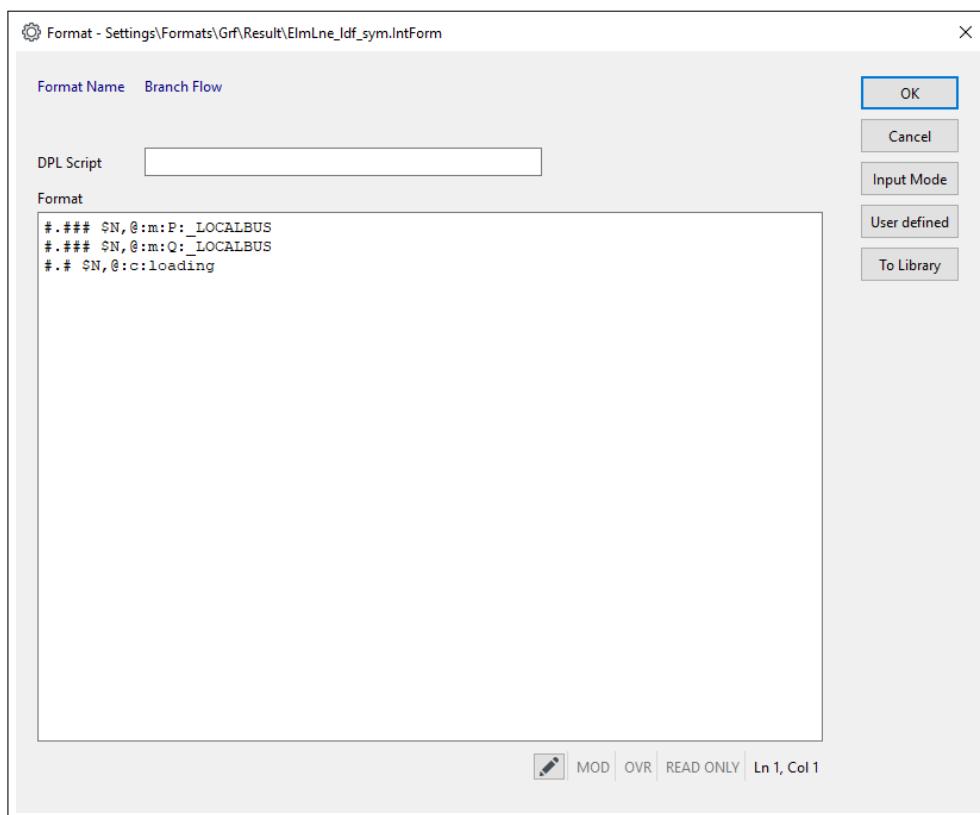


Figure D.0.1: The Form text editor

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 D.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

## D.1 Format string, Variable names and text Lines

A standard line consists of three parts (see Figure D.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 D.1.1: Basic parts of the report format

The format string is normally much longer.

## D.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 '\#'.

## D.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

## D.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 D.4.1: Colour Codes

A single item can be coloured by using the '\_COLOR(Variable name; color code)'.

## D.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.

## D.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, ..
```

## D.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)

## D.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"

# Appendix E

## Element Symbol Definition

### E.1 Introduction

The symbols used in the graphic windows of *PowerFactory* are defined by the so called 'Symbol' objects (*IntSym*). *DlgSILENT* provides a complete set of symbols to represent any of the defined network components; additionally the users have the possibility to define their own symbols and use them in the graphical windows of their projects.

In the proceeding sections the variables used to define symbol objects are presented.

### E.2 General Symbol Definition

The general definitions of the symbols are given in the *General* page of the object's dialog.

#### **Symbol Description**

The description of a symbol is shown in the list of symbols when *Show Layer...* is used and a symbol has to be selected on the page *Configuration*.

#### **Object Type**

Class name of the element which shall be represented.

#### **Type of Representation**

Branch or node object

#### **ID**

The icon ID of the icons from the graphic toolbar. If this value is set the symbol will be used when a new element is inserted. In case of '0' the symbol will not be used as default.

#### **Width/Height**

The width and height is defines the range of the fang. The marking of an element in the graphic makes this range visible.

#### **Visible**

Visibility of the symbol

#### **Mirror**

Defines if the symbol can be mirror (right mouse button entry)

#### **Allow Moving**

Allows moving in graphic

**Show Connection Attributes**

Shows the square (resulting state of composite switches) at the end of connection lines

**Insertion Reference**

Defines the insertion point of an element (e.g. rectangular terminal = 4 → top left). The following matrix describes the relation between the insertion points and the insertion numbers:

4	3	2
5	0	1
6	7	8

**Additional Attributes**

Only used for elements whose representation shall be able to alter via specific changes of the element parameters (e.g. shunts, couplers)

**Connection Points**

Defines the position on the symbol where the connection lines start. The number of connection points is defined by the number of lines unequal (-9999,-9999). The points should be located on the grid, i.e. they should be a multiple of 4.375 (mm)

**Contents**

Containing objects of type *SetVitxt* defining the layout of the text boxes. The names must be unique. Labels beginning with "Label..." and result boxes beginning with "Res...". The name of symbol must also be part of the name of the *SetVitxt*.

## E.3 Geometrical Description

The geometrical description of the symbol is given in the *Geometry* page of the dialog. The geometry can be specified by means of geometrical primitives in the 'Geometrical Components and Attributes' field.

**Circle (C,iStyle,rWidth,iFill,iColor,iRsz,nPts,rMx,rMy,rPx,rPy)**

Defines a Circle by the centre (rMx, rMy) and a point on the edge (rPx, rPy). Parameter nPts must be 2.

**Arc (A,iStyle,rWidth,iFill,iColor,iRsz,nPts,rMx,rMy,rPx1,rPy1,rPx2,rPy2)**

Defines an arc by the centre (rMx, rMy) and 2 points (rPx1, rPy1) (rPx2, rPy2) on the edge, drawn clockwise. nPts must be set to 3.

**Polyline (L, iStyle,rWidth,iFill,iColor,iRsz,iRot,nPts,rPx,rPy)**

Defines an open polygonal line with nPts points. rPx and rPy are the coordinates of peg points. iRot can be defined as:

n → rotated along with the rotation angle of the symbol

y → only rotated downwards and to the right (used in symbols)

**Polygon (G, iStyle,rWidth,iFill,iColor,iRsz,nPts {,rPx,rPy})**

Defines a closed polygonal line with nPts points. rPx and rPy are coordinates of peg points

**Text (T, iColor,iRsz,iFont,iAlign,rHeight,iOri,iRot,sString,rPx,rPy)**

Defines a text with the following attributes:

iFont	font number ( > 0 )
iAlign	insertion point (0 = left top, 2 = center)
rHeight	height ( > 0 )
iOri	orientation ( 0 = horizontal , 1 = vertical )
iRot	rotate text with object ( 0 = no, 1 = yes, 2 = vert./horiz.,3 = only to the bottom and right, – used in symbols only –)
sString	text (max. 80 characters)
rPx,rPy	coordinates of insertion point
iRsz	resize_Mode (0=not possible, 1=shift only, 2=keep ratio,3=any (RS_NONE,RS_SHIFTONLY,RS_KEEPXY, RS_FREE))

All geometrical elements have the following attributes in common:

#### **iStyle (Line style)**

- 1 = normal line
- 2 = dotted
- 3 = dashed
- 4 = dotted and dashed

#### **rWidth (Line width in mm ( > 0 ))**

#### **iFill (Fill style)**

- 0 = not filled
- 1 = filled 100%
- 2 = horiz. stripes
- 3 = vertical stripes
- 4 = horizontal and vertical stripes
- 5 = diagonal from left bottom to right top
- 6 = diagonal from right bottom to left top
- 7 = diagonal grid of stripes
- 8 = filled 25%
- 9 = filled 50%
- 10 = filled 75%

#### **iColor (Colour)**

- 1 = colour of object
- 0 = white
- 1 = black
- 2 = bright red
- 3 = bright blue
- 4 = bright green
- 5 = yellow
- 6 = cyan
- 7 = magenta
- 8 = dark grey
- 9 = grey
- 10 = red
- 11 = dark red
- 12 = dark green
- 13 = green
- 14 = dark blue
- 15 = blue
- 16 = white
- 17 = bright grey

#### **Resize mode**

- 0 = not resizable
- 1 = shift only

2 = keep ratio

3 = resizable in any direction

**iSB**

No. of area (1..32, can only be used if set in source code, e.g. vector groups)

**iLay**

No. of graphic layer

**iSN**

Connection number (0..4)

**iIP**

Object is used for calculation of intersections (=1 only for node objects)

**xOff, yOff**

Offset used when object is inserted (optional)

## E.4 Including Graphic Files as Symbols

Graphic files in WMF and bitmap format can be selected as "Symbol File". The definitions of the geometrical primitives are not used if a "Symbol File" is defined. The picture will be adapted to the size of symbol in the single line diagram. After selection of a WMF file in the top entry field for the Symbol File (not rotated) a button **Create all other files** appears which allows to create automatically WMF files in the same folder with a rotation of 90, 180 and 270 degrees. Additionally pictures for open devices with the same angles can be entered in the bottom lines.

## Appendix F

# 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	$\text{floor}(1.78)=1.000$

---

Function	Description	Example
<b>time()</b>	current simulation time	time()=0.1234
<b>pi()</b>	3.141592...	pi()=3.141592...
<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] IEC 1000-3-6 Electromagnetic Compatibility (EMC) - Part 3: Limits - Section 6: Assessment of emission limits for distorting loads in MV and HV power systems - Basic EMC publication, 1996.
- [8] IEC 61660-1 Short-circuit currents in d.c. auxiliary installations in power plants and substations, 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] IEC 1000-4-15 Electromagnetic Compatibility (EMC) - Part 4: Testing and measurement techniques - Section 15: Flickemeter - Functional and desing specifications, 2003.
- [12] D-A-CH-CZ Technical Rules for the Assessment of Network Disturbances, 2007.
- [13] BDEW Technical Guideline for Generating Plants Connected to the Medium Voltage Network, 2008.
- [14] IEC 61400-21 Wind turbines - Part 21: Measurement and assessment of power quality characteristics of grid connected wind turbines, 2008.
- [15] VDE Power generation systems connected to the low-voltage distribution network, 2011.
- [16] D-A-CH-CZ Technical Rules for the Assessment of Network Disturbances - Extension Document, 2012.
- [17] BDEW Technical Guideline for Generating Plants Connected to the Medium Voltage Network - 4th Supplement, 2013.
- [18] IEC 60909 Short-circuit currents in three-phase A.C. systems, 2016.
- [19] 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.

- [20] VDE-AR-N 4110 Technical requirements for the connection and operation of customer installations to the medium voltage network (tar medium voltage), 2018.
- [21] H. Cramér. *Mathematical Methods of Statistics (PMS-9)*, volume 9. Princeton university press, 2016.
- [22] DGUV. Thermische Gefährdung durch Störlichtbögen Hilfe bei der Auswahl der persönlichen Schutzausrüstung.
- [23] General Electric. *GE Industrial Power Systems Data Book*. General Electric, 1956.
- [24] IEEE. IEEE 1584-2002. Guide for Performing Arc-Flash Hazard Calculations.
- [25] IEEE. IEEE 1584-2018. Guide for Performing Arc-Flash Hazard Calculations.
- [26] J. E. Kolassa. *Series approximation methods in statistics*, volume 88. Springer Science & Business Media, 2006.
- [27] R. L. Heinhold. *Kabel und Leitungen für Starkstrom*. Pirelli Kabel und Systeme GmbH & Co, 2005.
- [28] NFPA. NFPA 70E. Standard for Electrical Safety. Requirements for Employee Workplaces. 2012 Edition.
- [29] IEEE Power Engineering Society. Recommended practice for excitation system models for power system stability studies (421.5 - 2005), 2005.

# 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. When working with system stages, the Base Case is considered to be the highest level in a tree of hierarchical system stage designs. .

**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 Frame. See also: DSL primitive, Composite 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 .

**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 Board Window** The graphics board window is a multi document window which contains one or more graphical pages. These pages may be single line diagrams, plots, block diagrams etc.

The graphics board shows page tabs when more than one page is present. These tabs may be used to change the visible page or to change the page order by drag&drop on the page tab.

See also: Plot, Block Diagram, Page Tab, Drag&Drop .

**Grid** A Grid is a collection of power system elements which are all stored in one so-called “Grid Folder” in the database. 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. .

**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. .

**Page Tab** Page tabs are small indexes at the edge (mostly on the top or bottom) of a multi-page window. The tabs show the titles of the pages. Left-clicking the page tab opens the corresponding page. Page tabs are used in object dialogs, which often have different pages for different calculation functions, and in the Graphics Board Window, when more than one graphical page is present. .

**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 frame. A composite model is created from a composite 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 or system stage 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, System Stage .

**System Stage** System Stages were used in previous *PowerFactory* versions and have been replaced by Variations including Expansion Stages with *PowerFactory* Version 14.  
A System Stage is an alternative design or variation for a particular grid. A system stage is stored in a system stage folder, which keeps track of all differences from the design in the higher hierarchical level. The highest level is formed by the base grid folder. It is possible to have system stages of system stages.  
See also: Grid, Base Case, Variation, Expansion Stage .

**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: System Stage, 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 Data 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, Switzerland, Colombia, Brazil, Peru, China and India. DIGSILENT services and software installations are used in more than 150 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 150 COUNTRIES



For more information, visit  
[www.digsilent.de](http://www.digsilent.de)



**DIgSILENT GmbH**  
Heinrich-Hertz-Straße 9  
72810 Gomaringen (Germany)  
T: +49 7072 9168-0  
mail@digsilent.de



DIgSILENT 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)