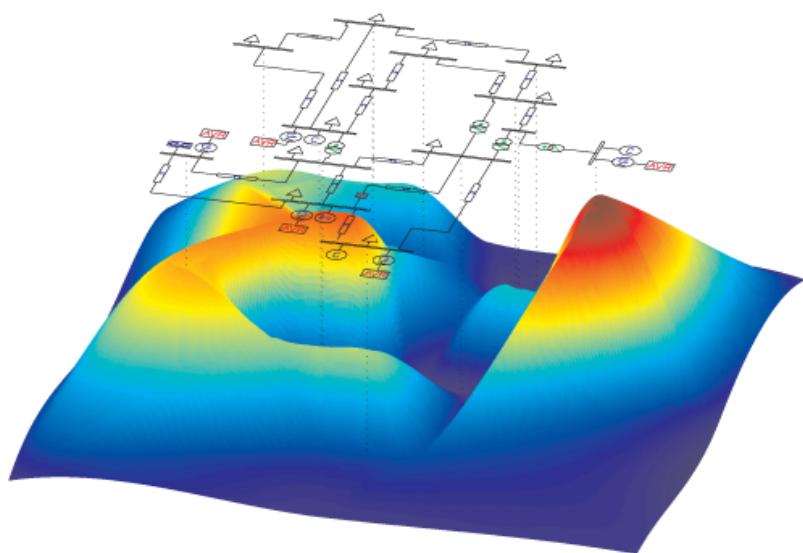


PSAT

Power System Analysis Toolbox
Documentation for PSAT version 2.1.6, May 13, 2010



Federico Milano

Copyright © 2010, 2003 Federico Milano. All rights reserved.

Ai miei genitori

Foreword

PSAT is a MATLAB toolbox for static and dynamic analysis of electric power systems. The PSAT project began in September 2001, while I was a Ph.D. candidate at the Universitá degli Studi di Genova, Italy. The first public version date back to November 2002, when I was a Visiting Scholar at the University of Waterloo, Canada. I am currently maintaining PSAT in the spare time, while I am working as associate professor at the Universidad de Castilla-La Mancha, Spain.

PSAT is provided free of charge, in the hope it can be useful and other people can use and improve it, but please be aware that this toolbox comes with ABSOLUTELY NO WARRANTY; for details type **warranty** at the MATLAB prompt. PSAT is free software, and you are welcome to redistribute it under certain conditions. Refer to the GNU General Public License (GPL) in Appendix K for details.

The Cathedral of Milan took almost six hundreds years to be completed, from 1386 to 1965. After that, renovations began. In Italy, when we wants to say that a project never ends, we say that it is like the Cathedral of Milan. PSAT is my personal tiny cathedral dome. Features, classes and data formats can be partially or completely changed in future versions. Please visit often my web-page in order to get the last version:

<http://www.uclm.es/area/gsee/Web/Federico/psat.htm>

If you find bugs, have comments or want to give your suggestions, please send me an e-mail at:

Federico.Milano@uclm.es

You can also subscribe to the PSAT Forum, which is available at:

<http://groups.yahoo.com/groups/psatforum>

Acknowledgments

I wish to thank very much Prof. Claudio Cañizares for his invaluable help, teachings and advises. Thanks also for providing me a web-page and a link to my software in the main web-page of the E&CE Department, University of Waterloo, Canada.

Many thanks to the moderators of the PSAT Forum: Tamer Abdelazim, Igor Kopcak, Juan Carlos Morataya, Raul Rabinovici, Ivo Šmon, Luigi Vanfretti, and Zhen Wang.

Thanks to Hugo M. Ayres, Marcelo S. Castro, Alberto Del Rosso, Igor Kopcak, Lars Lindgren, Juan Carlos Morataya, and Luigi Vanfretti for their relevant contributions, corrections and bug fixes.

Contents

I Outlines	1
1 Introduction	3
1.1 Overview	3
1.2 PSAT vs. Other Matlab Toolboxes	6
1.3 Outlines of the Full PSAT Documentation	6
1.4 Users	7
2 Getting Started	9
2.1 Download	9
2.2 Requirements	9
2.3 Installation	10
2.4 Launching PSAT	11
2.5 Loading Data	12
2.6 Running the Program	12
2.7 Displaying Results	14
2.8 Saving Results	14
2.9 Settings	15
2.10 Network Design	15
2.11 Tools	16
2.12 Interfaces	16
3 Notation	17
3.1 General rules	17
3.2 Frequent variables, functions and parameters	18
4 News	21
4.1 Version 2	21
4.1.1 News in version 2.1.6	21
4.1.2 News in version 2.1.5	21
4.1.3 News in version 2.1.4	22
4.1.4 News in version 2.1.3	22
4.1.5 News in version 2.1.2	22
4.1.6 News in version 2.1.1	22
4.1.7 News in version 2.1.0	22
4.1.8 News in version 2.0.0	23

4.2	Version 1	23
4.2.1	News in version 1.3.4	23
4.2.2	News in version 1.3.3	24
4.2.3	News in version 1.3.2	24
4.2.4	News in version 1.3.1	25
4.2.5	News in version 1.3.0	25
4.2.6	News in version 1.2.2	26
4.2.7	News in version 1.2.1	26
4.2.8	News in version 1.2.0	26
4.2.9	News in version 1.1.0	27
4.2.10	News in version 1.0.1	27
II	Routines	29
5	Power Flow	31
5.1	Power Flow Solvers	31
5.1.1	Newton-Raphson's Method	31
5.1.2	Fast Decoupled Power Flow	32
5.1.3	Distributed Slack Bus Model	33
5.1.4	Other Power Flow Solvers	34
5.1.5	Initialization of State Variables	34
5.2	Settings	35
5.3	<i>Example</i>	37
6	Bifurcation Analysis	47
6.1	Direct Methods	48
6.1.1	Saddle-Node Bifurcation	48
6.1.2	Limit Induced Bifurcation	48
6.2	Continuation Power Flow	49
6.2.1	Predictor Step	49
6.2.2	Corrector Step	51
6.2.3	$N - 1$ Contingency Analysis	52
6.2.4	Graphical User Interface and Settings	53
6.3	<i>Examples</i>	54
7	Optimal Power Flow	61
7.1	Interior Point Method	61
7.2	OPF Routines	62
7.2.1	Maximization of the Social Benefit	62
7.2.2	Maximization of the Distance to Collapse	62
7.2.3	Multi-Objective Optimization	64
7.2.4	Lagrangian Function	65
7.3	OPF Settings	65
7.4	<i>Example</i>	66

8 Small Signal Stability Analysis	71
8.1 Small Signal Stability Analysis	71
8.1.1 <i>Example</i>	74
8.2 Power Flow Sensitivity Analysis	77
8.2.1 <i>Example</i>	78
8.3 Graphical User Interface	81
9 Time Domain Simulation	83
9.1 Integration Methods	83
9.1.1 Forward Euler's Method	84
9.1.2 Trapezoidal Method	84
9.2 Settings	86
9.3 Output Variable Selection	88
9.4 Snapshots	89
9.5 Disturbances	91
9.6 <i>Examples</i>	92
10 PMU Placement	95
10.1 Linear Static State Estimation	95
10.2 PMU Placement Rules	96
10.3 Algorithms	96
10.3.1 Depth First	96
10.3.2 Graph Theoretic Procedure	97
10.3.3 Bisecting Search Method	97
10.3.4 Recursive Security N Algorithm	97
10.3.5 Single Shot Security N Algorithm	98
10.3.6 Recursive and Single-Shot Security $N - 1$ Algorithms	98
10.4 PMU Placement GUI and Settings	103
10.4.1 <i>Example</i>	103
III Models	107
11 Generalities	109
11.1 General Device Model	109
11.2 Common Properties	110
11.3 Common Methods	110
12 Power Flow Data	113
12.1 Bus	113
12.2 Transmission Line	114
12.3 Transformer	116
12.3.1 Two-Winding Transformer	116
12.3.2 Three-Winding Transformer	117
12.4 Slack and $v\theta$ Generator	118
12.5 PV Generator	121

12.6 PQ Load	121
12.7 PQ Generator	123
12.8 Shunt	124
12.9 Area & Region	124
13 CPF and OPF Data	127
13.1 Generator Supply	128
13.2 Generator Reserve	130
13.3 Generator Power Ramp	130
13.4 Load Demand	131
13.5 Demand Profile	132
13.6 Load Ramp	134
14 Faults & Breakers	137
14.1 Fault	137
14.2 Breaker	138
15 Measurements	141
15.1 Bus Frequency Measurement	141
15.2 Phasor Measurement Unit	142
16 Loads	147
16.1 Voltage Dependent Load	147
16.2 ZIP Load	148
16.3 Frequency Dependent Load	150
16.4 Voltage Dependent Load with Dynamic Tap Changer	151
16.5 Exponential Recovery Load	153
16.6 Thermostatically Controlled Load	154
16.7 Jimma's Load	156
16.8 Mixed Load	157
16.9 Remarks on Non-conventional Loads	159
17 Machines	161
17.1 Synchronous Machine	161
17.1.1 Order II	167
17.1.2 Order III	167
17.1.3 Order IV	168
17.1.4 Order V, Type 1	168
17.1.5 Order V, Type 2	169
17.1.6 Order V, Type 3	169
17.1.7 Order VI	170
17.1.8 Order VIII	170
17.1.9 Center of Inertia	171
17.2 Induction Machine	174
17.2.1 Order I	176
17.2.2 Order III (single cage)	176

17.2.3 Order V (double cage)	177
18 Controls	179
18.1 Turbine Governor	179
18.1.1 TG Type I	182
18.1.2 TG Type II	183
18.2 Automatic Voltage Regulator	184
18.2.1 AVR Type I	185
18.2.2 AVR Type II	186
18.2.3 AVR Type III	187
18.3 Power System Stabilizer	189
18.3.1 Type I	190
18.3.2 Type II	192
18.3.3 Type III	192
18.3.4 Type IV and V	193
18.4 Over Excitation Limiter	193
18.5 Secondary Voltage Control	197
18.6 Power Oscillation Damper	199
19 Regulating Transformers	203
19.1 Under Load Tap Changer	203
19.2 Phase Shifting Transformer	205
20 FACTS	209
20.1 SVC	209
20.2 TCSC	212
20.3 STATCOM	215
20.4 SSSC	216
20.5 UPFC	219
20.6 HVDC	225
21 Wind Turbines	231
21.1 Wind Models	231
21.1.1 Weibull's Distribution	232
21.1.2 Composite Wind Model	234
21.1.3 Mexican Hat Wavelet Model	237
21.1.4 Measurement Data	237
21.2 Wind Turbines	238
21.2.1 Constant Speed Wind Turbine	238
21.2.2 Doubly Fed Induction Generator	243
21.2.3 Direct Drive Synchronous Generator	248
22 Other Models	253
22.1 Dynamic Shaft	253
22.2 Sub-synchronous Resonance Model	255
22.3 Solid Oxide Fuel Cell	258

IV CAD	265
23 Network Design	267
23.1 Simulink Library	267
23.2 Extracting Data from Simulink Models	267
23.3 Displaying Results in Simulink Models	275
23.4 <i>Examples</i>	275
24 Block Usage	279
24.1 Block Connections	279
24.2 Standard Blocks	280
24.3 Nonstandard Blocks	282
24.3.1 Buses	282
24.3.2 Links	282
24.3.3 Breakers	282
24.3.4 Power Supplies and Demands	283
24.3.5 Generator Ramp	283
24.3.6 Generator Reserves	283
24.3.7 Non-conventional Loads	283
24.3.8 Synchronous Machines	285
24.3.9 Primary Regulators	285
24.3.10 Secondary Voltage Regulation	287
24.3.11 Under Load Tap Changers	287
24.3.12 SVCs & STATCOMs	288
24.3.13 Solid Oxide Fuel Cells	288
24.3.14 Dynamic Shafts	288
25 Block Masks	291
25.1 Blocks vs. Global Structures	291
25.2 Editing Block Masks	292
25.2.1 Mask Initialization	292
25.2.2 Mask Icon	294
25.2.3 Mask Documentation	294
25.3 Syntax of Mask Parameter Names	296
25.4 Remarks on Creating Custom Blocks	297
V Tools	299
26 Data Format Conversion	301
27 Utilities	305
27.1 Command History	305
27.2 3D Temperature Map	305
27.3 Advanced Settings	305
27.4 Sparse Matrix Visualization	308

27.5 Themes	308
27.6 Text Viewer	310
27.7 Benchmarks	310
28 Command Line Usage	315
28.1 Basics	315
28.2 Advanced Usage	318
28.3 Command Line Options	319
28.4 <i>Example</i>	320
29 PSAT on GNU Octave	323
29.1 Setting up PSAT for Running on GNU Octave	323
29.1.1 How does the conversion work?	324
29.2 Basic Commands	324
29.3 Plot Variables	325
VI Interfaces	327
30 GAMS Interface	329
30.1 Getting Started	329
30.2 GAMS Solvers	330
30.3 PSAT-GAMS Interface	330
30.4 PSAT-GAMS Models	331
30.5 Multi-period Market Clearing Model	334
30.5.1 Notation	334
30.5.2 Model Equations and Constraints	335
30.6 <i>Example</i>	337
31 UWPFLOW Interface	345
31.1 Getting Started	345
31.2 Graphical User Interface	346
31.3 Limitations and ToDos	346
31.4 <i>Example</i>	348
VII Libraries	353
32 Linear Analysis	355
32.1 Description	355
32.2 Example	357
33 Numerical Differentiation	361

VIII Appendices	363
A Global Structures & Classes	365
A.1 General Settings	365
A.2 Other Settings	371
A.3 System Properties and Settings	374
A.4 Outputs and Variable Names	380
A.5 Models	381
A.6 Command Line Usage	383
A.7 Interfaces	384
A.8 Classes	385
B Matlab Functions	387
C Other Files and Folders	393
D Third Party Matlab Code	397
E Power System Software	399
F Test System Data	401
F.1 3-bus Test System	401
F.2 6-bus Test System	402
F.3 9-bus Test System	404
F.4 14-bus Test System	406
G FAQs	411
G.1 Getting Started	411
G.2 Simulink Library	413
G.3 Power Flow	413
G.4 Optimal & Continuation Power Flow	414
G.5 Time Domain Simulation	414
G.6 Data Conversion	415
G.7 Interfaces	416
H PSAT Forum	419
I Citations & Links	423
I.1 Books	423
I.2 Ph.D. Dissertations	423
I.3 Journals	423
I.4 Conference Proceedings	424
I.5 Web-pages	425
J Letters of Reference	427
K GNU General Public License	431

CONTENTS

xv

Bibliography	456
---------------------	------------

List of Figures

1.1	PSAT at a glance.	5
1.2	PSAT around the world.	8
2.1	Main graphical user interface of PSAT.	13
5.1	Initialization of device state and internal algebraic variables.	34
5.2	GUI for general settings.	36
5.3	GUI for displaying power flow results.	38
5.4	2D visualization of power flow results.	45
5.5	3D visualization of power flow results.	46
6.1	GUI for saddle-node bifurcation settings.	49
6.2	GUI for limit-induced bifurcation settings.	50
6.3	Continuation Power Flow: tangent vector	51
6.4	Continuation Power Flow: perpendicular intersection	52
6.5	Continuation Power Flow: local parametrization	53
6.6	GUI for the continuation power flow settings.	55
6.7	GUI for plotting CPF results.	56
6.8	Nose curves for the 6-bus test system (SNB)	57
6.9	Nose curves for the 6-bus test system (LIB)	57
6.10	Nose curves for the 9-bus test system (HB)	58
7.1	GUI for OPF analysis.	66
7.2	GUI for displaying OPF results.	67
7.3	GUI for plotting OPF Pareto's sets.	70
8.1	Eigenvalue Analysis: <i>S</i> -domain.	73
8.2	Eigenvalue Analysis: <i>Z</i> -domain.	73
8.3	Eigenvalue Analysis: <i>QV</i> sensitivity.	79
8.4	GUI for the small signal stability analysis.	81
9.1	Time domain integration block diagram.	85
9.2	GUI for general settings.	87
9.3	GUI for plot variable selection.	90
9.4	GUI for snapshot settings.	91
9.5	GUI for plotting time domain simulations.	93

9.6	Generator speeds and bus voltages for the 9-bus test system.	94
10.1	PMU placement rules.	97
10.2	Flowchart of the Graph Theoretic Procedure.	98
10.3	Flowchart of the Bisecting Search.	99
10.4	Pseudo-code of the simulated Annealing Algorithm.	100
10.5	Recursive N Security Method.	101
10.6	Search of alternative placement sets.	101
10.7	Pure transit node filtering.	101
10.8	Single-Shot N Security Method.	102
10.9	Recursive $N - 1$ Security Method.	103
10.10	Single Shot $N - 1$ Security Method.	104
10.11	GUI for the PMU placement methods.	105
12.1	Transmission line π circuit.	115
12.2	Three-winding transformer equivalent circuit.	118
13.1	Example of daily demand profile.	135
15.1	Bus frequency measurement filter.	141
15.2	Phasors from sample data.	143
16.1	Measure of frequency deviation.	151
16.2	Voltage dependent load with dynamic tap changer.	152
16.3	Thermostatically controlled load.	155
16.4	Jimma's load.	157
17.1	Synchronous machine scheme.	162
17.2	Synchronous machine: block diagram of stator fluxes.	163
17.3	Field saturation characteristic of synchronous machines.	164
17.4	9-bus test system: ideal synchronous speed reference.	172
17.5	9-bus test system: COI reference.	173
17.6	Order I induction motor: electrical circuit.	176
17.7	Order III induction motor: electrical circuit.	177
17.8	Order V induction motor: electrical circuit.	178
18.1	Basic functioning of the primary frequency regulator.	181
18.2	Turbine governor type I.	182
18.3	Turbine governor type II.	183
18.4	Exciter Type I.	186
18.5	Detail of the double lead-lag block of Exciter Type I.	186
18.6	Exciter Type II.	188
18.7	Exciter Type III.	189
18.8	Power system stabilizer Type I.	192
18.9	Power system stabilizer Type II.	193
18.10	Power system stabilizer Type III.	193
18.11	Power system stabilizer Type IV.	194

18.12	Power system stabilizer Type V.	194
18.13	Over excitation limiter.	196
18.14	Secondary voltage control scheme.	198
19.1	ULTC: equivalent π circuit.	204
19.2	ULTC: secondary voltage control scheme.	204
19.3	Phase shifting transformer circuit.	207
19.4	Phase shifting transformer control scheme.	207
20.1	SVC Type 1 Regulator.	210
20.2	SVC Type 2 Regulator.	211
20.3	TCSC Regulator.	214
20.4	STATCOM circuit and control block diagram.	216
20.5	SSSC circuit.	219
20.6	SSSC control block diagram.	219
20.7	UPFC circuit.	222
20.8	UPFC phasor diagram.	222
20.9	UPFC control block diagrams.	223
20.10	HVDC scheme.	227
20.11	HVDC current control.	227
21.1	Low-pass filter to smooth wind speed variations.	232
21.2	Weibull's distribution model of the wind speed.	234
21.3	Composite model of the wind speed.	235
21.4	Mexican hat model of the wind speed.	237
21.5	Wind turbine types	239
21.6	Rotor speed control scheme.	245
21.7	Voltage control scheme of the doubly-fed induction generator.	246
21.8	Pitch angle control scheme.	246
21.9	Voltage control scheme of the direct drive synchronous generator.	250
22.1	Synchronous machine mass-spring shaft model.	254
22.2	Generator with dynamic shaft and compensated line.	256
22.3	Solid Oxide Fuel Cell scheme.	261
22.4	Solid Oxide Fuel Cell connection with the ac grid.	263
22.5	Control of the ac voltage for the Solid Oxide Fuel Cell.	263
23.1	Simulink library: Main Window.	268
23.2	Simulink library: Connections.	268
23.3	Simulink library: Power Flow data.	269
23.4	Simulink library: OPF & CPF data.	270
23.5	Simulink library: Faults & Breakers.	270
23.6	Simulink library: Measurements.	270
23.7	Simulink library: Loads.	271
23.8	Simulink library: Machines.	271
23.9	Simulink library: Regulators.	272

23.10	Simulink library: Regulating Transformers.	272
23.11	Simulink library: FACTS controllers.	273
23.12	Simulink library: Wind Turbines.	274
23.13	Simulink library: Other models.	274
23.14	GUI for Simulink model settings.	275
23.15	Simulink model of the WSCC 3-generator 9-bus test system. . . .	276
23.16	Simulink model of the IEEE 14-bus test system.	277
23.17	Simulink model of the 6-bus test system.	278
24.1	Examples of standard blocks of the PSAT SIMULINK Library.	280
24.2	Examples of allowed connections.	281
24.3	Examples of not allowed connections.	281
24.4	Examples of infeasible connections.	281
24.5	Bus block usage.	282
24.6	Breaker block usage.	283
24.7	Supply and Demand block usage.	284
24.8	Generator Ramp block usage.	284
24.9	Generator Reserve block usage.	285
24.10	Non-conventional Load block usage.	286
24.11	Synchronous Machine block usage.	286
24.12	Primary Regulator block usage.	287
24.13	Secondary Voltage Regulation block usage.	288
24.14	Under Load Tap Changer block usage.	289
24.15	SVC block usage.	289
24.16	Solid Oxide Fuel Cell block usage.	289
24.17	Dynamic Shaft block usage.	290
25.1	SIMULINK blocks vs. PSAT global structures	292
25.2	Mask GUI of a PSAT-SIMULINK block.	293
25.3	Mask initialization GUI for a PSAT-SIMULINK block.	294
25.4	Mask icon GUI of a PSAT-SIMULINK block.	295
25.5	Mask documentation GUI of a PSAT-SIMULINK block.	295
26.1	GUI for data format conversion.	302
27.1	GUI for the command history.	306
27.2	GUI for 3D map visualization.	307
27.3	GUI for advanced settings.	308
27.4	GUI for sparse matrix visualization.	309
27.5	GUI for PSAT theme selection.	310
27.6	GUI for text viewer selection.	311
27.7	Output plots of the benchmark tests.	313
28.1	Master-slave architecture.	318
29.1	Example of graph obtained using GNU OCTAVE and gplot.	326

30.1	Structure of the PSAT-GAMS interface.	332
30.2	GUI of the PSAT-GAMS interface.	333
30.3	PSAT-SIMULINK model of the three-bus test system.	338
30.4	Demand profile for the multi-period auction.	342
30.5	Multi-period auction without active power flow limits.	343
30.6	Multi-period auction with active power flow limits.	344
31.1	GUI of the PSAT-UWPFLOW interface.	347
31.2	UWPFLOW nose curves for the 6-bus test systems.	352
32.1	Linear analysis of the WSCC system: voltages at buses 6 and 7. .	359
32.2	Linear analysis of the WSCC system: reactive power of line 6-4. .	359
F.1	3-bus test system.	402
F.2	6-bus test system.	404
F.3	WSCC 3-generator 9-bus test system.	407
F.4	IEEE 14-bus test system.	409
H.1	PSAT Forum main page	420
H.2	PSAT Forum statistics	421

List of Tables

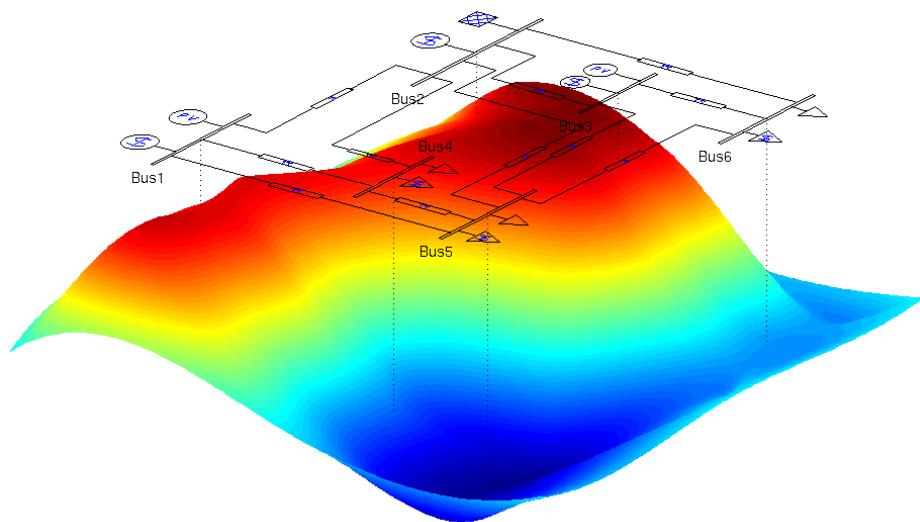
1.1	Matlab-based packages for power system analysis	6
6.1	$N - 1$ Contingency Analysis Report	59
12.1	Bus Data Format	114
12.2	Line Data Format	116
12.3	Alternative Line Data Format	117
12.4	Transformer Data Format	118
12.5	Three-Winding Transformer Data Format	119
12.6	Slack Generator Data Format	120
12.7	PV Generator Data Format	122
12.8	PQ Load Data Format	123
12.9	PQ Generator Data Format	124
12.10	Shunt Admittance Data Format	125
12.11	Area & Regions Data Format	125
13.1	Power Supply Data Format	129
13.2	Power Reserve Data Format	130
13.3	Generator Power Ramp Data Format	131
13.4	Power Demand Data Format	133
13.5	Demand Profile Data Format	135
13.6	Load Ramp Data Format	136
14.1	Fault Data Format	138
14.2	Breaker Data Format	139
15.1	Bus Frequency Measurement Data Format	142
15.2	Phasor Measurement Unit Data Format	145
16.1	Voltage Dependent Load Data Format	148
16.2	ZIP Load Data Format	149
16.3	Frequency Dependent Load Data Format	151
16.4	Typical load coefficients	152
16.5	Load with Dynamic Tap Changer Data Format	153
16.6	Exponential Recovery Load Data Format	154
16.7	Thermostatically Controlled Load Data Format	156

16.8	Jimma's Load Data Format	158
16.9	Mixed Load Data Format	159
16.10	Non-conventional Load Usage	160
17.1	Synchronous Machine Data Format	166
17.2	Reference table for synchronous machine parameters.	167
17.3	Induction Machine Data Format	175
18.1	Turbine Governor Type I Data Format	183
18.2	Turbine Governor Type II Data Format	184
18.3	Exciter Type I Data Format	187
18.4	Exciter Type II Data Format	188
18.5	Exciter Type III Data Format	190
18.6	Power System Stabilizer Data Format	191
18.7	Over Excitation Limiter Data Format	196
18.8	Central Area Controller Data Format	199
18.9	Cluster Controller Data Format	200
18.10	Power Oscillation Damper Data Format	201
19.1	Load Tap Changer Data Format	205
19.2	Phase Shifting Transformer Data Format	208
20.1	SVC Type 1 Data Format	211
20.2	SVC Type 2 Data Format	212
20.3	TCSC Data Format	215
20.4	STATCOM Data Format	217
20.5	SSSC Data Format	220
20.6	UPFC Data Format	224
20.7	HVDC Data Format	229
21.1	Wind Speed Data Format	233
21.2	Roughness length for various ground surfaces	236
21.3	Recent wind turbines	238
21.4	Constant Speed Wind Turbine Data Format	242
21.5	Doubly Fed Induction Generator Data Format	247
21.6	Direct Drive Synchronous Generator Data Format	251
22.1	Dynamic Shaft Data Format	255
22.2	SSR Data Format	259
22.3	Solid Oxide Fuel Cell Data Format	262
25.1	Mask parameter symbols	296
25.2	Example of well formed mask variable names	296
25.3	Mask parameter constants	297
28.1	Routine Conventional Names for Command Line Usage.	317
28.2	General Options for Command Line Usage.	318

28.3	Structures to be modified to change default behavior.	319
30.1	PSAT IPM-based OPF report for the three-bus test system.	339
30.2	PSAT-GAMS OPF report for the three-bus test system.	340
30.3	Input file <code>psatglobs.gms</code> for the three-bus test system.	340
30.4	Input file <code>psatdata.gms</code> for the three-bus test system.	341
30.5	Output file <code>psatsol.m</code> for the three-bus test system.	341
31.1	IEEE CDF file to be used within UWPFLOW	349
31.2	UWPFLOW power flow results	350
31.3	Input file which defines power directions in UWPFLOW	351
31.4	UWPFLOW output file with CPF results	351
33.1	State matrix eigenvalues for the WSCC 9-bus test system	362

Part I

Outlines



Chapter 1

Introduction

This chapter presents an overview of PSAT features and a comparison with other MATLAB toolboxes for power system analysis. The outlines of this documentation and a list of PSAT users around the world are also provided.

1.1 Overview

PSAT is a MATLAB toolbox for electric power system analysis and control. The command line version of PSAT is also GNU OCTAVE compatible. PSAT includes power flow, continuation power flow, optimal power flow, small signal stability analysis, and time domain simulation. All operations can be assessed by means of graphical user interfaces (GUIs) and a SIMULINK-based library provides an user friendly tool for network design.

The core of PSAT is the power flow routine, which also initialize all state and algebraic variables. After completing the power flow analysis, further static and/or dynamic analysis can be solved. These routines are:

1. Continuation power flow;
2. Optimal power flow;
3. Small signal stability analysis;
4. Time domain simulations;
5. Phasor measurement unit (PMU) placement.

In order to perform accurate power system analysis, PSAT supports a variety of static and dynamic component models, as follows:

- ◊ *Power Flow Data:* Bus bars, transmission lines and transformers, slack buses, PV generators, constant power loads, and shunt admittances.
- ◊ *CPF and OPF Data:* Power supply bids and limits, generator power reserves and ramp data, and power demand bids, limits and ramp data.

- ◊ *Switching Operations:* Transmission line faults and transmission line breakers.
- ◊ *Measurements:* Bus frequency and phasor measurement units (PMU).
- ◊ *Loads:* Voltage dependent loads, frequency dependent loads, ZIP (impedance, constant current and constant power) loads, exponential recovery loads [55, 66], voltage dependent loads with embedded dynamic tap changers, thermostatically controlled loads [57], Jimma's loads [64], and mixed loads.
- ◊ *Machines:* Synchronous machines (dynamic order from 2 to 8) and induction motors (dynamic order from 1 to 5).
- ◊ *Controls:* Turbine Governors, Automatic Voltage Regulators, Power System Stabilizer, Over-excitation limiters, Secondary Voltage Regulation (Central Area Controllers and Cluster Controllers), and a Supplementary Stabilizing Control Loop for SVCs.
- ◊ *Regulating Transformers:* Load tap changer with voltage or reactive power regulators and phase shifting transformers.
- ◊ *FACTS:* Static Var Compensators, Thyristor Controlled Series Capacitors, Static Synchronous Source Series Compensators, Unified Power Flow Controllers, and High Voltage DC transmission systems.
- ◊ *Wind Turbines:* Wind models, Constant speed wind turbine with squirrel cage induction motor, variable speed wind turbine with doubly fed induction generator, and variable speed wind turbine with direct drive synchronous generator.
- ◊ *Other Models:* Synchronous machine dynamic shaft, sub-synchronous resonance (SSR) model, and Solid Oxide Fuel Cell.

PSAT also provides a variety of utilities, as follows:

1. One-line network diagram editor (Simulink library);
2. GUIs for settings system and routine parameters;
3. GUI for plotting results;
4. Filters for converting data to and from other formats;
5. Command logs.

Finally, PSAT provides bridges to GAMS and UWPFLOW programs, which highly extend PSAT ability of performing optimization and continuation power flow analysis. Figure 1.1 depicts the structure of PSAT.

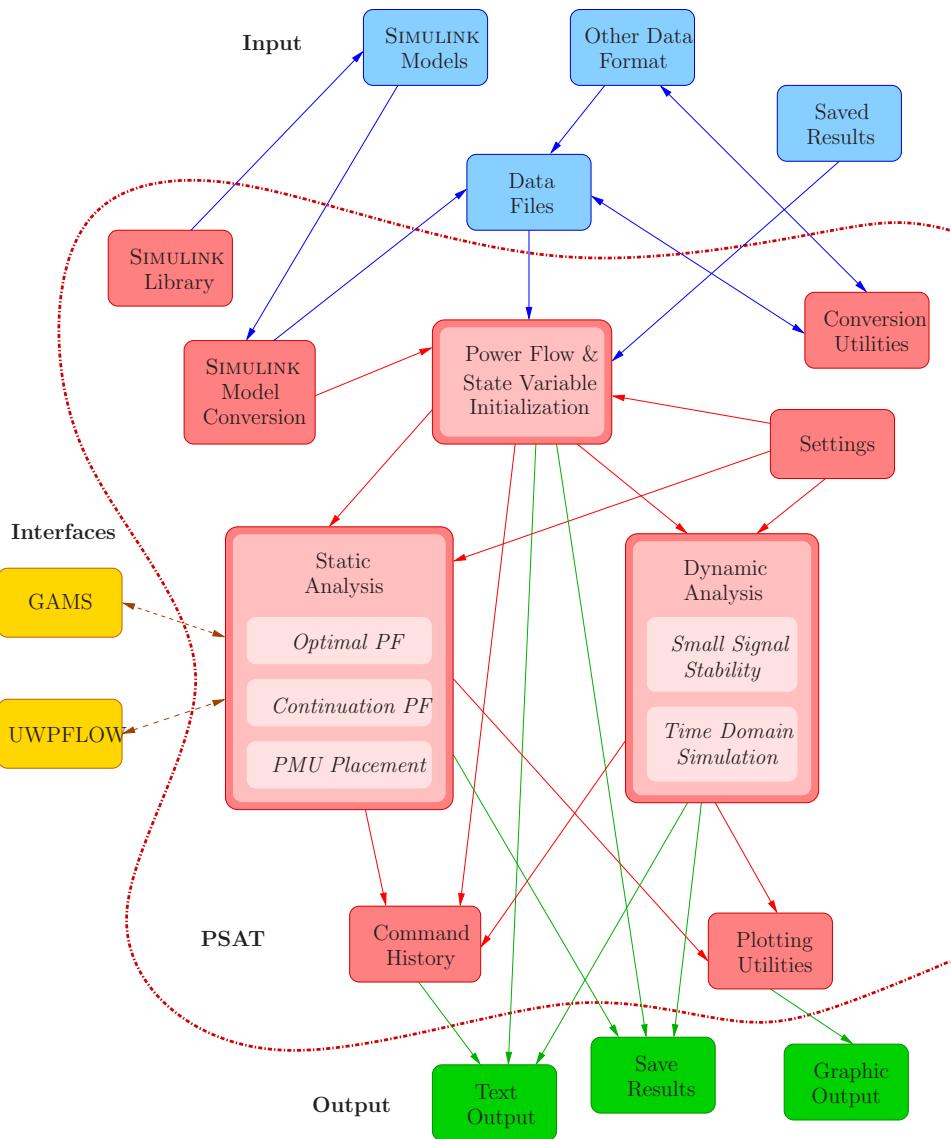


Figure 1.1: PSAT at a glance.

Table 1.1: MATLAB-based packages for power system analysis

Package	PF	CPF	OPF	SSSA	TDS	EMT	GUI	CAD
EST	✓			✓	✓			✓
MatEMTP					✓	✓	✓	✓
MATPOWER	✓		✓					
PAT	✓			✓	✓			✓
PSAT	✓	✓	✓	✓	✓		✓	✓
PST	✓	✓		✓	✓			
SPS	✓			✓	✓	✓	✓	✓
VST	✓	✓		✓	✓		✓	

1.2 PSAT vs. Other Matlab Toolboxes

Table 1.1 depicts a rough comparison of the currently available MATLAB-based software packages for power electric system analysis. These are:

1. Educational Simulation Tool (EST) [132];
2. MatEMTP [79];
3. MATPOWER [147];
4. Power System Toolbox (PST) [36, 34, 33]
5. Power Analysis Toolbox (PAT) [113];
6. SimPowerSystems (SPS) [123];¹
7. Voltage Stability Toolbox (VST) [32, 99].

The features illustrated in the table are standard power flow (PF), continuation power flow and/or voltage stability analysis (CPF-VS), optimal power flow (OPF), small signal stability analysis (SSSA) and time domain simulation (TDS) along with some “aesthetic” features such as graphical user interface (GUI) and graphical network construction (CAD).

1.3 Outlines of the Full PSAT Documentation

The full PSAT documentation consists in seven parts, as follows.

Part I provides an introduction to PSAT features and a quick tutorial, news of PSAT releases, and the notation used in throughout this manual.

Part II describes the routines and algorithms for power system analysis.

¹Since MATLAB Release 13, SimPowerSystems has replaced the Power System Blockset package.

Part III illustrates models and data formats of all components included in PSAT.

Part IV describes the SIMULINK library for designing network and provides hints for the correct usage of SIMULINK blocks.

Part V provides a brief description of the tools included in the toolbox.

Part VI presents PSAT interfaces for GAMS and UWPFLOW programs.

Part VII illustrates functions and libraries contributed by PSAT users.

Part VIII depicts a detailed description of PSAT global structures, functions, along with test system data and frequent asked questions. The GNU General Public License and the GNU Free Documentation License are also reported in this part.

1.4 Users

PSAT is currently used in more than 50 countries. These include: Algeria, Argentina, Australia, Austria, Barbados, Belgium, Brazil, Bulgaria, Canada, Chile, China, Colombia, Costa Rica, Croatia, Cuba, Czech Republic, Ecuador, Egypt, El Salvador, France, Germany, Great Britain, Greece, Guatemala, Hong Kong, India, Indonesia, Iran, Israel, Italy, Japan, Korea, Laos, Macedonia, Malaysia, Mexico, Nepal, Netherlands, New Zealand, Nigeria, Norway, Perú, Philippines, Poland, Puerto Rico, Romania, Spain, Slovenia, South Africa, Sudan, Sweden, Switzerland, Taiwan, Thailand, Tunisia, Turkey, Uruguay, USA, Venezuela, Vietnam and Zambia. Figure 1.2 depicts PSAT users around the world.

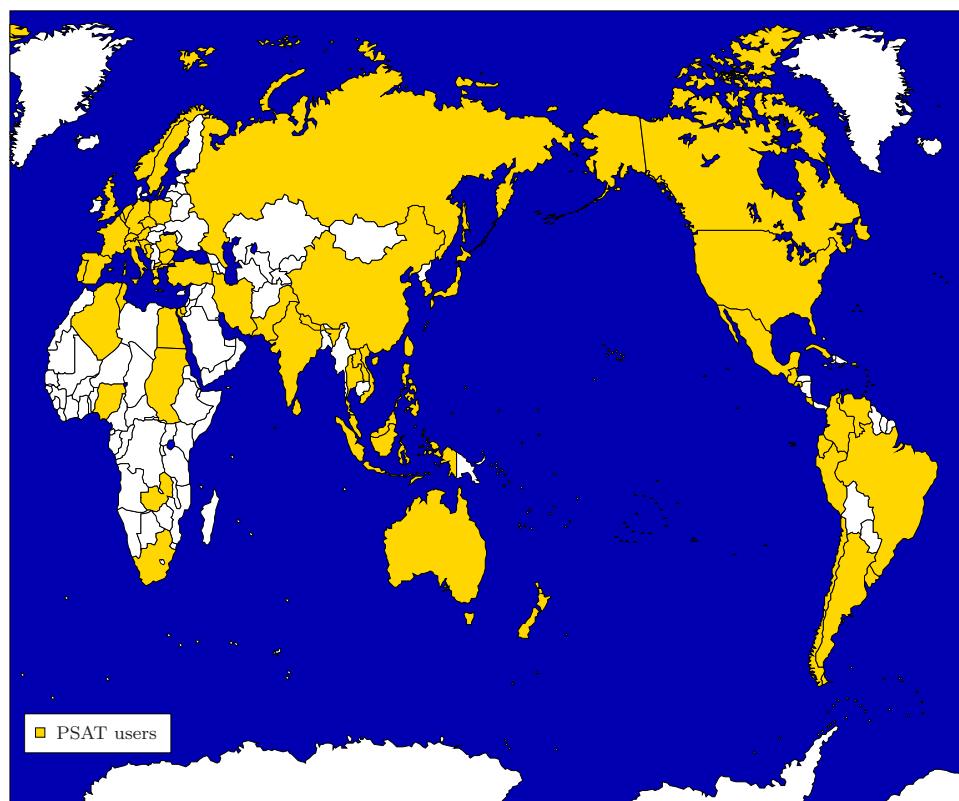


Figure 1.2: PSAT around the world.

Chapter 2

Getting Started

This chapter explains how to download, install and run PSAT. The structure of the toolbox and a brief description of its main features are also provided.

2.1 Download

PSAT can be downloaded at:

www.uclm.es/area/gsee/Web/Federico/psat.htm

or following the “Downloads” link at:

www.power.uwaterloo.ca

The latter link is kindly provided by Prof. Claudio A. Cañizares, University of Waterloo, Canada.

2.2 Requirements

PSAT 2.1.6 can run on Linux, Unix, Mac OS X, and Windows operating systems and on MATLAB versions from 5.3 to 7.6 (R2008a) and GNU OCTAVE version 3.0.0.¹ The SIMULINK library and the GUIs can be used on MATLAB 7.0 (R14) or higher. On older versions of MATLAB and on GNU OCTAVE, only the command line mode of PSAT is available. Chapters 28 and 29 provide further details on the command line usage on MATLAB and on GNU OCTAVE.

The requirements of PSAT for running on MATLAB are minimal: only the basic MATLAB and SIMULINK packages are needed.

¹ Available at www.gnu.org/software/octave

2.3 Installation

Since PSAT is a collection of scripts, the installation simply consists in expand the distribution tarball somewhere in the hard disk. For the sake of completeness, the steps are as follows.

Extract the zipped files from the distribution tarball in a new directory (DO NOT overwrite an old PSAT directory). On Unix or Unix-like environment, make sure the current path points at the folder where you downloaded the PSAT tarball and type at the terminal prompt:

```
$ gunzip psat-<version>.tar.gz
$ tar xvf psat-<version>.tar
```

or:

```
$ tar zxvf psat-<version>.tar
```

or, if the distribution archive comes in the *zip* format:

```
$ unzip psat-<version>.zip
```

where *version* is the current PSAT version code. The procedure described above creates in the working directory a `psat2` folder that contains all files and all directories necessary for running PSAT. On a Windows platform, use WinZip or similar program to unpack the PSAT tarball. Most recent releases of Windows zip programs automatically supports `gunzip` and `tar` compression and archive formats. Some of these programs (e.g. WinZip) ask for creating a temporary directory where to expand the *tar* file. If this is the case, just accept and extract the PSAT package. Finally, make sure that the directory tree is correctly created.

Then launch MATLAB. Before you can run PSAT, you need to update your MATLAB path, i.e. the list of folders where MATLAB looks for functions and scripts. You may proceed in one of the following ways:

1. Open the GUI available at the menu *File/Set Path* of the main MATLAB window. Then type or browse the PSAT folder and save the session. Note that on some Unix platforms, it is not allowed to overwrite the `pathdef.m` file and you will be requested to write a new `pathdef.m` in a writable location. If this is the case, save it in a convenient folder but remember to start future MATLAB session from that folder in order to make MATLAB to use your custom path list.
2. If you started MATLAB with the `-nojvm` option, you cannot launch the GUI from the main window menu. In this case, use the `addpath` function, which will do the same job as the GUI but at the MATLAB prompt. For example:

```
>> addpath /home/username/psat
```

or:

```
>> addpath 'c:\Document and Settings\username\psat'
```

For further information, refer to the on-line documentation of the function **addpath** or the MATLAB documentation for help.

3. Change the current MATLAB working directory to the PSAT folder and launch PSAT from there. This works since PSAT checks the current MATLAB path list definition when it is launched. If PSAT does not find itself in the list, it will use the **addpath** function as in the previous point. Using this PSAT feature does not always guarantee that the MATLAB path list is properly updated and is not recommended. However, this solution is the best choice in case you wish maintaining different PSAT versions on your disk.

Note 1: PSAT will not work properly if the MATLAB path does not contain the PSAT folder.

Note 2: PSAT needs four internal folders (**images**, **build**, **themes**, and **filters**). It is highly recommended not to change the position and the names of these folders. PSAT can work properly only if the current MATLAB folder and the data file folders are writable.

Note 3: To be able to run different PSAT versions, make sure that your **pathdef.m** file does not contain any PSAT folder. You should also reset the MATLAB path or restart MATLAB any time you wish to change PSAT version.

2.4 Launching PSAT

After setting the PSAT folder in the MATLAB path, type from the MATLAB prompt:

```
>> psat
```

This will create all the classes and the structures required by the toolbox, as follows:²

```
>> who
```

Your variables are:

Algeb	Demand	Jimma	PQ	SAE1	Sssc	Upfc
Area	Dfig	LIB	PQgen	SAE2	Statcom	Varname
Breaker	Exc	Line	PV	SAE3	State	Varout

²By default, all variables previously initialized in the workspace are cleared. If this is not desired, just comment or remove the **clear all** statement at the beginning of the script file **psat.m**.

Bus	Exload	Lines	Param	SNB	Supply	Vltn
Buses	Fault	Ltc	Path	SSR	Svc	Wind
Busfreq	Fig	Mass	Phs	SSSA	Syn	Ypdः
CPF	File	Mixed	Pl	SW	Tap	ans
Cac	F1	Mn	Pmu	Servc	Tcsc	clpsat
Cluster	GAMS	Mot	Pod	Settings	Tg	filemode
Comp	Hdl	NLA	Pss	Shunt	Theme	jay
Cswt	History	OPF	Rmpg	Snapshot	Thload	
DAE	Hvdc	Oxl	Rmpl	Sofc	Twt	
Ddsg	Initl	PMU	Rsrv	Source	UWPFLOW	

and will open the main user interface window³ which is depicted in Fig. 2.1. All modules and procedures can be launched from this window by means of menus, push buttons and/or shortcuts.

2.5 Loading Data

Almost all operations require that a data file is loaded. The name of this file is always displayed in the edit text *Data File* of the main window. To load a file simply double click on this edit text, or use the first button of the tool-bar, the menu *File/Open/Data File* or the shortcut <Ctr-d> when the main window is active. The data file can be either a *.m* file in PSAT format or a SIMULINK model created with the PSAT library.

If the source is in a different format supported by the PSAT format conversion utility, first perform the conversion in order to create the PSAT data file.

It is also possible to load results previously saved with PSAT by using the second button from the left of the tool-bar, the menu *File/Open/Saved System* or the shortcut <Ctr-y>. To allow portability across different computers, the *.out* files used for saving system results include also the original data which can be saved in a new *.m* data file. Thus, after loading saved system, all operations are allowed, not only the visualization of results previously obtained.

There is a second class of files that can be optionally loaded, i.e. perturbation or disturbance files. These are actually MATLAB functions and are used for setting independent variables during time domain simulations (refer to Chapter 9 for details). In order to use the program, it is not necessary to load a perturbation file, not even for running a time domain simulation.

2.6 Running the Program

Setting a data file does not actually load or update the component structures. To do this, one has to run the power flow routine, which can be launched in several ways from the main window (e.g. by the shortcut <Ctr-p>). Refer to Chapter 5

³This window should always be present during all operations. If it is closed, it can be launched again by typing `fm_main` at the prompt. In this way, all data and global variables are preserved.

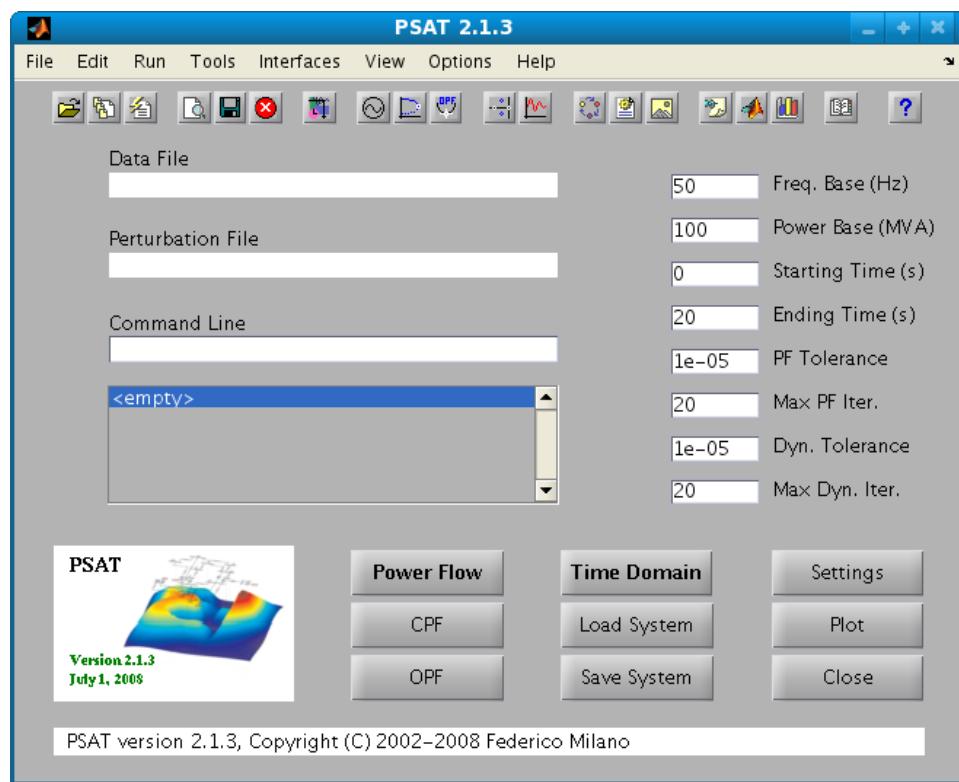


Figure 2.1: Main graphical user interface of PSAT.

for details. The last version of the data file is read each time the power flow is performed. The data are updated also in case of changes in the SIMULINK model originally loaded. Thus it is not necessary to load again the file every time it is modified.

After solving the first power flow, the program is ready for further analysis, such as Continuation Power Flow (Chapter 6), Optimal Power Flow (Chapter 7), Small Signal Stability Analysis (Chapter 8), Time Domain Simulation (Chapter 9), PMU placement (Chapter 10), etc. Each of these procedures can be launched from the tool-bar or the menu-bar of the main window.

2.7 Displaying Results

Results can be generally displayed in more than one way, either by means of a graphical user interface in MATLAB or as a ASCII text file. For example power flow results, or whatever is the actual solution of the power flow equations of the current system, can be inspected with a GUI (in the main window, look for the menu *View/Static Report* or use the shortcut <Ctr-v>). Then, the GUI allows to save the results in a text file. The small signal stability and the PMU placement GUIs have similar behaviors. Other results requiring a graphical output, such as continuation power flow results, multi-objective power flow computations or time domain simulations, can be depicted and saved in .eps files with the plotting utilities (in the main window, look for the menu *View/Plotting Utilities* or use the shortcut <Ctr-w>). Refer to the chapters where these topics are discussed for details and examples.

When working, PSAT displays several messages on the MATLAB prompt and in the banner at the bottom of the main GUI. These messages are stored in the *History* structure. By double clicking on the banner or using the menu *Options/History* of the main GUI, a user interface will display the last messages. This utility can be useful for debugging data errors or for checking the performances of the procedures.⁴

2.8 Saving Results

The menu *File/Save/Current System* or the shortcut <Ctr-a> can be invoked at any time for saving the actual system status in a .mat file. All global structures used by PSAT are stored in this file that is placed in the folder of the current data file and has the extension .out. Also the data file itself is saved, to ensure full portability.

Static analyses allow creating a report in a text file that can be stored and used later. The extensions for these files are as follows:

⁴All errors displayed in the command history are not actually errors of the program, but are due to wrong sequence of operations or inconsistencies in the data. On the other hand, errors and warnings that are displayed on the MATLAB prompt are more likely bugs and it would be of great help if you could report these errors to the PSAT web forum whenever you encounter one (see Appendix H).

- .txt for reports in plain text;
- .htm for reports in HTML;
- .xls for reports in Excel;
- .tex for reports in L^AT_EX.

The report file name are built as follows:

[data_file_name]_[xx].[ext]

where xx is a progressive number, thus previous report files will not be overwritten.⁵ All results are placed in the folder of the current data file, thus it is important to be sure to have the authorization for writing in that folder.

Also the text contained in the command history can be saved, fully or in part, in a [data_file_name]_[xx].log file.

2.9 Settings

The main settings of the system are directly included in the main window an they can be modified at any time. These settings are the frequency and power bases, starting and ending simulation times, static and dynamic tolerance and maximum number of iterations. Other general settings, such as the fixed time step used for time domain simulations or the setting to force the conversion of PQ loads into constant impedances after power flow computations, can be modified in a separate windows (in the main window, look for the menu *Edit/General Settings* or use the shortcut <Ctrl-k>). All these settings and data are stored in the **Settings** structure which is fully described in Appendix A. The default values for some fields of the **Settings** structure can be restored by means of the menu *Edit/Set Default*. Customized settings can be saved and used as default values for the next sessions by means of the menu *File/Save/Settings*.

Computations requiring additional settings have their own structures and GUIs for modifying structure fields. For example, the continuation power flow analysis refers to the structure **CPF** and the optimal power flow analysis to the structure **OPF**. These structures are described in the chapters dedicated to the corresponding topics.

A different class of settings is related to the PSAT graphical interface appearance, the preferred text viewer for the text outputs and the settings for the command history interface. These features are described in Chapter 27.

2.10 Network Design

The SIMULINK environment and its graphical features are used in PSAT to create a CAD tool able to design power networks, visualize the topology and change the

⁵Well, after writing the 99th file, the file with the number 01 is actually overwritten without asking for any confirmation.

data stored in it, without the need of directly dealing with lists of data. However, SIMULINK has been thought for control diagrams with outputs and inputs variables, and this is not the best way to approach a power system network. Thus, the time domain routines that come with SIMULINK and its ability to build control block diagrams are not used. PSAT simply reads the data from the SIMULINK model and writes down a data file.

The library can be launched from the main window by means of the button with the SIMULINK icon in the menu-bar, the menu *Edit/Network/Edit Network/Simulink Library* or the shortcut <Ctr-s>. A full description of this library and its interactions with the rest of the program is presented in Chapter 23.

2.11 Tools

The most important tool provided with PSAT is the set of filters for importing data from other power system software packages (see Chapter 26). Be aware that, in some cases, the conversion cannot be complete since data defined for commercial software have more features than the ones implemented in PSAT. PSAT static data files can be converted into the IEEE CDF, EPRI and ODM formats.

Other PSAT tools and utilities, such as the command history, the sparse matrix visualization GUI, the theme selector, and the text viewer selector are briefly described in Chapter 27.

2.12 Interfaces

PSAT provides interfaces to GAMS and UWPFLOW, which highly extend PSAT ability to perform OPF and CPF analysis respectively.

The General Algebraic Modeling System (GAMS) is a high-level modeling system for mathematical programming problems. It consists of a language compiler and a variety of integrated high-performance solvers. GAMS is specifically designed for large and complex scale problems, and allows creating and maintaining models for a wide variety of applications and disciplines [17]. Refer to Chapter 30 for a more detailed description of the routine and the GUI which interfaces PSAT to GAMS.

UWPFLOW is an open source program for sophisticated continuation power flow analysis [22]. It consists of a set of C functions and libraries designed for voltage stability analysis of power systems, including voltage dependent loads, HVDC, FACTS and secondary voltage control. Refer to Chapter 31 for a more detailed description of the PSAT-UWPFLOW interface, which allows exporting PSAT models to UWPFLOW. The interface is currently in an early stage; refer to Section 31.3 for limitations and ToDos.

Chapter 3

Notation

This chapter describes, from a general point of view, the notation used throughout this manual. The meaning of specific symbols is given the first time they appear in the text.

3.1 General rules

The general rules are as follows.

1. Scalar functions, variables and parameters expressed in p.u. are in lower case, italic face, Latin characters. For example: x, v, p .
2. Scalar angles expressed in rad/s are typically in lower case, normal face, Greek characters. For example θ, δ, α .
3. Scalar variables and parameters expressed in absolute values are generally in upper case, italic face, Latin characters. For example S_N (MVA), T_1 (s).
4. Vectors of functions or are in lower case, bold italic face. For example $\mathbf{f}, \mathbf{g}, \mathbf{x}$.
5. Matrices are in upper case, bold italic face. For example $\mathbf{J}_{LF}, \mathbf{I}_n, \mathbf{H}$.
6. Jacobian matrices are in lower case, bold italic face with a sub-index that indicates the variables with respect to which derivatives are computed. For example $\mathbf{g}_y = \nabla_y^T \mathbf{g}$.
7. Hessian matrices are as Jacobian ones but with two sub-indexes. For example $\mathbf{g}_{yy} = \nabla_y^T \mathbf{g}_y \mathbf{g}$.
8. To maintain a light notation, the use of indexes is avoided wherever is possible. Device equations implicitly imply an index i in all variables and parameters, $\forall i = 1, \dots, n$, where n is the number of elements of the same device. Indexes are also implicit in summations, if their lack does not lead to confusion.

3.2 Frequent variables, functions and parameters

f differential equations.

f_n nominal frequency (Hz).

f_x Jacobian matrix $\nabla_x^T f$.

f_y Jacobian matrix $\nabla_y^T f$.

g algebraic equations.

g_x Jacobian matrix $\nabla_x^T g$.

g_y Jacobian matrix $\nabla_y^T g$.

K control loop gain.

k_G distributed slack bus variable.

p active powers.

q reactive powers.

S_n nominal power (MVA).

t time.

T time constant.

u controlled variables.

v bus voltage magnitudes.

V_n nominal voltage (kV).

w left eigenvector.

x state variables.

\dot{x} first time derivatives of state variables.

y algebraic variables.

z state and algebraic variables $[x^T, y^T]^T$.

δ generator rotor angles.

θ bus voltage angles.

Θ temperature.

ν right eigenvector.

λ loading parameter.

- τ shaft torque of rotating machines.
- ψ differential and algebraic equations $[f^T, g^T]^T$.
- ω generator rotor speeds.

Chapter 4

News

This chapter briefly presents the new features of the current PSAT version. If not otherwise specified, each new release includes all changes of the previous ones.

4.1 Version 2

Version 2 marks a new and mature stage in the development of PSAT. The whole code has been rethought and rewritten. Classes and object oriented programming techniques are now extensively used throughout the program. The rewriting process has taken almost three years and has now reached a reasonable stable version. Nevertheless, the “to do” list is still huge...

To enumerate all changes with respect to the previous versions 1.x is impossible and, likely, useless. Subsection 4.1.8 lists only most important changes.

4.1.1 News in version 2.1.6

1. PSAT is now compatible with Matlab 7.10 (R2010a). Backward compatibility with older versions is maintained.
2. PSS models have been fixed to support the off-line status.
3. Fixed a bug in the induction machine device of the GNU Octave variant of PSAT.

4.1.2 News in version 2.1.5

1. *S*-domain eigenvalue loci include plots of 5%, 10% and 15% damping ratio.
2. Improved error and warning messages during initialization of dynamic components.
3. Minor changes in the ULTC, phase shifting transformers and synchronous machines models.

4. Corrected some bugs in FACTS device models.

4.1.3 News in version 2.1.4

1. Added the Mexican hat wavelet wind model based on the standard IEC 61400-1 [59].
2. Added the filter for the format REDS (REpository for Distribution Systems), for distribution networks. See also Appendix E.
3. Fixed a few bugs in device classes.

4.1.4 News in version 2.1.3

1. Several bugs and inconsistencies of the previous version have been fixed.
2. The PSAT full documentation has been extensively revised.

4.1.5 News in version 2.1.2

1. Fixed some bugs of the function that generates static reports of the CPF analysis.
2. Fixed a bug of the CPF routine that showed up when static PQ loads were put off-line (e.g. one or more `PQ.u` are zero).
3. Fixed some bugs related to the visualization of snapshots after the CPF analysis.

4.1.6 News in version 2.1.1

1. Corrected a bug in the model of the AVR type 1: time constants T_3 and T_4 were swapped in the equations.
2. The Subsection 18.2.1 about AVR type I has been corrected and improved.
3. Added the field `resetangles` to the structure `Settings` (see A.1). The issue is the initialization of voltage angles and magnitudes for computing the point right after the fault clearing during time domain simulations. The default behavior of PSAT can be changed by setting `Settings.reset.angles` to 1. In some cases, this can improve the convergence after clearing the fault.

4.1.7 News in version 2.1.0

1. PSAT 2.1.0 can run on any MATLAB version back to 5.3 and on GNU OCTAVE.
2. A GUI for 3D visualization for static and dynamic power system analyses has been added (see the example of Chapter 5).

3. The library for Numerical Linear Analysis has been completely rewritten and split into a Linear Analysis library (see Chapter 32) and a Numerical Differentiation library (see Chapter 33).
4. A filter for the ODM (open data model) has been included.
5. The GUIs have been optimized for running on Quartz (Mac OS X) and X11 (Linux) graphical systems.

4.1.8 News in version 2.0.0

1. The whole PSAT code has been rethought and rewritten using classes and object oriented programming techniques. Each device is defined by a class with attributes and methods.
2. The algorithms of PF, CPF, OPF, SSSA and TD have been rewritten, improved and made more robust.
3. The SIMULINK library has been renewed using “physical” components. This avoids the directionality of control blocks and allows producing high quality network schemes.
4. Added the *status* field to most components. A component can be put on-line or off-line by toggling its status.
5. New more reliable versions of TCSC, SSSC and UPFC devices and Power Oscillations damper model has been provided by H. Ayres, M. S. Castro and A. Del Rosso. The HVDC model has also been rewritten.
6. Several new filters for data format conversion have been added. Most filters has been provided by J. C. Morataya.
7. PSAT has been tested with very large static and dynamic networks (up to 15000 buses).
8. The logo of PSAT has been changed.

4.2 Version 1

The set of versions 1.x constitutes the first stage in the development of PSAT. With version 1.3.4 the program has reached a stable but hard to improve level. Version 1 has been frozen as it is and will not be maintained any longer.

4.2.1 News in version 1.3.4

1. Added unit commitment and multi-period market clearing models for the PSAT-GAMS interface (see Section 30.5).

2. Added the Phasor Measurement Unit (PMU) model (Section 15.2); the Jimma's load model (Section 16.7); and a Mixed Load model (Section 16.8).
3. Added a filter to convert NEPLAN data file (see Chapter 26).
4. Added the possibility of exporting plots as MTV plot files and as MATLAB scripts.
5. Added a better step control for the continuation power flow analysis. The step control can be disabled (menu Options of the CPF GUI), resulting in faster but likely imprecise continuation analysis.
6. Added the option of stopping time domain simulations when the machine angle degree is greater than a given $\Delta\delta^{\max}$ (default value 180°).
7. Added the function `fm_connectivity` to detect separation in areas following a breaker operation during time domain simulation (by courtesy of Laurent Lenoir).
8. Added a check during the initialization of synchronous machines to see if a PV or slack generator are connected to the machine bus. In the case that no PV or slack generator are found a warning message is displayed. Observe the initialization routine does not fail, but the machine is likely not properly initialized.
9. Patched the `fm_sim` function. It is now allowed using bus names with carriage return characters.
10. Several minor function patches.

4.2.2 News in version 1.3.3

1. Minor release with a few bug fixes and a revision of PSAT documentation.
2. The *linear* recovery load has been renamed *exponential* recovery load in order to be consistent with the definition given in [66]. The corresponding component structure has been renamed `Exload`.

4.2.3 News in version 1.3.2

1. First release fully tested on MATLAB 7.0 (R14).
2. Added a Physical Model Component Library for SIMULINK (Only for MATLAB 6.5.1 or greater).
3. Fixed a bug which did not allow setting fault times $t = 0$ in dynamic simulations.
4. Added the possibilities of exporting time domain simulations as ASCII files.
5. Fixed some bugs in the filter for PSS/E 29 data format.

6. Modified the TCSC control system (the first block is now a wash-out filter).
7. Fixed a bug in time domain simulation which produced an error when handling snapshots.
8. Corrected several minor bugs in the functions and typos in the documentation (the latter thanks to Marcos Miranda).
9. Successful testing on MATLAB 7.0 and GNU OCTAVE 2.1.57 & octave-forge 2004-07-07 for Mac OS X 10.3.5 (by Randall Smith).

4.2.4 News in version 1.3.1

1. Added a numeric linear analysis library (contribution by Alberto Del Rosso).
2. Added a new wind turbine model with direct drive synchronous generator (*development*).
3. Improved models of synchronous generators (which now include a simple q -axis saturation), AVR s and PSS s .
4. Added a filter for PSS/E 29 data format.
5. Added base conversion for flow limits of transmission lines.
6. Corrected a bug in the `fm_pq` function (computation of Jacobian matrices when voltage limit control is enabled).
7. Improved continuation power flow routine.
8. Corrected several minor bugs in the functions and typos in the documentation.
9. Fixed a few GNU OCTAVE compatibility issues.

4.2.5 News in version 1.3.0

1. Added the command line version.
2. Basic compatibility with GNU OCTAVE (only for command line version).
3. Added wind models, i.e. Weibull's distribution and composite wind model. Wind measurement data are supported as well.
4. Added wind turbine models (constant speed wind turbine and doubly fed induction generator).
5. Bus frequency measurement block.
6. Improved continuation and optimal power flow routines. The continuation power flow routine allows now using dynamic components (experimental).

7. Improved model of LTC transformers. Discrete tap ratio is now better supported and includes a time delay.
8. Improved PSAT/GAMS interface.
9. Improved the routine for small signal stability analysis. Results and settings are now contained in the structure `SSSA`. Output can be exported to Excel, `.xls` for reports in Excel; `LATEX` or plain text formats.
10. PMU placement reports can be exported to Excel, `LATEX` or plain text formats.
11. Corrected a few bugs in the PSS function.

4.2.6 News in version 1.2.2

1. Added the `autorun.m` function which allows launching any routine without solving the power flow analysis first.
2. Power flow reports can be exported to Excel, `TeX` or plain text formats.
3. Added filters to convert data files into BPA and Tshing Hua University formats.
4. Improved model of solid oxide fuel cell. Reactive power output is now included in the converted model.
5. Overall improvement of the toolbox and its documentation. The stablest release so far.

4.2.7 News in version 1.2.1

Minor bug-fix release. Main improvements are in functions `psat.m`, `fm_base.m` and `fm_sim.m`.

4.2.8 News in version 1.2.0

1. First PSAT release that is MATLAB version independent.
2. Installation of PSAT folder is now not required, although recommended.
3. Several bug fixes in continuation and optimal power flow routines.
4. Improved fault computation for time domain simulations. These improvements remove simulation errors which occurred in previous PSAT versions.
5. Added a new filters in Perl language for data format conversion.
6. Several bugs and typos were removed thanks to Liulin.

4.2.9 News in version 1.1.0

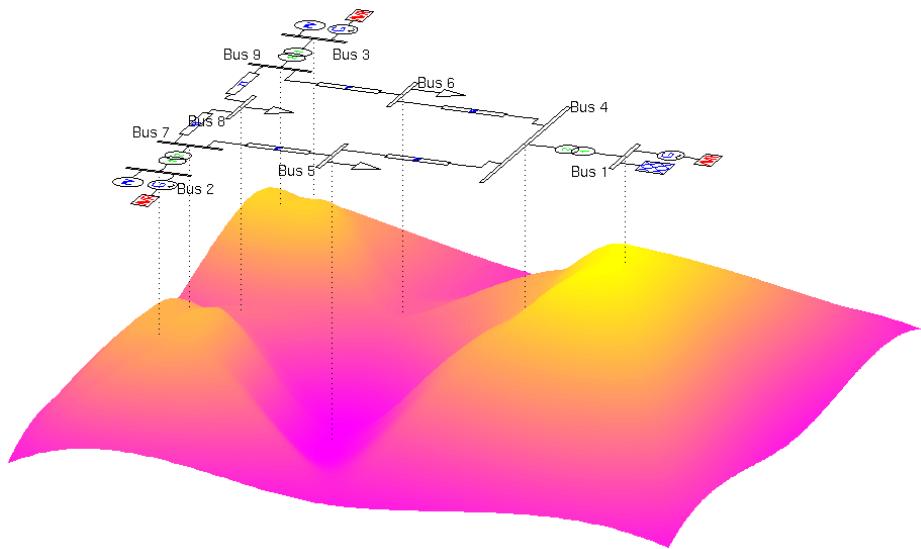
1. Created the PSAT Forum (<http://groups.yahoo.com/group/psatforum>).
2. Added PSAT/GAMS interface.
3. Added PSAT/UWPFLOW interface.
4. Added phase shifting transformer model.
5. Added filter for CYMFLOW data format.
6. Corrected some bugs in the filter for MatPower data format.

4.2.10 News in version 1.0.1

Minor bug-fix release.

Part II

Routines



Chapter 5

Power Flow

This chapter describes routines, settings and graphical user interfaces for power flow computations. The standard Newton-Raphson's method [127] and the Fast Decoupled Power Flow (XB and BX variations [121, 120, 129]) are implemented. A power flow with a distributed slack bus model is also included.

5.1 Power Flow Solvers

The power flow problem is formulated as the solution of a nonlinear set of equations in the form:

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{0} = \mathbf{f}(\mathbf{x}, \mathbf{y}) \\ \mathbf{0} &= \mathbf{g}(\mathbf{x}, \mathbf{y})\end{aligned}\tag{5.1}$$

where $\mathbf{y} \in \mathbb{R}^m$ are the algebraic variables, i.e. voltage amplitudes \mathbf{v} and phases $\boldsymbol{\theta}$ at the network buses and all other algebraic variables such as generator field voltages, AVR reference voltages, etc., $\mathbf{x} \in \mathbb{R}^n$ are the state variables, $\mathbf{g} \in \mathbb{R}^m$ are the algebraic equations and $\mathbf{f} \in \mathbb{R}^n$ are the differential equations. Observe that algebraic variables and equations are at least twice the number of buses defined in the network. Differential equations are included in (5.1) since PSAT initializes the state variables of some dynamic components (e.g. induction motors and load tap changers) during power flow computations. Other state variables and control parameters are initialized after solving the power flow solution (e.g. synchronous machines and regulators). Refer to Section 5.1.5 for the complete list of components that are included in or initialized after the power flow solution.

5.1.1 Newton-Raphson's Method

The Newton-Raphson's method for solving the power flow problem is described in many books and papers (e.g. [127]). At each iteration, the Jacobian matrix of (5.1)

is updated and the following linear problem is solved:

$$\begin{aligned}\begin{bmatrix} \Delta \mathbf{x}^i \\ \Delta \mathbf{y}^i \end{bmatrix} &= - \begin{bmatrix} \mathbf{f}_x^i & -\mathbf{f}_y^i \\ \mathbf{g}_x^i & \mathbf{g}_y^i \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{f}^i \\ \mathbf{g}^i \end{bmatrix} \\ \begin{bmatrix} \mathbf{x}^{i+1} \\ \mathbf{y}^{i+1} \end{bmatrix} &= \begin{bmatrix} \mathbf{x}^i \\ \mathbf{y}^i \end{bmatrix} + \begin{bmatrix} \Delta \mathbf{x}^i \\ \Delta \mathbf{y}^i \end{bmatrix}\end{aligned}\quad (5.2)$$

where $\mathbf{f}_x = \nabla_{\mathbf{x}}^T \mathbf{f}$, $\mathbf{f}_y = \nabla_{\mathbf{y}}^T \mathbf{f}$, $\mathbf{g}_x = \nabla_{\mathbf{x}}^T \mathbf{g}$, and $\mathbf{g}_y = \nabla_{\mathbf{y}}^T \mathbf{g}$. If the variable increments $\Delta \mathbf{x}$ and $\Delta \mathbf{y}$ are lower than a given tolerance ϵ or the number of iteration is greater than a given limit ($i > i_{\max}$) the routine stops. Observe that the standard power flow Jacobian matrix sometimes referred to as \mathbf{J}_{LF} is a sub-matrix of \mathbf{g}_y (see also next section). Furthermore, the following conditions applies:

- The column of the derivatives with respect to the reference angle is set to zero;
- The columns of the derivatives with respect to generator voltages are set to zero;
- The row of the derivatives of the slack bus active power balance is set to zero;
- The rows of the derivatives of generator reactive power balances are set to zero;
- Diagonal elements at the intersections of the columns and the rows described above are set to one;
- The elements of the vector \mathbf{g} associated with the generator reactive powers and the slack bus active power are set to zero.

These assumptions are equivalent to the equations:

$$\begin{aligned}\theta_{\text{slack}} &= \theta_{\text{slack}}^0 \\ \mathbf{v}_G &= \mathbf{v}_G^0\end{aligned}\quad (5.3)$$

where θ_{slack} is the voltage phase of the reference bus and \mathbf{v}_G the vector of generator voltages. Although forcing the dimensions of \mathbf{g}_y to be always maximum (i.e. m), this formulation is not computationally expensive, since the properties of MATLAB sparse matrices are used.

5.1.2 Fast Decoupled Power Flow

The Fast Decoupled Power Flow (FDPF) was originally proposed in [121] and has been further developed and generalized in several variations. PSAT uses the XB and BX methods presented in [129].

This method can be used if algebraic variables are only voltage magnitudes and phases. In this case, $\mathbf{J}_{\text{LF}} = \mathbf{g}_y$. The power flow Jacobian matrix \mathbf{J}_{LF} can be decomposed in four sub-matrices:

$$\mathbf{J}_{\text{LF}} = \begin{bmatrix} \mathbf{g}_{p,\theta} & \mathbf{g}_{p,v} \\ \mathbf{g}_{q,\theta} & \mathbf{g}_{q,v} \end{bmatrix}\quad (5.4)$$

where we have defined \mathbf{g}_p and \mathbf{g}_q as the subsets of \mathbf{g} corresponding to the active and reactive power balances, respectively. Thus, one has: $\mathbf{g}_{p,\theta} = \nabla_{\theta}^T \mathbf{g}_p$, $\mathbf{g}_{p,v} = \nabla_v^T \mathbf{g}_p$, $\mathbf{g}_{q,\theta} = \nabla_{\theta}^T \mathbf{g}_q$, and $\mathbf{g}_{q,v} = \nabla_v^T \mathbf{g}_q$.

The basic assumptions of FDPF methods are:

$$\begin{aligned}\mathbf{g}_{p,v} &= \mathbf{0} \\ \mathbf{g}_{q,\theta} &= \mathbf{0} \\ \mathbf{g}_{p,\theta} &\approx \mathbf{B}' \\ \mathbf{g}_{q,v} &\approx \mathbf{B}''\end{aligned}\tag{5.5}$$

where \mathbf{B}' and \mathbf{B}'' are simplified admittance matrices, as follows:

1. Line charging, shunts and transformer tap ratios are neglected when computing \mathbf{B}' ;
2. Phase shifters are neglected and line charging and shunts are doubled when computing \mathbf{B}'' .

The XB and BX variations differ only in further simplifications of the \mathbf{B}' and \mathbf{B}'' matrices respectively, as follows:

XB: line resistances are neglected when computing \mathbf{B}' ;

BX: line resistances are neglected when computing \mathbf{B}'' .

Thus the FDPF consists in turn of solving two linear systems at each iteration, as follows:

$$\begin{aligned}\boldsymbol{\theta}^{i+1} &= \boldsymbol{\theta}^i - [\mathbf{B}']^{-1} \mathbf{g}_p(\mathbf{v}^i, \boldsymbol{\theta}^i) / \mathbf{v}^i \\ \mathbf{v}^{i+1} &= \mathbf{v}^i - [\mathbf{B}'']^{-1} \mathbf{g}_q(\mathbf{v}^i, \boldsymbol{\theta}^{i+1}) / \mathbf{v}^i\end{aligned}\tag{5.6}$$

To reduce the iteration number, the angles $\boldsymbol{\theta}^{i+1}$ are used to compute the reactive power mismatches \mathbf{g}_q . Since \mathbf{B}' and \mathbf{B}'' are constant, these matrices can be factorized only once. If reactive power limits of PV buses are enforced, \mathbf{B}'' has to be re-factorized each time a PV bus is switched to a constant PQ bus.

PSAT allows using FDPF methods for systems that contain only PV generators, PQ loads and one slack bus. The standard Newton-Raphson's method is used, if other components are present in the network.

5.1.3 Distributed Slack Bus Model

The distributed slack bus model is based on a generalized power center concept and consists in distributing losses among all generators [12]. This is practically obtained by including in (5.1) a scalar variable k_G (which substitutes the active power of the slack bus) and by rewriting all generator active powers (slack bus included) as follows:

$$p_{Gi} = (1 + k_G \gamma_i) p_{Gi}^0, \quad \forall i = 1, \dots, n_G\tag{5.7}$$

where p_{Gi}^0 is the assigned power dispatch and γ_i allows weighting the participation of each generator to network losses. In a single slack bus formulation, $\gamma_i = 0$ for all generators except for the slack bus that has $\gamma_{\text{slack}} = 0$ (see also Subsections 12.4 and 12.5). Finally, (5.2) are modified by adding to the Jacobian matrix \mathbf{J}_{LF} the row of the derivatives of the slack bus active power balance and a column for the derivatives of differential and algebraic equations with respect to k_G . If the distributed slack bus model is enforced, FDPF methods are automatically disabled.

5.1.4 Other Power Flow Solvers

PSAT also implements non-conventional power flow solvers such as Iwamoto's method [62] and some solvers based on the continuous Newton's method [82]. The reader is invited to read references [62] and [82] for mathematical details.

These solvers are useful in case of ill-conditioned systems but do not generally work for unsolvable cases. Unsolvable cases are better tackled using the continuation power flow technique.

5.1.5 Initialization of State Variables

Dynamic components and non-conventional loads can be included in or initialized after the power flow solution. If a device is initialized after the power flow analysis, bus voltages ($\hat{\mathbf{v}}$ and $\hat{\boldsymbol{\theta}}$) and power injections ($\hat{\mathbf{p}}$ and $\hat{\mathbf{q}}$) are used as input parameters for computing the initial state variable vector $\hat{\mathbf{x}}$, internal algebraic variables $\hat{\mathbf{y}}$, and/or reference signals $\hat{\mathbf{u}}$, as depicted in Fig. 5.1. In Fig. 5.1, the symbol $(\hat{\cdot})$

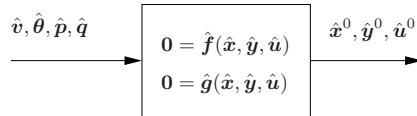


Figure 5.1: Initialization of device state and internal algebraic variables.

indicates the subset of equations or variables that are specific of the device that has to be initialized. Let us consider an example to clarify this concept. The set of equations for initializing the classical model of the synchronous machine are (see

Section 17.1):

$$\begin{aligned}
 0 &= \omega_n(\omega - 1) \\
 0 &= (p_m - p_e - D(\omega - 1))/M \\
 0 &= (v_q + r_a i_q) i_q + (v_d + r_a i_d) i_d - p_e \\
 0 &= v_d i_d + v_q i_q - p \\
 0 &= v_q i_d - v_d i_q - q \\
 0 &= v_q + r_a i_q - e'_q + x'_d i_d \\
 0 &= v_d + r_a i_d - x'_d i_q \\
 0 &= v \sin(\delta - \theta) - v_d \\
 0 &= v \cos(\delta - \theta) - v_q \\
 0 &= p_{m0} - p_m \\
 0 &= e'_{q0} - e'_q
 \end{aligned}$$

where v , θ , p and q are known from the power flow solution and:

$$\begin{aligned}
 \hat{\mathbf{x}} &= [\delta, \omega] \\
 \hat{\mathbf{y}} &= [v_d, v_q, i_d, i_q, e'_q, p_e, p_m] \\
 \hat{\mathbf{u}} &= [p_{m0}, e'_{q0}]
 \end{aligned}$$

The following components are included in the power flow equation set:

Hvdc	Lines	Ltc	Mn	Mot	PQ	PV
Phs	P1	SW	Tap			

whereas the following ones are initialized after solving the power flow problem:

Busfreq	Cac	Cluster	Cswt	Ddsg	Dfig	Exc
F1	Exload	Jimma	Mass	Mixload	Mn	Oxl
P1	Pmu	Pod	Pss	SSR	Sofc	Statcom
Sssc	Svc	Syn	Tcsc	Tg	Thload	Upfc

Wind

Voltage dependent and ZIP loads (Mn and P1) appears in both lists since their inclusion in the power flow computation is an available option. Refer to Subsections 16.1 and 16.2 for details.

5.2 Settings

General settings for power flow computations, i.e. power and frequency rates of the system, convergence tolerance and maximum number of iterations used for

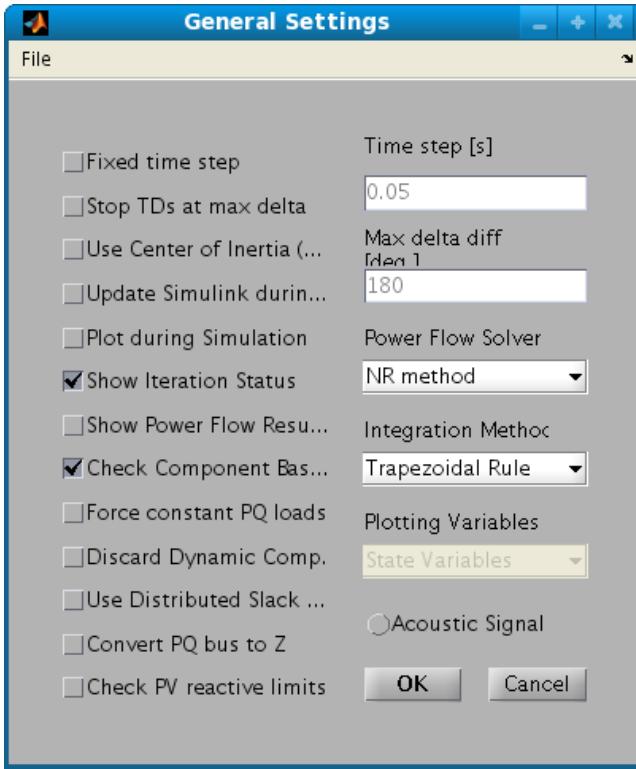


Figure 5.2: GUI for general settings.

the power flow solver can be set in the main window. Other parameters can be customized in the GUI for general settings (menu *Edit/General Settings* or shortcut <Ctrl-k> in the main window), which is depicted in Fig. 5.2. The following options are available for power flow analysis:

Power Flow Solver (Settings.pfsolver): these are the Newton-Raphson's (NR) method, the fast decoupled XB and BX methods, and three robust power flow methods.

Use Distributed Slack Bus (Settings.distrlsw): this option allows using distributed slack bus model, i.e. all PV buses contributes to system losses, not only the reference angle bus. This option is disabled if there are dynamic component included in the power flow analysis.

Check Component Bases (Settings.conv): enable checking component power and voltage ratings. The check of the consistency of component ratings is made by the function `fm_base.m`. Only a reduced number of component are checked. Refer to the code for details.

Discard Dynamic Components (Settings.static): if this option is enabled, dynamic components initialized after the power flow are discarded. Observe that dynamic components that are included in the power flow analysis are retained.

Check PV Reactive Limits (Settings.pv2pq): this option forces checking reactive power limits of PV buses. If a limit is reached, the PV bus is converted into a PQ bus. No voltage recovery is taken into account. More precise results can be obtained using the continuation power flow analysis.

Force constant PQ loads (Settings.forcepq): this option forces using constant powers for PQ loads, regardless the bus voltage level. This option allows neglecting the information given in the PQ load data.

Show Iteration Status (Settings.show): display the absolute minimum convergence error in the main window during power flow analysis.

Show Power Flow Results (Settings.showlf): open the static report GUI and display power flow results when power flow analysis has completed.

Power flow settings are stored in the structure `Settings`, which contains also general settings and parameters for time domain simulations. This structure is described in Appendix A.

5.3 Example

Figure 5.3 depicts the GUI for power flow results. Data refer to a 9-bus test system presented and discussed in [111]. The GUI reports the bus names and their correspondent voltages and total power injections. Voltage profiles can be plotted using the buttons on top of the lists for voltage magnitudes and angles. Angles can be expressed either in radians or degrees. If the loaded system presents state and control variables, these are reported in the GUI as well. Power flow results can be saved using the *Report* button. A log file will be created using the selected format (plain text, L^AT_EX, Excel) and displayed with the selected viewer (see Section 27.6 for details). For example, the plain text power flow solution for the WSCC 9-bus test system is as follows:

```
POWER FLOW REPORT

P S A T  2.1.0

Author: Federico Milano, (c) 2002-2008
e-mail: Federico.Milano@uclm.es
website: http://www.uclm.es/area/gsee/Web/Federico

File: /Users/fmilano/Applications/psat2/tests/d_009.mdl
Date: 27-May-2008 12:20:23
```

NETWORK STATISTICS

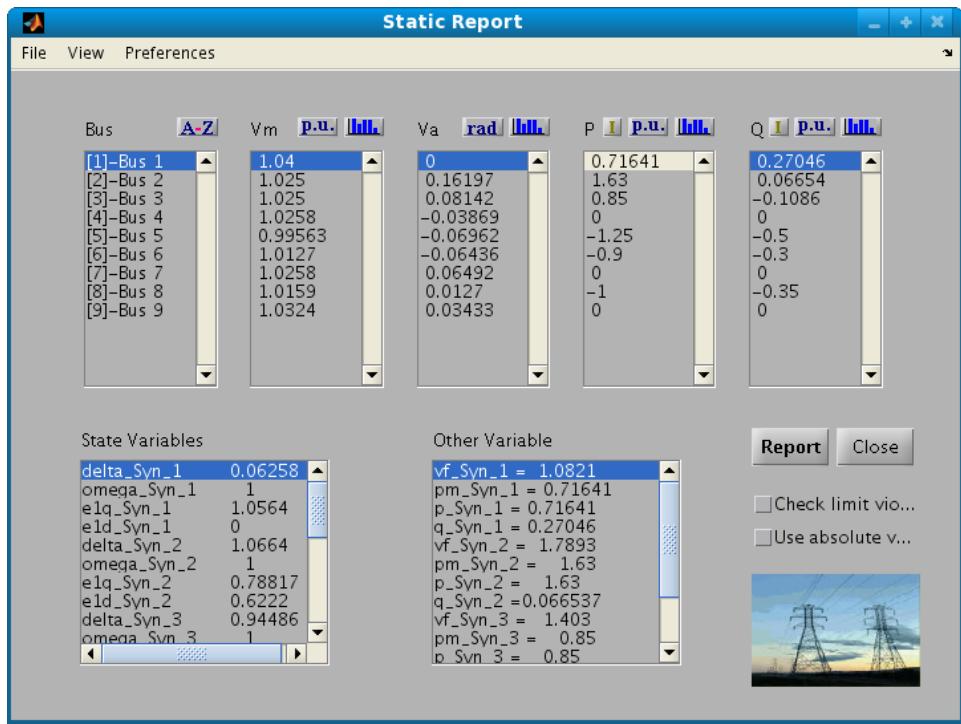


Figure 5.3: GUI for displaying power flow results.

Buses:	9
Lines:	6
Transformers:	3
Generators:	3
Loads:	3

SOLUTION STATISTICS

Number of Iterations:	4
Maximum P mismatch [p.u.]	0
Maximum Q mismatch [p.u.]	0
Power rate [MVA]	100

POWER FLOW RESULTS

Bus	V [p.u.]	phase [rad]	P gen [p.u.]	Q gen [p.u.]	P load [p.u.]	Q load [p.u.]
Bus 1	1.04	0	0.71641	0.27046	0	0
Bus 2	1.025	0.16197	1.63	0.06654	0	0
Bus 3	1.025	0.08142	0.85	-0.1086	0	0
Bus 4	1.0258	-0.03869	0	0	0	0
Bus 5	0.99563	-0.06962	0	0	1.25	0.5
Bus 6	1.0127	-0.06436	0	0	0.9	0.3
Bus 7	1.0258	0.06492	0	0	0	0
Bus 8	1.0159	0.0127	0	0	1	0.35
Bus 9	1.0324	0.03433	0	0	0	0

STATE VARIABLES

delta_Syn_1	0.06258
omega_Syn_1	1
e1q_Syn_1	1.0564
e1d_Syn_1	0
delta_Syn_2	1.0664
omega_Syn_2	1
e1q_Syn_2	0.78817
e1d_Syn_2	0.6222
delta_Syn_3	0.94486
omega_Syn_3	1
e1q_Syn_3	0.76786
e1d_Syn_3	0.62424
vm_Exc_1	1.04
vr1_Exc_1	1.1006
vr2_Exc_1	-0.19479
vf_Exc_1	1.0822
vm_Exc_2	1.025
vr1_Exc_2	1.8951
vr2_Exc_2	-0.32208
vf_Exc_2	1.7893
vm_Exc_3	1.025
vr1_Exc_3	1.446
vr2_Exc_3	-0.25254
vf_Exc_3	1.403

OTHER ALGEBRAIC VARIABLES

vf_Syn_1	1.0324
pm_Syn_1	1.0822
p_Syn_1	0.71641
q_Syn_1	0.71641
vf_Syn_2	0.27046
pm_Syn_2	1.7893
p_Syn_2	1.63
q_Syn_2	1.63
vf_Syn_3	0.06654
pm_Syn_3	1.403
p_Syn_3	0.85
q_Syn_3	0.85
vref_Exc_1	-0.1086
vref_Exc_2	1.095
vref_Exc_3	1.1198

LINE FLOWS

From Bus	To Bus	Line	P Flow [p.u.]	Q Flow [p.u.]	P Loss [p.u.]	Q Loss [p.u.]
Bus 9	Bus 8	1	0.24183	0.0312	0.00088	-0.21176
Bus 7	Bus 8	2	0.7638	-0.00797	0.00475	-0.11502
Bus 9	Bus 6	3	0.60817	-0.18075	0.01354	-0.31531
Bus 7	Bus 5	4	0.8662	-0.08381	0.023	-0.19694
Bus 5	Bus 4	5	-0.4068	-0.38687	0.00258	-0.15794
Bus 6	Bus 4	6	-0.30537	-0.16543	0.00166	-0.15513
Bus 2	Bus 7	7	1.63	0.06654	0	0.15832
Bus 3	Bus 9	8	0.85	-0.1086	0	0.04096
Bus 1	Bus 4	9	0.71641	0.27046	0	0.03123

LINE FLOWS

From Bus	To Bus	Line	P Flow [p.u.]	Q Flow [p.u.]	P Loss [p.u.]	Q Loss [p.u.]
Bus 8	Bus 9	1	-0.24095	-0.24296	0.00088	-0.21176
Bus 8	Bus 7	2	-0.75905	-0.10704	0.00475	-0.11502
Bus 6	Bus 9	3	-0.59463	-0.13457	0.01354	-0.31531
Bus 5	Bus 7	4	-0.8432	-0.11313	0.023	-0.19694
Bus 4	Bus 5	5	0.40937	0.22893	0.00258	-0.15794
Bus 4	Bus 6	6	0.30704	0.0103	0.00166	-0.15513
Bus 7	Bus 2	7	-1.63	0.09178	0	0.15832
Bus 9	Bus 3	8	-0.85	0.14955	0	0.04096
Bus 4	Bus 1	9	-0.71641	-0.23923	0	0.03123

GLOBAL SUMMARY REPORT

TOTAL GENERATION

REAL POWER [p.u.] 3.1964
 REACTIVE POWER [p.u.] 0.2284

TOTAL LOAD

REAL POWER [p.u.]	3.15
REACTIVE POWER [p.u.]	1.15

TOTAL LOSSES

REAL POWER [p.u.]	0.04641
REACTIVE POWER [p.u.]	-0.9216

The report above lists buses, line flows and state variables in separate blocks. This is the default format. One may want to include the information about line flows and state variables in the bus report, instead. This can be done by checking on the menu option *Preferences / Embed line flows in bus report*. The resulting report for the 9-bus system is as follows.

POWER FLOW REPORT

P S A T 2.1.0

Author: Federico Milano, (c) 2002-2008
 e-mail: Federico.Milano@uclm.es
 website: <http://www.uclm.es/area/gsee/Web/Federico>

File: /Users/fmilano/Applications/psat2/tests/d_009.mdl
 Date: 27-May-2008 12:21:01

NETWORK STATISTICS

Buses:	9
Lines:	6
Transformers:	3
Generators:	3
Loads:	3

SOLUTION STATISTICS

Number of Iterations:	4
Maximum P mismatch [p.u.]	0
Maximum Q mismatch [p.u.]	0
Power rate [MVA]	100

POWER FLOW RESULTS

Bus 1	V [p.u.]	phase [rad]	P gen [p.u.]	Q gen [p.u.]	P load [p.u.]	Q load [p.u.]
	1.04	0	0.71641	0.27046	0	0
To Bus	Line		P Flow [p.u.]	Q Flow [p.u.]	P Loss [p.u.]	Q Loss [p.u.]
Bus 4	9		0.71641	0.27046	0	0.03123

STATE VARIABLES

delta_Syn_1	0.06258
omega_Syn_1	1
e1q_Syn_1	1.0564
e1d_Syn_1	0
vm_Exc_1	1.04
vr1_Exc_1	1.1006
vr2_Exc_1	-0.19479
vf_Exc_1	1.0822

OTHER ALGEBRAIC VARIABLES

vf_Syn_1	1.0822
pm_Syn_1	0.71641
p_Syn_1	0.71641
q_Syn_1	0.27046
vref_Exc_1	1.095

Bus 2	V [p.u.]	phase [rad]	P gen [p.u.]	Q gen [p.u.]	P load [p.u.]	Q load [p.u.]
	1.025	0.16197	1.63	0.06654	0	0
To Bus	Line		P Flow [p.u.]	Q Flow [p.u.]	P Loss [p.u.]	Q Loss [p.u.]
Bus 7	7		1.63	0.06654	0	0.15832

STATE VARIABLES

delta_Syn_2	1.0664
omega_Syn_2	1
e1q_Syn_2	0.78817
e1d_Syn_2	0.6222
vm_Exc_2	1.025
vr1_Exc_2	1.8951
vr2_Exc_2	-0.32208
vf_Exc_2	1.7893

OTHER ALGEBRAIC VARIABLES

vf_Syn_2	1.7893
pm_Syn_2	1.63
p_Syn_2	1.63
q_Syn_2	0.06654
vref_Exc_2	1.1198

Bus 3	V [p.u.]	phase [rad]	P gen [p.u.]	Q gen [p.u.]	P load [p.u.]	Q load [p.u.]
	1.025	0.08142	0.85	-0.1086	0	0
To Bus	Line		P Flow [p.u.]	Q Flow [p.u.]	P Loss [p.u.]	Q Loss [p.u.]
Bus 9	8		0.85	-0.1086	0	0.04096

STATE VARIABLES

delta_Syn_3	0.94486
omega_Syn_3	1
e1q_Syn_3	0.76786
e1d_Syn_3	0.62424
vm_Exc_3	1.025
vr1_Exc_3	1.446
vr2_Exc_3	-0.25254
vf_Exc_3	1.403

OTHER ALGEBRAIC VARIABLES

vf_Syn_3	1.403
pm_Syn_3	0.85
p_Syn_3	0.85
q_Syn_3	-0.1086
vref_Exc_3	1.0973

Bus 4	V [p.u.]	phase [rad]	P gen [p.u.]	Q gen [p.u.]	P load [p.u.]	Q load [p.u.]
	1.0258	-0.03869	0	0	0	0
To Bus	Line		P Flow [p.u.]	Q Flow [p.u.]	P Loss [p.u.]	Q Loss [p.u.]
Bus 5	5		0.40937	0.22893	0.00258	-0.15794
Bus 6	6		0.30704	0.0103	0.00166	-0.15513
Bus 1	9		-0.71641	-0.23923	0	0.03123
Bus 5	V [p.u.]	phase [rad]	P gen [p.u.]	Q gen [p.u.]	P load [p.u.]	Q load [p.u.]
	0.99563	-0.06962	0	0	1.25	0.5
To Bus	Line		P Flow [p.u.]	Q Flow [p.u.]	P Loss [p.u.]	Q Loss [p.u.]
Bus 4	5		-0.4068	-0.38687	0.00258	-0.15794
Bus 7	4		-0.8432	-0.11313	0.023	-0.19694
Bus 6	V [p.u.]	phase [rad]	P gen [p.u.]	Q gen [p.u.]	P load [p.u.]	Q load [p.u.]
	1.0127	-0.06436	0	0	0.9	0.3
To Bus	Line		P Flow [p.u.]	Q Flow [p.u.]	P Loss [p.u.]	Q Loss [p.u.]
Bus 4	6		-0.30537	-0.16543	0.00166	-0.15513
Bus 9	3		-0.59463	-0.13457	0.01354	-0.31531
Bus 7	V [p.u.]	phase [rad]	P gen [p.u.]	Q gen [p.u.]	P load [p.u.]	Q load [p.u.]
	1.0258	0.06492	0	0	0	0

	To Bus	Line	P Flow [p.u.]	Q Flow [p.u.]	P Loss [p.u.]	Q Loss [p.u.]
	Bus 8	2	0.7638	-0.00797	0.00475	-0.11502
	Bus 5	4	0.8662	-0.08381	0.023	-0.19694
	Bus 2	7	-1.63	0.09178	0	0.15832
Bus 8	V [p.u.]	phase [rad]	P gen [p.u.]	Q gen [p.u.]	P load [p.u.]	Q load [p.u.]
	1.0159	0.0127	0	0	1	0.35
	To Bus	Line	P Flow [p.u.]	Q Flow [p.u.]	P Loss [p.u.]	Q Loss [p.u.]
	Bus 9	1	-0.24095	-0.24296	0.00088	-0.21176
	Bus 7	2	-0.75905	-0.10704	0.00475	-0.11502
Bus 9	V [p.u.]	phase [rad]	P gen [p.u.]	Q gen [p.u.]	P load [p.u.]	Q load [p.u.]
	1.0324	0.03433	0	0	0	0
	To Bus	Line	P Flow [p.u.]	Q Flow [p.u.]	P Loss [p.u.]	Q Loss [p.u.]
	Bus 8	1	0.24183	0.0312	0.00088	-0.21176
	Bus 6	3	0.60817	-0.18075	0.01354	-0.31531
	Bus 3	8	-0.85	0.14955	0	0.04096

GLOBAL SUMMARY REPORT

TOTAL GENERATION

REAL POWER [p.u.]	3.1964
REACTIVE POWER [p.u.]	0.2284

TOTAL LOAD

REAL POWER [p.u.]	3.15
REACTIVE POWER [p.u.]	1.15

TOTAL LOSSES

REAL POWER [p.u.]	0.04641
REACTIVE POWER [p.u.]	-0.9216

Results can also be displayed using a two or three-dimensional colored map (see Figs. 5.4 and 5.5). The GUI for network visualization is available from the menu *View / Network Visualization* of the main PSAT window. The variables that can be displayed are voltage magnitudes and angles, transmission line apparent flows, generator rotor angles and speeds, and LMPs and NCPs. The same GUI can be used to create movies for CPF analysis and time domain simulations. See Section 27.2 for more details on the GUI for 3D map visualization.

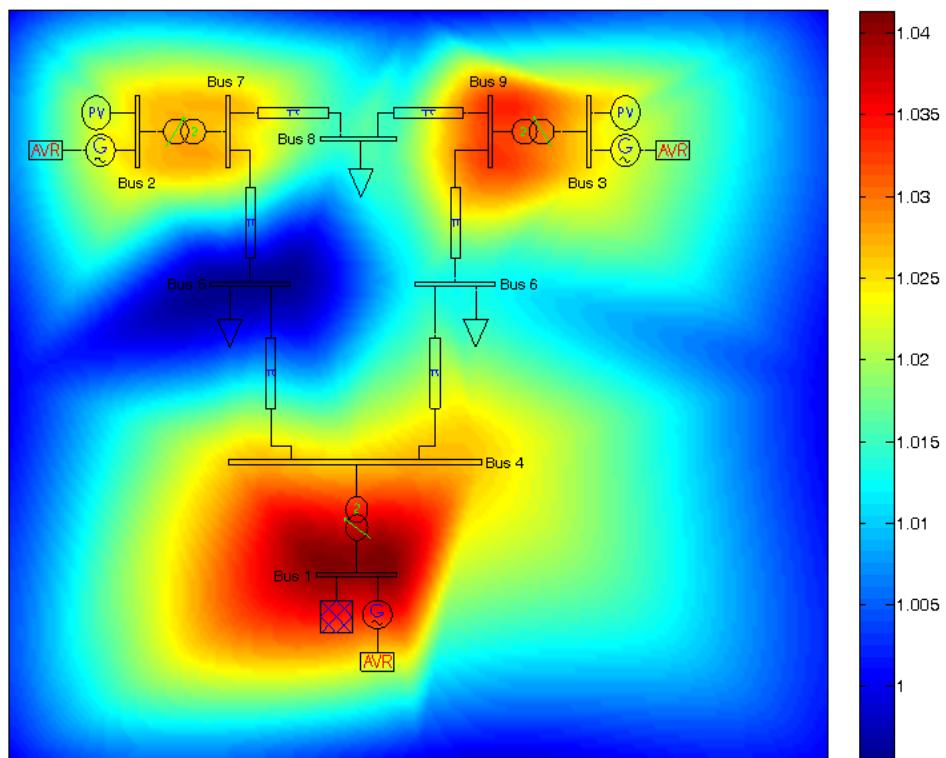


Figure 5.4: 2D visualization of power flow results. Voltage magnitudes for the 9-bus test system.

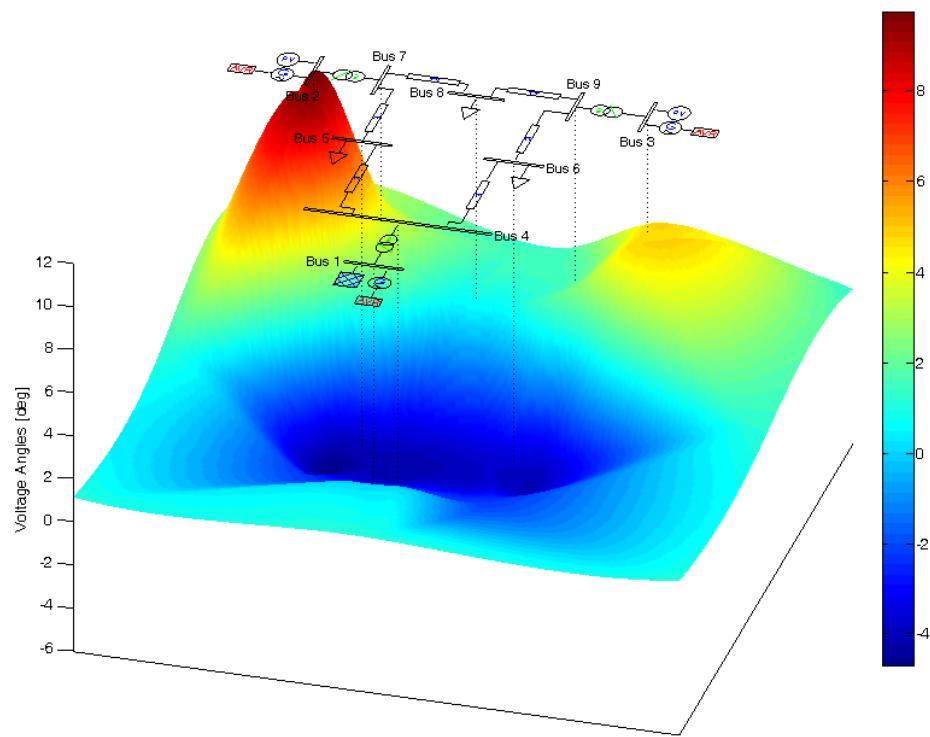


Figure 5.5: 3D visualization of power flow results. Voltage angles for the 9-bus test system.

Chapter 6

Bifurcation Analysis

This chapter describes Direct Methods (DM) for computing Saddle-Node Bifurcation (SNB) points and Limit-Induced Bifurcation (LIB) points, and a Continuation Power Flow (CPF) technique based on [20]. The CPF analysis is more general than DMs, and can be used also for determining generator reactive power limits, voltage limits and flow limits of transmission lines.

Bifurcation analysis requires steady-state equations of power system models, as follows:

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{0} = \mathbf{f}(\mathbf{x}, \mathbf{y}, \lambda) \\ \mathbf{0} &= \mathbf{g}(\mathbf{x}, \mathbf{y}, \lambda)\end{aligned}\tag{6.1}$$

where \mathbf{x} are the state variables, \mathbf{y} the algebraic variables (voltage amplitudes and phases) and λ is the *loading parameter*, i.e. a scalar variable which multiplies generator and load directions, as follows:

$$\begin{aligned}\mathbf{p}_G &= \mathbf{p}_G^0 + (\lambda + \gamma k_G) \mathbf{p}_S^0 \\ \mathbf{p}_L &= \mathbf{p}_L^0 + \lambda \mathbf{p}_D^0 \\ \mathbf{q}_L &= \mathbf{p}_L^0 + \lambda \mathbf{p}_D^0\end{aligned}\tag{6.2}$$

In (6.2), \mathbf{p}_G^0 , \mathbf{p}_L^0 and \mathbf{q}_L^0 are the “base case” generator and load powers, whereas \mathbf{p}_S^0 , \mathbf{p}_D^0 and \mathbf{q}_D^0 are the generator and load power directions. Power directions are defined in the structures **Supply** and **Demand** (see also Sections 13.1 and 13.4 for more details). If these data are not defined, the base case powers are used as load directions and (6.2) become:

$$\begin{aligned}\mathbf{p}_G &= (\lambda + \gamma k_G) \mathbf{p}_G^0 \\ \mathbf{p}_L &= \lambda \mathbf{p}_L^0 \\ \mathbf{q}_L &= \lambda \mathbf{q}_L^0\end{aligned}\tag{6.3}$$

Observe that power directions (6.2) and (6.3) used in the bifurcation analysis differ from (7.3), i.e. the power directions used in the voltage stability constrained OPF described in Chapter 7. The distributed slack bus variable k_G and the generator participation coefficients γ are optional.

6.1 Direct Methods

Direct Methods which are implemented in PSAT allow to compute the value of the loading parameter λ for at Saddle-Node Bifurcation points and at Limit-Induced Bifurcation points.

In PSAT, Direct Methods can perform only “static” bifurcation analysis, i.e. make use of static power flow models (see Chapter 12). Thus, (6.1) reduce to the algebraic set \mathbf{g} . Before running any direct method routine, the power flow analysis has to be run first to initialize the algebraic variables.

6.1.1 Saddle-Node Bifurcation

The conditions for a SNB point are as follows:

$$\begin{aligned}\mathbf{g}(\mathbf{y}, \lambda) &= \mathbf{0} \\ \mathbf{g}_{\mathbf{y}\boldsymbol{\nu}} &= \mathbf{0} \\ |\boldsymbol{\nu}| &= 1\end{aligned}\tag{6.4}$$

or

$$\begin{aligned}\mathbf{g}(\mathbf{y}, \lambda) &= \mathbf{0} \\ \mathbf{g}_{\mathbf{y}}^T \mathbf{w} &= \mathbf{0} \\ |\mathbf{w}| &= 1\end{aligned}\tag{6.5}$$

where $\boldsymbol{\nu}$ and \mathbf{w} are the right and the left eigenvectors respectively, and the Euclidean norm is used for the $|\cdot|$ operator. The Euclidean norm reduces the sparsity of the Jacobian matrix, but allows to avoid re-factorizations (as happens in the case of ∞ -norm) and is numerically more stable than the 1-norm. The solution for (6.4) and (6.5) are obtained by means of a Newton-Raphson’s method, and the complete Jacobian matrix is computed explicitly:

$$\begin{bmatrix} \mathbf{g}_{\mathbf{y}} & 0 & \mathbf{g}_{\lambda} \\ \mathbf{g}_{\mathbf{y}\mathbf{y}\boldsymbol{\nu}} & \mathbf{g}_{\mathbf{y}} & 0 \\ 0 & |\boldsymbol{\nu}|_{\boldsymbol{\nu}} & 0 \end{bmatrix}\tag{6.6}$$

Since the Hessian matrix $\mathbf{g}_{\mathbf{y}\mathbf{y}}$ is computed analytically, this method can be applied only to a limited number of components, namely (SW, PV, PQ and Line), which anyway are the standard models used in power flow analysis. The SNB routine searches a “good” initial guess for the eigenvectors $\boldsymbol{\nu}$ and \mathbf{w} . However the best way to initialize the SNB routine is to run first a CPF analysis.

Figure 6.1 depicts the GUI for SNB settings. A complete description of SNB settings is reported in Appendix A.

6.1.2 Limit Induced Bifurcation

Limit Induced Bifurcation points are defined as the solution of the following system:

$$\begin{aligned}\mathbf{0} &= \mathbf{g}(\mathbf{y}, \lambda) \\ 0 &= \rho(\mathbf{y})\end{aligned}\tag{6.7}$$

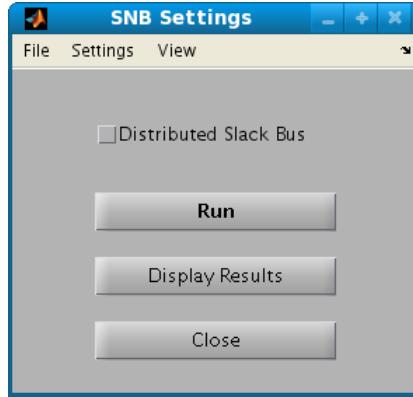


Figure 6.1: GUI for saddle-node bifurcation settings.

where $\rho(\mathbf{y})$ is an additional constraint that can be:

$$q_G = q_G^{\lim} \quad (6.8)$$

for slack or PV generator buses , or

$$v_L = v_L^{\lim} \quad (6.9)$$

for PQ load buses . Only reactive power limits of generator buses can lead to saddle limit induced bifurcation (SLIB) points, that are associated to a maximum loading condition.

Figure 6.2 depicts the GUI for LIB settings. A complete description of LIB structure is reported in Appendix A.

6.2 Continuation Power Flow

The Continuation Power Flow (CPF) method implemented in PSAT consists in a predictor step realized by the computation of the tangent vector and a corrector step that can be obtained either by means of a local parametrization or a perpendicular intersection. The CPF analysis is fully supported for static devices but generally works also with dynamic components.

6.2.1 Predictor Step

At a generic equilibrium point p , the following relation applies:

$$\psi(\mathbf{z}_p, \lambda_p) = \mathbf{0} \Rightarrow \left. \frac{d\psi}{d\lambda} \right|_p = \mathbf{0} = \psi_{\mathbf{z}}|_p \left. \frac{d\mathbf{z}}{d\lambda} \right|_p + \psi_{\lambda}|_p \quad (6.10)$$

where ψ is the vector of differential and algebraic equations and \mathbf{z} it the vector of state and algebraic variables that describe the DAE system.

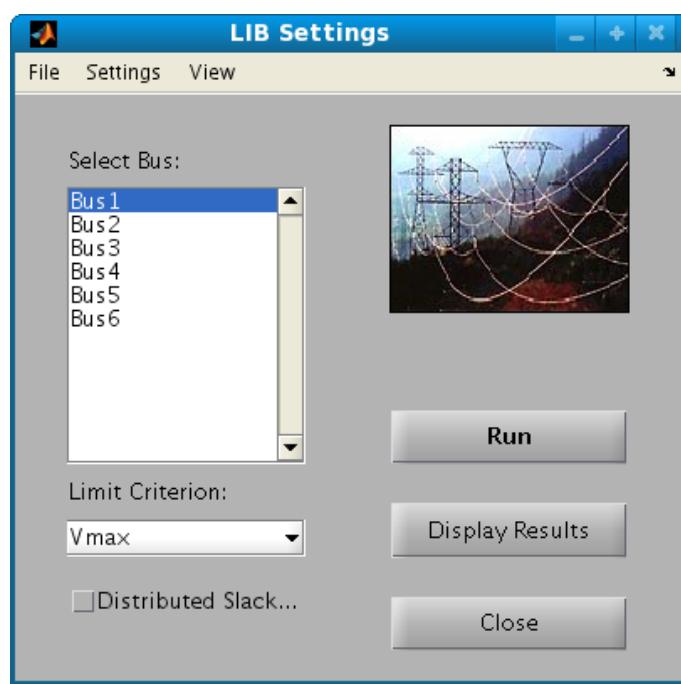


Figure 6.2: GUI for limit-induced bifurcation settings.

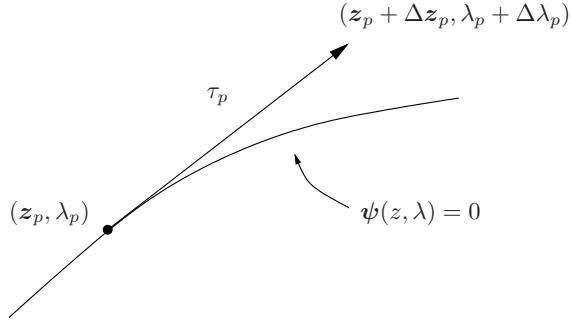


Figure 6.3: Continuation Power flow: predictor step obtained by means of tangent vector.

The tangent vector at the equilibrium point p can be approximated by:

$$\tau_p = \frac{d\mathbf{z}}{d\lambda} \Big|_p \approx \frac{\Delta z_p}{\Delta \lambda_p} \quad (6.11)$$

From (6.10) and (6.11), one has:

$$\begin{aligned} \tau_p &= -\psi_{\mathbf{z}}^{-1}|_p \psi_{\lambda}|_p \\ \Delta z_p &= \tau_p \Delta \lambda_p \end{aligned} \quad (6.12)$$

A step size control k has to be chosen for determining the increment Δz_p and $\Delta \lambda_p$, along with a normalization to avoid large step when $|\tau_p|$ is large:

$$\Delta \lambda_p \triangleq \frac{k}{|\tau_p|} \quad \Delta y_p \triangleq \frac{k \tau_p}{|\tau_p|} \quad (6.13)$$

where $k = \pm 1$, and its sign determines the increase or the decrease of λ . Figure 6.3 presents a pictorial representation of the predictor step.

6.2.2 Corrector Step

In the corrector step, a set of $n + 1$ equations is solved, as follows:

$$\begin{aligned} \psi(\mathbf{z}, \lambda) &= \mathbf{0} \\ \rho(\mathbf{z}, \lambda) &= 0 \end{aligned} \quad (6.14)$$

where the solution of ψ must be in the bifurcation manifold and ρ is an additional equation to guarantee a non-singular set at the bifurcation point. As for the choice of ρ , there are two options: the *perpendicular intersection* and the *local parametrization*.

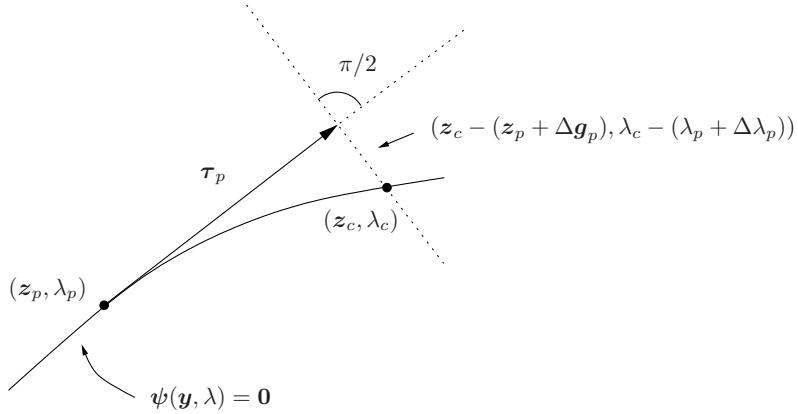


Figure 6.4: Continuation Power flow: corrector step obtained by means of perpendicular intersection.

In case of perpendicular intersection, whose pictorial representation is reported in Fig. 6.4, the expression of ρ becomes:

$$\rho(\mathbf{z}, \lambda) = \begin{bmatrix} \Delta \mathbf{z}_p \\ \Delta \lambda_p \end{bmatrix}^T \begin{bmatrix} \mathbf{z}_c - (\mathbf{z}_p + \Delta \mathbf{z}_p) \\ \lambda_c - (\lambda_p + \Delta \lambda_p) \end{bmatrix} = 0 \quad (6.15)$$

whereas for the local parametrization, either the parameter λ or a variable y_i is forced to be a fixed value:

$$\rho(\mathbf{z}, \lambda) = \lambda_c - \lambda_p - \Delta \lambda_p \quad (6.16)$$

or

$$\rho(\mathbf{z}, \lambda) = \mathbf{z}_{c_i} - \mathbf{z}_{p_i} - \Delta \mathbf{z}_{p_i} \quad (6.17)$$

The choice of the variable to be fixed depends on the bifurcation manifold of ψ , as depicted in Fig. 6.5.

6.2.3 $N - 1$ Contingency Analysis

PSAT is provided with a $N - 1$ contingency analysis that allows computing active power flow limits in transmission lines and transformers taking into account security limits (transmission line thermal limits, generator reactive power limits and voltage security limits) and voltage stability constraints.

The $N - 1$ contingency analysis consists in a continuation power flow analysis for each line outage. If the line outage leads to an unfeasible base case ($\lambda^{\max} < 1$), that line outage is neglected. Then all the contingencies are sorted in a “worst line contingency” order looking for the minimum power flows in each transmission line and transformers. These minimum power flows are the power flow limits and are thus the output of the $N - 1$ contingency analysis.

To launch the $N - 1$ contingency analysis, select the menu *Run/N-1 Contingency Analysis* in the main window or in the GUI for continuation power flow analysis.

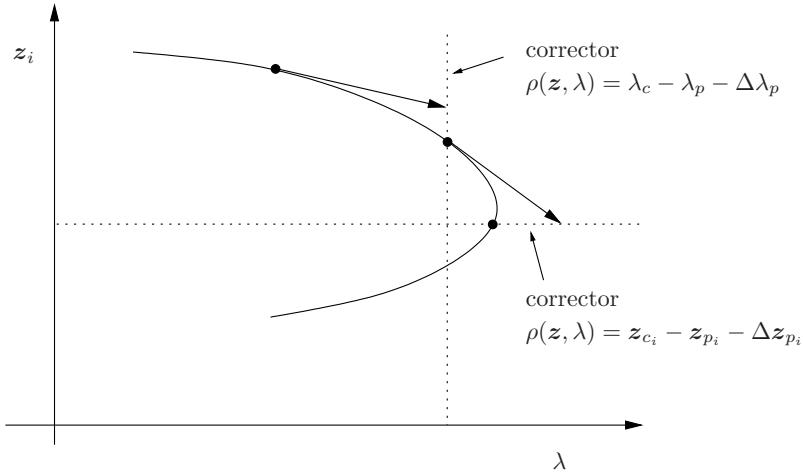


Figure 6.5: Continuation Power flow: corrector step obtained by means of local parametrization.

6.2.4 Graphical User Interface and Settings

Figure 6.6 depicts the user interface for continuation power flow analysis. Several options allow adjusting the performance and customizing routine outputs. It is possible to set the tolerance of the convergence test of the Newton-Raphson's method in the corrector step, the step size of the predictor step, and the total number of points determined by the routine. Furthermore, the routine can use a single slack bus or a distributed slack bus model and check for voltage limits, generator reactive power limits, and flow limits in the transmission lines and transformers. It is also possible to set the tolerances required to determine the voltage, reactive power and flow limits. As in the case of Optimal Power Flow routine, flow limits can be current amplitudes, active powers or apparent powers. For all of these flows, both ϕ_{ij} and ϕ_{ji} are checked. Three stopping criteria are available:

1. complete nose curve: the routine terminates when the maximum number of point is reached or when λ becomes negative;
2. if either a SNB or a LIB point is encountered: the LIB that causes the end of the routine corresponds also to the maximum loading parameter;
3. if either a bifurcation point or a limit is encountered.

In the menu *Options* of the CPF GUI, the following options can be selected:

1. Enforce the check for Hopf bifurcations. Checking for Hopf bifurcations is disabled by default.

2. Enforce the step size control during the CPF analysis. If the step size control is disabled, the CPF analysis will be faster but likely less accurate close to the maximum loading point. Step size control is enforced by default.
3. Include negative active power loads in CPF analysis. This option only takes effects if the user does not define the structure `Demand.con` for load directions. If this option is enabled, negative active power loads will be included in (6.3), while, by default, they would be excluded.

Negative loads are typically of two kinds: pure reactive compensators or constant PQ generators. In the latter case it may make sense to include them in the CPF analysis. The option will look for only negative active power loads. Pure reactive compensators will not be used in CPF analysis.

4. Include only negative active power loads or only PQ generators in CPF analysis. This option only takes effects if the user does not define the structure `Demand.con` for load directions. If this option is enabled, negative active power loads will be included in (6.3), while, by default, they would be excluded.

These options can be useful if one wants to study the effects of increasing the production of PQ generators on the network. See also the discussion above.

The trace of the CPF computations is stored in the Command History. All outputs can be plotted versus the loading parameter λ using the Plotting Utilities. Appendix A fully illustrates the CPF structure.

6.3 Examples

Figure 6.7 depicts CPF nose curves as displayed by means of the PSAT GUI for plotting results. The figure refers to three load voltages of the IEEE 14-bus test system (see Appendix F.4). Since no power directions are defined in the `Supply` and `Demand` data, base powers are used, as defined in the slack and PV generators and PQ load data.

Figures 6.8 and 6.9 depict CPF results for the 6-bus test system (see Appendix F.2) with distributed slack bus model. In this example, the power directions are defined in the `Supply` and `Demand` data. Figures 6.9 and 6.8 are obtained without and with generator reactive power limits and show a saddle-node bifurcation and a limit-induced bifurcation, respectively.

Figure 6.10 illustrates a bifurcation curve obtained for the 9-bus test system with dynamic models of synchronous generators and AVRs. By checking the radio button “Plot Snapshots” and selecting the style “Colors and symbols”, PSAT automatically adds a circle to relevant points (e.g. bifurcation points) and a text tag. This utility works only for a single curve. The check for Hopf bifurcations must be enforced in the CPF GUI (menu *Options*). Each point marked with a tag is internally stored as a snapshot (see also Section 9.4 for further details). After the CPF analysis, these points can be inspected as any snapshots through the Snapshot GUI and a report of state and algebraic variables at each bifurcation point can be produced.

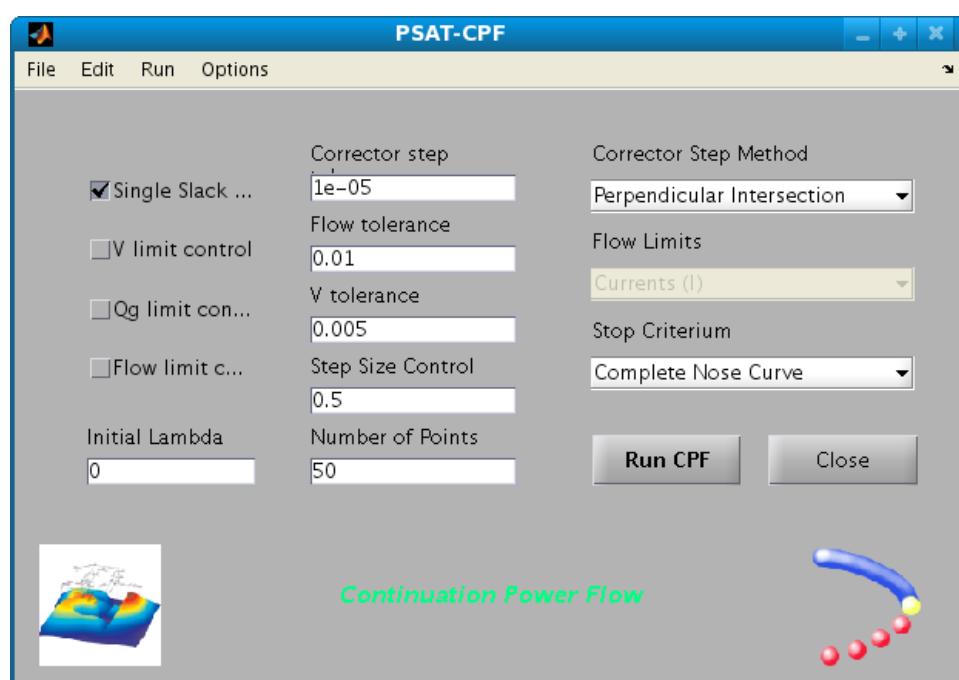


Figure 6.6: GUI for the continuation power flow settings.

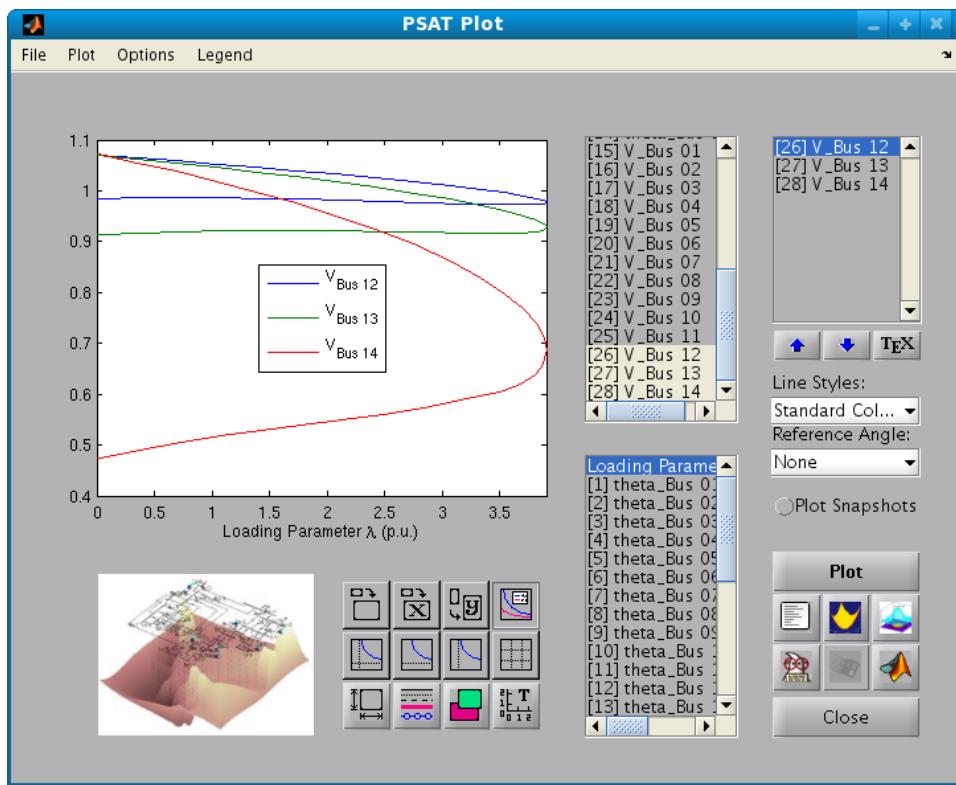


Figure 6.7: GUI for plotting CPF results. The nose curves refers to three load voltages of the IEEE 14-bus test system.

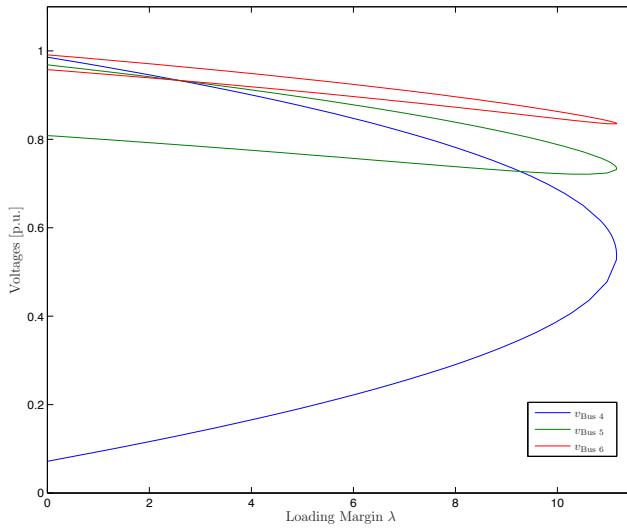


Figure 6.8: Nose curves for the 6-bus test system without generator reactive power limits. The maximum loading condition is given by a saddle-node bifurcation.

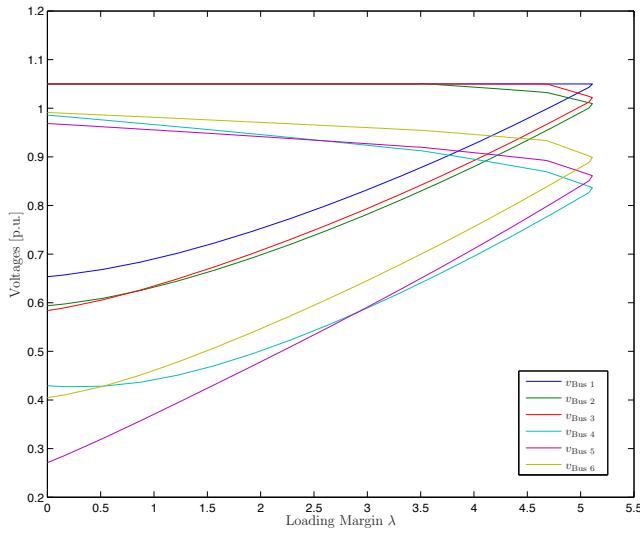


Figure 6.9: Nose curves for the 6-bus test system with generator reactive power limits. The maximum loading condition is given by a limit-induced bifurcation.

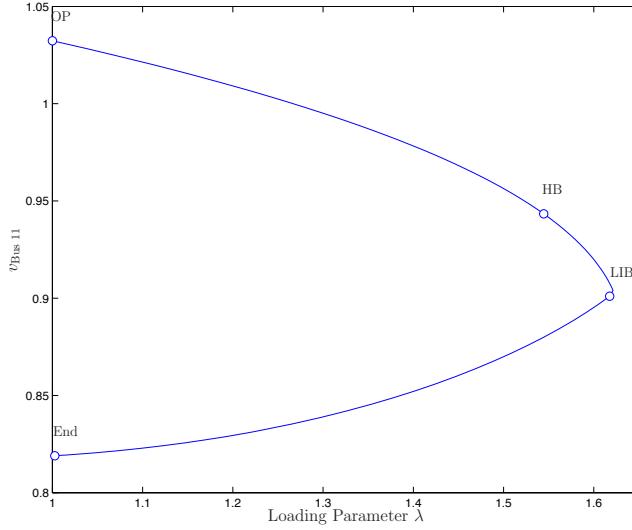


Figure 6.10: Nose curves for the 9-bus test system with dynamic models of synchronous generators and AVR_s. The stability limit is given by a Hopf bifurcation.

Table 6.1 illustrates the results of the $N - 1$ contingency analysis for the 6-bus test system. The output is organized in four columns: the first column depicts the transmission line or transformer while the second one shows for which line outage it has been found the minimum power in that line. The last two columns depict the actual power flow and the power flow limit, respectively, in the transmission line or transformer.

Table 6.1: $N - 1$ Contingency Analysis Report for the 6-bus test system

N-1 CONTINGENCY ANALYSIS						
P S A T 2.1.3						
Author: Federico Milano, (c) 2002-2008						
e-mail: Federico.Milano@uclm.es						
website: http://www.uclm.es/area/gsee/Web/Federico						
File: /Users/fmilano/Applications/psat2/tests/d_006_mdl						
Date: 12-Nov-2008 10:57:04						
POWER FLOW LIMITS						
Line	Outage of this line	Worst case line outage [p.u.]	P _{ij} base	P _{ij} max [p.u.]	S _{ij} base [p.u.]	S _{ij} max [p.u.]
2-3	Feasible	3-5	0.15013	0.03617	0.1618	0.04436
3-6	Unfeasible	2-5	0.50254	0.66986	0.71838	0.86285
4-5	Feasible	5-6	0.07867	0.03328	0.08698	0.03831
3-5	Feasible	2-3	0.24653	0.34162	0.31889	0.45633
5-6	Feasible	4-5	0.0199	0.00021	0.09683	0.08725
2-4	Unfeasible	3-5	0.60904	0.68989	0.71045	0.79409
1-2	Feasible	3-5	0.11245	0.09004	0.13601	0.09322
1-4	Unfeasible	3-5	0.40302	0.41724	0.4615	0.49851
1-5	Unfeasible	3-5	0.38453	0.42655	0.41873	0.4995
2-6	Unfeasible	3-5	0.44108	0.40119	0.46299	0.45016
2-5	Feasible	1-2	0.30954	0.34761	0.35258	0.44386

Chapter 7

Optimal Power Flow

This chapter describes the Optimal Power Flow (OPF) problem and its implementation in PSAT. The Interior Point Method (IPM) is used for solving the nonlinear set of equations of the OPF problem as described in [128]. A discussion of diverse objective functions and OPF models used in the program is presented along with a detailed description of the structures and the data needed to solve the OPF. Finally, a simple 6-bus system example is presented and the graphical user interface and text outputs are described.

7.1 Interior Point Method

In [58], several strategies were proposed for an OPF with active power dispatching and voltage security using an IPM that proved to be robust, especially in large networks, as the number of iterations increase slightly with the number of constraints and network size. Most implementations of IPM for solving market problems, accounting somewhat for system security, use a linear programming technique [122, 93, 4].

In [104] and [128], the authors present a comprehensive investigation of the use of IPM for non-linear problems, and describe the application of Newton's direction method and Merhotra's predictor-corrector to the OPF. The latter highly reduces the number of iterations to obtain the final solution. Both methods which were described in [128] are implemented in the IPM-NLP problem.

Non-linear optimization techniques have also been shown to be adequate for addressing a variety of voltage stability issues, such as the maximization of the loading parameter in voltage collapse studies, as discussed in [61], [18], [26] and [25]. In [78] and [77], non-linear IPM techniques are applied to the solution of diverse OPF market problems. The OPF routines implemented in the program also uses the techniques proposed in [24] and [87], where the authors proposed diverse methods to account for system security through the use of voltage stability based constraints in an OPF-IPM market representation, so that security is not simply modeled through the use of voltage and power transfer limits, typically determined off-line, but it is

properly represented in on-line market computations.

7.2 OPF Routines

In PSAT, three different objective functions are available: the maximization of the social benefit, the maximization of the distance to the maximum loading condition and also a multi-objective approach similar to the one proposed in [25]. The following sections describe each model and the constraints implemented and tested so far.¹ Section 7.2.4 presents the Lagrangian function which is minimized by means of the IPM-NLP method.

7.2.1 Maximization of the Social Benefit

The OPF-based approach is basically a non-linear constrained optimization problem, and consists of a scalar objective function and a set of equality and inequality constraints. A typical OPF-based market model can be represented using the following security constrained optimization problem (e.g. [138]):

$$\begin{aligned}
 \text{Min. } & G = -(\sum \mathbf{c}_D(\mathbf{p}_D) - \sum \mathbf{c}_S(\mathbf{p}_S)) && \rightarrow \text{Social benefit} && (7.1) \\
 \text{s.t. } & \mathbf{g}(\boldsymbol{\delta}, \mathbf{v}, \mathbf{q}_G, \mathbf{p}_S, \mathbf{p}_D) = \mathbf{0} && \rightarrow \text{PF equations} \\
 & \mathbf{p}_S^{\min} \leq \mathbf{p}_S \leq \mathbf{p}_S^{\max} && \rightarrow \text{Sup. bid blocks} \\
 & \mathbf{p}_D^{\max} \leq \mathbf{p}_D \leq \mathbf{p}_D^{\max} && \rightarrow \text{Dem. bid blocks} \\
 & |\phi_{ij}(\boldsymbol{\delta}, \mathbf{v})| \leq \phi_{ij}^{\max} && \rightarrow \text{Power transfer lim.} \\
 & |\phi_{ji}(\boldsymbol{\delta}, \mathbf{v})| \leq \phi_{ji}^{\max} && \\
 & \mathbf{q}_G^{\min} \leq \mathbf{q}_G \leq \mathbf{q}_G^{\max} && \rightarrow \text{Gen. } q \text{ lim.} \\
 & \mathbf{v}^{\min} \leq \mathbf{v} \leq \mathbf{v}^{\max} && \rightarrow \mathbf{v} \text{ "security" lim.}
 \end{aligned}$$

where \mathbf{c}_S and \mathbf{c}_D are vectors of supply and demand bids in \$/MWh, respectively; \mathbf{q}_G stand for the generator reactive powers; \mathbf{v} and $\boldsymbol{\delta}$ represent the bus phasor voltages; ϕ_{ij} and ϕ_{ji} represent the active powers (or apparent powers or currents) flowing through the lines in both directions; and \mathbf{p}_S and \mathbf{p}_D represent bounded supply and demand power bids in MW. In this model, which is typically referred to as a security constrained OPF, active power flow limits in transmission lines are typically obtained by means of off-line angle and/or voltage stability studies. In practice, these limits are usually determined based only on power flow based voltage stability studies [49] and can be determined using the $N - 1$ contingency analysis that is described in Chapter 6.

7.2.2 Maximization of the Distance to Collapse

The following optimization problem is implemented to properly represent system security through the use of voltage stability conditions, based on what was proposed

¹Some additional constraints can be included or will be included in future versions.

in [26], [25], [24]:

$$\begin{aligned}
 \text{Min.} \quad & G = -\lambda && (7.2) \\
 \text{s.t.} \quad & \mathbf{g}(\boldsymbol{\delta}, \mathbf{v}, \mathbf{q}_G, \mathbf{p}_S, \mathbf{p}_D) = \mathbf{0} && \rightarrow \text{PF equations} \\
 & \mathbf{g}^c(\boldsymbol{\delta}^c, \mathbf{v}^c, \mathbf{q}_G^c, \lambda_c, \mathbf{p}_S, \mathbf{p}_D) = \mathbf{0} && \rightarrow \text{Max load PF eqs.} \\
 & \lambda^{\min} \leq \lambda \leq \lambda^{\max} && \rightarrow \text{loading margin} \\
 & \mathbf{p}_S^{\min} \leq \mathbf{p}_S \leq \mathbf{p}_S^{\max} && \rightarrow \text{Sup. bid blocks} \\
 & \mathbf{p}_D^{\min} \leq \mathbf{p}_D \leq \mathbf{p}_D^{\max} && \rightarrow \text{Dem. bid blocks} \\
 & \phi_{ij}(\boldsymbol{\delta}, \mathbf{v}) \leq \phi_{ij}^{\max} && \rightarrow \text{Flow limits} \\
 & \phi_{ji}(\boldsymbol{\delta}, \mathbf{v}) \leq \phi_{ji}^{\max} && \\
 & \phi_{ij}(\boldsymbol{\delta}^c, \mathbf{v}^c) \leq \phi_{ij}^{\max} && \\
 & \phi_{ji}(\boldsymbol{\delta}^c, \mathbf{v}^c) \leq \phi_{ji}^{\max} && \\
 & \mathbf{q}_G^{\min} \leq \mathbf{q}_G \leq \mathbf{q}_G^{\max} && \rightarrow \text{Gen. } q \text{ limits} \\
 & \mathbf{q}_G^{\min} \leq \mathbf{q}_G^c \leq \mathbf{q}_G^{\max} && \\
 & \mathbf{v}^{\min} \leq \mathbf{v} \leq \mathbf{v}^{\max} && \rightarrow v \text{ "security" lim.} \\
 & \mathbf{v}^{\min} \leq \mathbf{v}^c \leq \mathbf{v}^{\max} &&
 \end{aligned}$$

In this case, a second set of power flow equations and constraints with a superscript c is introduced to represent the system at the limit or *critical* conditions associated with the loading margin λ that drives the system to its maximum loading condition. The critical power flow equations \mathbf{g}^c can present a line outage. The maximum or critical loading point could be either associated with a thermal or bus voltage limit or a voltage stability limit (collapse point) corresponding to a system singularity (saddle-node bifurcation) or system controller limits like generator reactive power limits (limit induced bifurcation) [20, 108]. Thus, for the current and maximum loading conditions, the generator and load powers are defined as follows:

$$\begin{aligned}
 \mathbf{p}_G &= \mathbf{p}_G^0 + \mathbf{p}_S && (7.3) \\
 \mathbf{p}_L &= \mathbf{p}_L^0 + \mathbf{p}_D \\
 \mathbf{p}_G^c &= (1 + \lambda + k_G^c) \mathbf{p}_G \\
 \mathbf{p}_L^c &= (1 + \lambda) \mathbf{p}_L
 \end{aligned}$$

where \mathbf{p}_G^0 and \mathbf{p}_L^0 stand for generator and load powers which are not part of the market bidding (e.g. must-run generators, inelastic loads), and k_G^c represents a scalar variable which distributes system losses associated *only* with the solution of the critical power flow equations in proportion to the power injections obtained in the solution process (distributed slack bus model). It is assumed that the losses corresponding to the maximum loading level defined by λ in (7.2) are distributed among all generators. Observe that power directions (7.3) used in the voltage stability constrained OPF differ from (6.2), i.e. the power directions used in the bifurcation analysis presented in Chapter 6.

7.2.3 Multi-Objective Optimization

A multi-objective optimization is also implemented, based on what was proposed in [87], so that system security which is modeled through the use of voltage stability conditions is combined with the electricity market:

$$\begin{aligned}
 \text{Min. } & G = -\omega_1(\sum \mathbf{c}_D(\mathbf{p}_D) - \sum \mathbf{c}_S(\mathbf{p}_S)) - \omega_2 \lambda & (7.4) \\
 \text{s.t. } & \mathbf{g}(\boldsymbol{\delta}, \mathbf{v}, \mathbf{q}_G, \mathbf{p}_S, \mathbf{p}_D) = \mathbf{0} & \rightarrow \text{PF equations} \\
 & \mathbf{g}^c(\boldsymbol{\delta}^c, \mathbf{v}^c, \mathbf{q}_G^c, \lambda, \mathbf{p}_S, \mathbf{p}_D) = \mathbf{0} & \rightarrow \text{Max load PF eqs.} \\
 & \lambda^{\min} \leq \lambda \leq \lambda^{\max} & \rightarrow \text{loading margin} \\
 & \mathbf{p}_S^{\min} \leq \mathbf{p}_S \leq \mathbf{p}_S^{\max} & \rightarrow \text{Sup. bid blocks} \\
 & \mathbf{p}_D^{\min} \leq \mathbf{p}_D \leq \mathbf{p}_D^{\max} & \rightarrow \text{Dem. bid blocks} \\
 & \phi_{ij}(\boldsymbol{\delta}, \mathbf{v}) \leq \phi_{ij}^{\max} & \rightarrow \text{Thermal limits} \\
 & \phi_{ji}(\boldsymbol{\delta}, \mathbf{v}) \leq \phi_{ij}^{\max} & \\
 & \phi_{ij}(\boldsymbol{\delta}^c, \mathbf{v}^c) \leq \phi_{ij}^{\max} & \\
 & \phi_{ji}(\boldsymbol{\delta}^c, \mathbf{v}^c) \leq \phi_{ij}^{\max} & \\
 & \mathbf{q}_G^{\min} \leq \mathbf{q}_G \leq \mathbf{q}_G^{\max} & \rightarrow \text{Gen. } q \text{ limits} \\
 & \mathbf{q}_G^{\min} \leq \mathbf{q}_G^c \leq \mathbf{q}_G^{\max} & \\
 & \mathbf{v}^{\min} \leq \mathbf{v} \leq \mathbf{v}^{\max} & \rightarrow \mathbf{v} \text{ "security" lim.} \\
 & \mathbf{v}^{\min} \leq \mathbf{v}^c \leq \mathbf{v}^{\max} &
 \end{aligned}$$

In the multi-objective function, two terms are present, with their influence on the final solution being determined by the value of the weighting factors ω_1 and ω_2 ($\omega_1 > 0$, $\omega_2 > 0$). The first term represents the social benefit, whereas the second term guarantees that the "distance" between the market solution and the critical point is maximized [26]. Observe that $\omega_1 > 0$, since for $\omega_1 = 0$ there would be no representation of the market in the proposed OPF formulation, rendering it useless. Furthermore, $\omega_2 > 0$, otherwise λ_c will not necessarily correspond to a maximum loading condition of the system. Notice that the two terms of the objective function are expressed in different units, since the social benefit would be typically in \$/h, whereas the "security" term would be in p.u., which will basically affect the chosen values of ω_1 and ω_2 (typically, $\omega_1 \gg \omega_2$). However, it is possible to assume that $\omega_1 = (1-\omega)$ and $\omega_2 = \omega$, with proper scaled values of ω for each system under study ($0 < \omega < 1$), as this simplifies the optimization problem without losing generality.

The representation of the generator and load powers in (7.4) is the same as in (7.3).

7.2.4 Lagrangian Function

Internally, the program represents the previous problems (7.1), (7.2) and (7.4) with the same model, ignoring the constraints that are not used or assuming proper values for the parameters that are not defined. The following general Lagrangian function is minimized:

$$\begin{aligned}
\text{Min. } \mathcal{L} = & G - \boldsymbol{\rho}^T \mathbf{g}(\boldsymbol{\delta}, \mathbf{v}, \mathbf{q}_G, \mathbf{p}_S, \mathbf{p}_D) \\
& - \boldsymbol{\rho}_c^T \mathbf{g}^c(\boldsymbol{\delta}^c, \mathbf{v}^c, \mathbf{q}_G^c, \lambda, \mathbf{p}_S, \mathbf{p}_D) \\
& - \mu_{\lambda^{\max}} (\lambda^{\max} - \lambda - s_{\lambda^{\max}}) \\
& - \mu_{\lambda^{\min}} (\lambda - \lambda^{\min} - s_{\lambda^{\min}}) \\
& - \boldsymbol{\mu}_{p_S^{\max}}^T (\mathbf{p}_S^{\max} - \mathbf{p}_S - \mathbf{s}_{p_S^{\max}}) \\
& - \boldsymbol{\mu}_{p_S^{\min}}^T (\mathbf{p}_S - \mathbf{p}_S^{\min} - \mathbf{s}_{p_S^{\min}}) \\
& - \boldsymbol{\mu}_{p_D^{\max}}^T (\mathbf{p}_D^{\max} - \mathbf{p}_D - \mathbf{s}_{p_D^{\max}}) \\
& - \boldsymbol{\mu}_{p_D^{\min}}^T (\mathbf{p}_D - \mathbf{p}_D^{\min} - \mathbf{s}_{p_D^{\min}}) \\
& - \boldsymbol{\mu}_{\phi_{ij}^{\max}}^T (\phi_{ij}^{\max} - \phi_{ij} - \mathbf{s}_{\phi_{ij}^{\max}}) \\
& - \boldsymbol{\mu}_{\phi_{ji}^{\max}}^T (\phi_{ji}^{\max} - \phi_{ji} - \mathbf{s}_{\phi_{ji}^{\max}}) \\
& - \boldsymbol{\mu}_{\phi_{ij}^{c\max}}^T (\phi_{ij}^{c\max} - \phi_{ijc} - \mathbf{s}_{\phi_{ij}^{c\max}}) \\
& - \boldsymbol{\mu}_{\phi_{ji}^{c\max}}^T (\phi_{ji}^{c\max} - \phi_{jic} - \mathbf{s}_{\phi_{ji}^{c\max}}) \\
& - \boldsymbol{\mu}_{q_G^{\max}}^T (\mathbf{q}_G^{\max} - \mathbf{q}_G - \mathbf{s}_{q_G^{\max}}) \\
& - \boldsymbol{\mu}_{q_G^{\min}}^T (\mathbf{q}_G - \mathbf{q}_G^{\min} - \mathbf{s}_{q_G^{\min}}) \\
& - \boldsymbol{\mu}_{q_G^{c\max}}^T (\mathbf{q}_G^{\max} - \mathbf{q}_G^c - \mathbf{s}_{q_G^{c\max}}) \\
& - \boldsymbol{\mu}_{q_G^{c\min}}^T (\mathbf{q}_G^c - \mathbf{q}_G^{\min} - \mathbf{s}_{q_G^{c\min}}) \\
& - \boldsymbol{\mu}_{v^{\max}}^T (\mathbf{v}^{\max} - \mathbf{v} - \mathbf{s}_{v^{\max}}) \\
& - \boldsymbol{\mu}_{v^{\min}}^T (\mathbf{v} - \mathbf{v}^{\min} - \mathbf{s}_{v^{\min}}) \\
& - \boldsymbol{\mu}_{v^{c\max}}^T (\mathbf{v}^{\max} - \mathbf{v}^c - \mathbf{s}_{v^{c\max}}) \\
& - \boldsymbol{\mu}_{v^{c\min}}^T (\mathbf{v}^c - \mathbf{v}^{\min} - \mathbf{s}_{v^{c\min}}) - \mu_s (\sum \ln \mathbf{s})
\end{aligned} \tag{7.5}$$

where $\mu_s \in \mathbb{R}$, $\mu_s > 0$, is the barrier parameter, and $\boldsymbol{\rho}$ and $\boldsymbol{\rho}^c \in \mathbb{R}^n$, and all the other $\boldsymbol{\mu}$ ($\mu_i > 0$, $\forall i$) correspond to the Lagrangian multipliers. The \mathbf{s} variables form the slack vector whose non-negativity condition ($s_i > 0$, $\forall i$) is ensured by including the logarithmic barrier terms $\sum \ln \mathbf{s}$.

7.3 OPF Settings

Figure 7.1 depicts the GUI for settings OPF parameters (menu *Edit/OPF Settings* or shortcut *<Ctrl-z>* in the main window). For a detailed description of the parameters used for the IPM refer to [128]. The parameters and the results of OPF

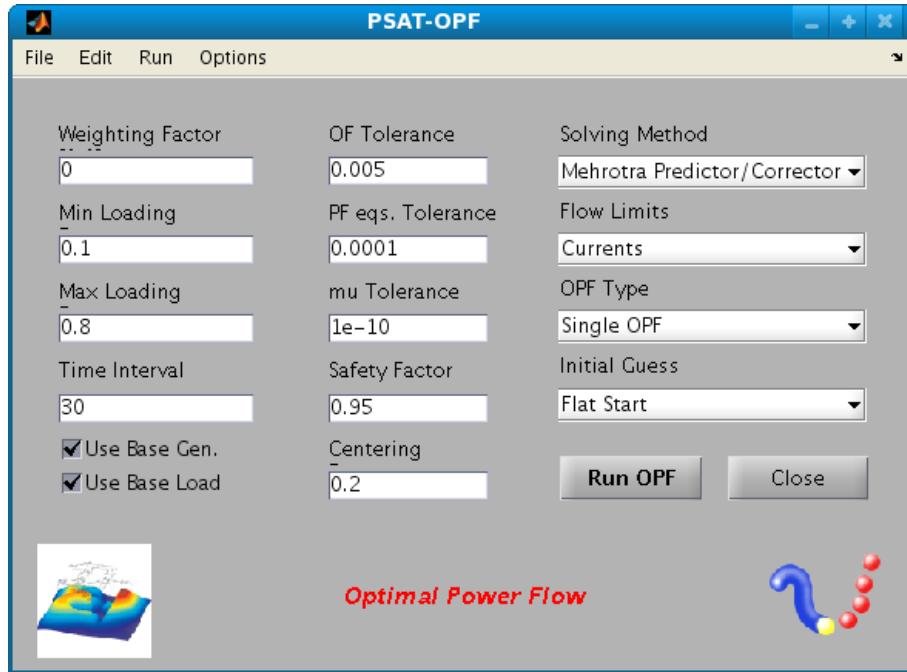


Figure 7.1: GUI for OPF analysis.

computations are contained in the structure `OPF`, which is described in Appendix A.

A special care has to be devoted to the status of the check boxes “Use Base Gen.” and “Use Base Load”. If these check boxes are active, base powers p_G^0 and p_L^0 are included in the OPF analysis, otherwise they are ignored. Improper settings can lead to unfeasible OPF or inconsistent results. The user should know whether it is the case of including base powers or not depending on the power limits that have been set in the supply and demand blocks.

7.4 Example

This section depicts OPF results for a 6-bus test system. The complete set of data for the 6-bus test system are reported in Appendix F.

OPF results can be displayed in the same GUI which is used for power flow results. The GUI will display the total transaction level and total bid losses, as well as the current voltages and power flows.

OPF results can be saved using the *Report* button. A log file will be created using the selected format (plain text, L^AT_EX, Excel) and displayed with the selected viewer (see Section 27.6 for details). For example, the plain text power flow solution for the 6-bus test system with $\omega = 0$ (standard OPF) is as follows:

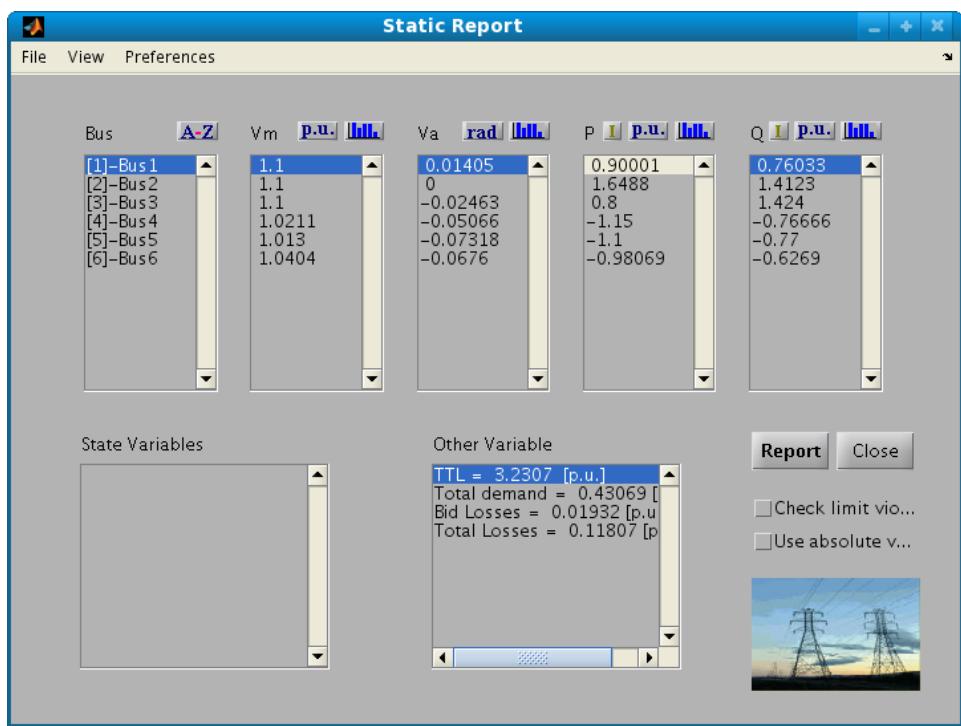


Figure 7.2: GUI for displaying OPF results.

OPTIMAL POWER FLOW REPORT
(Standard OPF)

P S A T 2.1.3

Author: Federico Milano, (c) 2002-2008
e-mail: Federico.Milano@uclm.es
website: <http://www.uclm.es/area/gsee/Web/Federico>

File: ~/Applications/psat2/tests/d_006.mdl
Date: 05-Nov-2008 23:18:46

NETWORK STATISTICS

Buses:	6
Lines:	11
Generators:	3
Loads:	3
Supplies:	3
Demands:	3

SOLUTION STATISTICS

Objective Function [\$/h]:	-121.6472
Active Limits:	8
Number of Iterations:	11
Barrier Parameter:	0
Variable Mismatch:	0
Power Flow Equation Mismatch	0
Objective Function Mismatch:	0

POWER SUPPLIES

Bus	mu min	Ps min [MW]	Ps [MW]	Ps max [MW]	mu max
Bus1	0.67964	0.001	0.001	20	0
Bus2	0	0.001	25	25	0.18047
Bus3	0	0.001	20	20	2.1455

POWER DEMANDS

Bus	mu min	Pd min [MW]	Pd [MW]	Pd max [MW]	mu max
Bus6	0	0.001	8.0693	20	0
Bus5	0	0.001	10	10	0.56118
Bus4	0	0.001	25	25	2.175

REACTIVE POWERS

Bus	mu min	Qg min [MVar]	Qg [MVar]	Qg max [MVar]	mu max
Bus1	0	-150	44.6239	150	0
Bus2	0	-150	76.207	150	0

Bus3	0	-150	72.0845	150	0	
VOLTAGES						
Bus	mu min	V min [p.u.]	V [p.u.]	V max [p.u.]	mu max	phase [rad]
Bus1	0	0.9	1.1	1.1	1.4535	0.01405
Bus2	0	0.9	1.1	1.1	0.73624	0
Bus3	0	0.9	1.1	1.1	0.30313	-0.02463
Bus4	0	0.9	1.0211	1.1	0	-0.05066
Bus5	0	0.9	1.013	1.1	0	-0.07318
Bus6	0	0.9	1.0404	1.1	0	-0.0676
POWER FLOW						
Bus	P [MW]	Q [MVar]	rho P [\$/MWh]	rho Q [\$/MVArh]	NCP [\$/MWh]	Pay [\$/h]
Bus1	90.001	44.6239	9.0204	1e-05	-0.04509	-812
Bus2	164.8754	76.207	8.9805	2e-05	0	-1481
Bus3	80	72.0845	9.1455	2e-05	0.07117	-732
Bus4	-115	-76.6665	9.563	0.39308	0.19391	1100
Bus5	-110	-77	9.6535	0.40764	0.27128	1062
Bus6	-98.0693	-62.6897	9.4284	0.21474	0.22339	925
FLOWS IN TRANSMISSION LINES						
From bus	To bus	I_ij [p.u.]	I_ji [p.u.]	I_ij max [p.u.]	mu I_ij	mu I_ji
Bus2	Bus3	0.11693	0.10451	0.3082	0	0
Bus3	Bus6	0.731	0.74506	1.3973	0	0
Bus4	Bus5	0.07149	0.06342	0.1796	0	0
Bus3	Bus5	0.33729	0.36729	0.6585	0	0
Bus5	Bus6	0.11578	0.0635	0.2	0	0
Bus2	Bus4	0.84776	0.8581	1.374	0	0
Bus1	Bus2	0.08127	0.06232	0.2591	0	0
Bus1	Bus4	0.49408	0.51836	0.9193	0	0
Bus1	Bus5	0.39214	0.42224	0.8478	0	0
Bus2	Bus6	0.4327	0.45115	0.9147	0	0
Bus2	Bus5	0.35683	0.3779	0.7114	0	0
TOTALS						
TOTAL LOSSES [MW]:		11.807				
BID LOSSES [MW]		1.932				
TOTAL DEMAND [MW]:		43.0693				
TOTAL TRANSACTION LEVEL [MW]		323.0693				
IMO PAY [\$/h]:		62.1225				

In the OPF report, the LMP (Locational Marginal Prices) column are dual variables $-\rho$, while the NCP column indicates Nodal Congestion Prices that are computed as follows. Using the decomposition formula for LMPs which has been

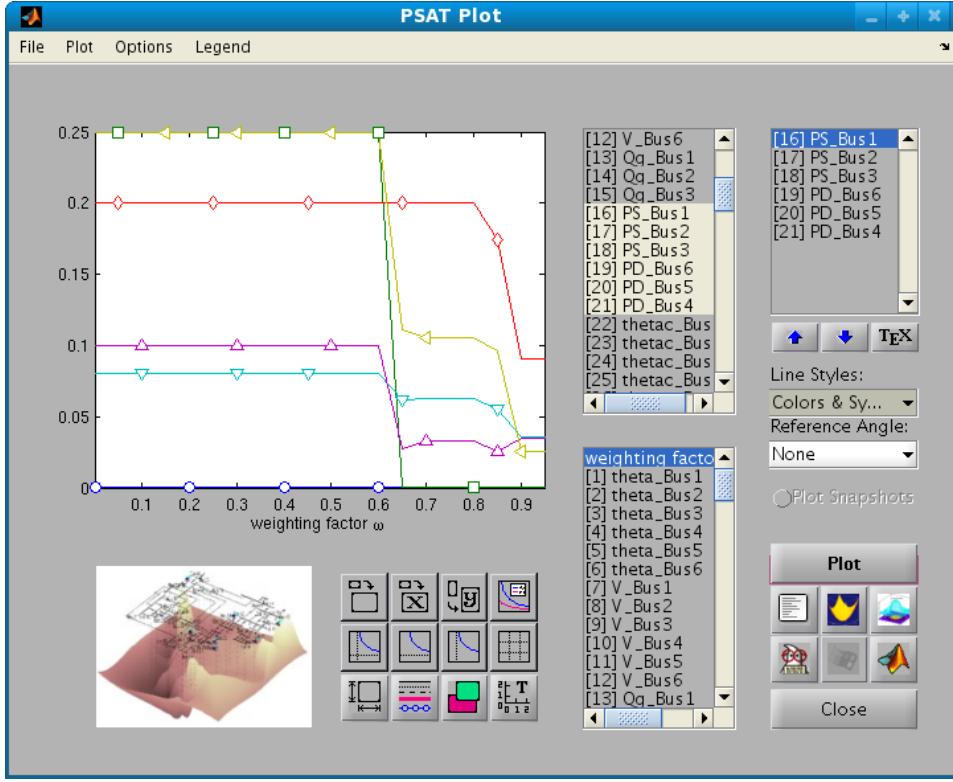


Figure 7.3: GUI for plotting OPF Pareto's sets.

proposed in [138], one has:

$$\text{NCP} = \mathbf{g}_y^{-1} \mathbf{h}_y^T (\boldsymbol{\mu}^{\max} - \boldsymbol{\mu}^{\min}) \quad (7.6)$$

where \mathbf{h} represents the inequality constraints of transmission lines, and $\boldsymbol{\mu}^{\max}$ and $\boldsymbol{\mu}^{\min}$ are the dual variables or shadow prices associated to such inequality constraints. Equation (7.6) is a vector of active and reactive nodal congestion prices.

Figure 7.3 depicts the graphical user interface for plotting the Pareto's set, which can be obtained by setting a vector of values for the weighting factor ω .

The GUI allows tuning a variety of parameters and settings, such as choosing the variables to plot, customizing the graphical appearance, adding and modifying a legend of the plotted variables and saving the graph to a color .eps file, which is placed in the folder of the current data file and automatically named with a progressive number (from 00 to 99).

Chapter 8

Small Signal Stability Analysis

This chapter describes small signal stability analysis available in PSAT and the associated graphical user interface. After solving the power flow problem, it is possible to compute and visualize the eigenvalues and the participation factors of the system. The eigenvalues can be computed for the state matrix of the dynamic system (small signal stability analysis) [111, 60], and for three different types of power flow Jacobian matrices (*qv* sensitivity analysis) [139].

The following sections describe the main features of the small signal stability analysis and of the power flow Jacobian eigenvalue analysis.

8.1 Small Signal Stability Analysis

The system used for the small signal stability analysis is a differential algebraic equation (DAE) set, in the form:

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}, \mathbf{y}) \\ \mathbf{0} &= \mathbf{g}(\mathbf{x}, \mathbf{y})\end{aligned}\tag{8.1}$$

where \mathbf{x} is the vector of the state variables and \mathbf{y} the vector of the algebraic variables, \mathbf{f} is the vector of differential equations, and \mathbf{g} is the vector of algebraic equations. Small signal stability studies the properties of equilibrium points (i.e. $\dot{\mathbf{x}} = 0$) of the DAE through an eigenvalue analysis of the state matrix \mathbf{A}_S of the system.

The state matrix \mathbf{A}_S is computed by manipulating the complete Jacobian matrix \mathbf{A}_C , that is defined by the linearization of the DAE system equations at an equilibrium point (8.1):

$$\begin{bmatrix} \Delta \dot{\mathbf{x}} \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_x & \mathbf{f}_y \\ \mathbf{g}_x & \mathbf{g}_y \end{bmatrix} \begin{bmatrix} \Delta \mathbf{x} \\ \Delta \mathbf{y} \end{bmatrix} = [\mathbf{A}_C] \begin{bmatrix} \Delta \mathbf{x} \\ \Delta \mathbf{y} \end{bmatrix}\tag{8.2}$$

where the Jacobian matrices that form the \mathbf{A}_C matrix are:

$$\begin{aligned}\mathbf{f}_x &\triangleq \nabla_{\mathbf{x}} \mathbf{f} \\ \mathbf{f}_y &\triangleq \nabla_{\mathbf{y}} \mathbf{f} \\ \mathbf{g}_x &\triangleq \nabla_{\mathbf{x}} \mathbf{g} \\ \mathbf{g}_y &\triangleq \nabla_{\mathbf{y}} \mathbf{g}\end{aligned}\tag{8.3}$$

The Jacobian matrix \mathbf{g}_y is the complete gradient of the algebraic equations, and contains the power flow Jacobian matrix. Other two types of power flow Jacobian matrices are defined in PSAT, namely \mathbf{J}_{LF} and \mathbf{J}_{LFD} , which are described in the next section.

The state matrix \mathbf{A}_S is simply obtained by eliminating the algebraic variables, and thus implicitly assuming that \mathbf{J}_{LFV} is non-singular (i.e. absence of singularity-induced bifurcations):

$$\mathbf{A}_S = \mathbf{f}_x - \mathbf{f}_y \mathbf{g}_y^{-1} \mathbf{g}_x\tag{8.4}$$

The computation of all eigenvalues can be a lengthy process if the dynamic order of the system is high. At this aim, it is possible to compute only a few eigenvalues with a particular property, i.e. largest or smallest magnitude, largest or smaller real or imaginary part.

When all the eigenvalues are computed, it is also possible to obtain the participation factors, that are evaluated in the following way. Let \mathbf{N} and \mathbf{W} be the right and the left eigenvector matrices respectively, such that $\mathbf{\Lambda} = \mathbf{W} \mathbf{A}_S \mathbf{N}$ and $\mathbf{W} = \mathbf{V}^{-1}$, then the participation factor p_{ij} of the i^{th} state variable to the j^{th} eigenvalue can be defined as:

$$p_{ij} = \frac{w_{ij} \nu_{ji}}{\mathbf{w}_j^T \boldsymbol{\nu}_j}\tag{8.5}$$

In case of complex eigenvalues, the amplitude of each element of the eigenvectors is used:

$$p_{ij} = \frac{|w_{ij}| |\nu_{ji}|}{\sum_{k=1}^n |w_{jk}| |\nu_{kj}|}\tag{8.6}$$

No normalization of the participation factors is performed.

The state matrix in (8.4) leads to the computation of the eigenvalues in the S -domain, i.e., the system is stable if the real part of the eigenvalues is less than 0. It is sometime useful to compute the eigenvalues in the Z -domain, which can also ease the visualization of very stiff systems. In this way, if the system is stable, all the eigenvalues are inside the unit circle. For the Z -domain eigenvalue computation, a bi-linear transformation is performed:

$$\mathbf{A}_Z = (\mathbf{A}_S + \rho \mathbf{I}_n)(\mathbf{A}_S - \rho \mathbf{I}_n)^{-1}\tag{8.7}$$

where ρ is a weighting factor, that in the program is set to 8. Even though more expensive, \mathbf{A}_Z can be useful for fastening the determination of the maximum amplitude eigenvalue (by means for example of a power method), especially in case of unstable equilibrium points with only one eigenvalue outside the unit circle.

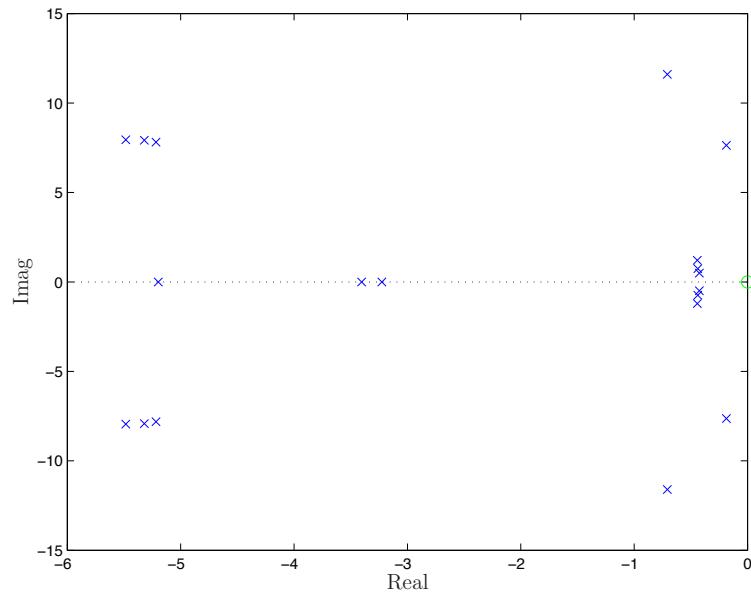


Figure 8.1: Eigenvalue Analysis: S -domain.

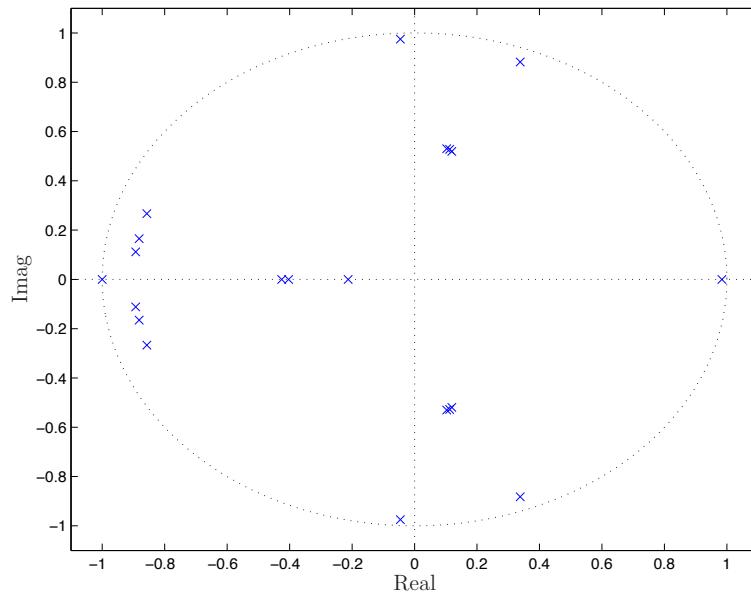


Figure 8.2: Eigenvalue Analysis: Z -domain.

8.1.1 Example

Figures 8.1 and 8.2, and the small signal stability report below depict the eigenvalue analysis for the WSCC 9-bus test system and have been generated with the Eigenvalue Analysis interface available in the View menu of the main window. Please refer to Subsection F.3 for the data of the WSCC 9-bus system. The report shows the pseudo-frequency and the frequency of the eigenvalues, which are defined as follows. Let $\alpha \pm j\beta$ be a pair of complex conjugate eigenvalues. The frequency ω_0 is defined as:

$$\omega_0 = \sqrt{\alpha^2 + \beta^2} \quad (8.8)$$

while the damping is:

$$\zeta = -\frac{\alpha}{\omega_0} \quad (8.9)$$

where $\zeta \in [-1, 1]$. The damping is positive if the mode is stable (i.e. $\alpha < 0$). Thus, one has:

$$\begin{aligned} \alpha &= -\zeta\omega_0 \\ \beta &= \sqrt{1 - \zeta^2}\omega_0 \end{aligned} \quad (8.10)$$

The frequency ω_0 is also called the frequency of resonance, or undamped frequency. However, the frequency that can be observed during the transient, namely the pseudo-frequency, depends on the damping ζ , as follows:

$$\omega_p = \sqrt{1 - \zeta^2}\omega_0 = \beta \quad (8.11)$$

The frequency of resonance and the pseudo-frequency are equal only if $\alpha = 0$.

The values of the frequency are computed using the system frequency base f_b . By default, PSAT uses the European frequency rate (50 Hz), which can be not consistent with the frequency that is used in American and some Asiatic systems (60 Hz). PSAT does not complain if the device frequency rates do not match the system base f_b , however the eigenvalue analysis will be solved using f_b .

```
EIGENVALUE REPORT
PSAT 2.1.3

Author: Federico Milano, (c) 2002-2008
e-mail: Federico.Milano@uclm.es
website: http://www.uclm.es/area/gsee/Web/Federico

File: ~/Applications/psat2/tests/d_009.mdl
Date: 05-Nov-2008 23:08:52
```

STATE MATRIX EIGENVALUES

Eigenvector	Most Associated States	Real part	Imag. Part	Pseudo-Freq.	Frequency
Eig As # 1	vm_Exc_1	-1000	0	0	0
Eig As # 2	vm_Exc_1	-1000	0	0	0
Eig As # 3	vm_Exc_3	-1000	0	0	0
Eig As # 4	delta_Syn_3, omega_Syn_3	-0.72015	12.7454	2.0285	2.0317
Eig As # 5	delta_Syn_3, omega_Syn_3	-0.72015	-12.7454	2.0285	2.0317
Eig As # 6	delta_Syn_2, omega_Syn_2	-0.19077	8.3658	1.3315	1.3318

Eig As # 7	delta_Syn_2, omega_Syn_2	-0.19077	-8.3658	1.3315	1.3318
Eig As # 8	vr1_Exc_2, vf_Exc_2	-5.4874	7.9474	1.2649	1.5371
Eig As # 9	vr1_Exc_2, vf_Exc_2	-5.4874	-7.9474	1.2649	1.5371
Eig As #10	vr1_Exc_1, vf_Exc_1	-5.2226	7.8139	1.2436	1.4958
Eig As #11	vr1_Exc_1, vf_Exc_1	-5.2226	-7.8139	1.2436	1.4958
Eig As #12	vr1_Exc_3, vf_Exc_3	-5.3237	7.9208	1.2606	1.5189
Eig As #13	vr1_Exc_3, vf_Exc_3	-5.3237	-7.9208	1.2606	1.5189
Eig As #14	e1d_Syn_2	-5.178	0	0	0
Eig As #15	e1d_Syn_3	-3.3996	0	0	0
Eig As #16	e1q_Syn_1, vr2_Exc_1	-0.44366	1.2111	0.19276	0.20528
Eig As #17	e1q_Syn_1, vr2_Exc_1	-0.44366	-1.2111	0.19276	0.20528
Eig As #18	e1q_Syn_1, e1q_Syn_2	-0.4391	0.73945	0.11769	0.13687
Eig As #19	e1q_Syn_1, e1q_Syn_2	-0.4391	-0.73945	0.11769	0.13687
Eig As #20	e1q_Syn_3, vr2_Exc_3	-0.4257	0.4961	0.07896	0.10404
Eig As #21	e1q_Syn_3, vr2_Exc_3	-0.4257	-0.4961	0.07896	0.10404
Eig As #22	omega_Syn_1	0	0	0	0
Eig As #23	delta_Syn_1	0	0	0	0
Eig As #24	e1d_Syn_1	-3.2258	0	0	0

PARTICIPATION FACTORS (Euclidean norm)

	delta_Syn_1	omega_Syn_1	e1q_Syn_1	e1d_Syn_1	delta_Syn_2
Eig As # 1	0	0	0	0	0
Eig As # 2	0	0	0	0	0
Eig As # 3	0	0	0	0	0
Eig As # 4	0.00462	0.00462	1e-05	0	0.08564
Eig As # 5	0.00462	0.00462	1e-05	0	0.08564
Eig As # 6	0.12947	0.12947	4e-05	0	0.30889
Eig As # 7	0.12947	0.12947	4e-05	0	0.30889
Eig As # 8	0.00021	0.00021	0.00022	0	0.00102
Eig As # 9	0.00021	0.00021	0.00022	0	0.00102
Eig As #10	0.00015	0.00015	0.01788	0	0.00014
Eig As #11	0.00015	0.00015	0.01788	0	0.00014
Eig As #12	0.00019	0.00019	0.00111	0	4e-05
Eig As #13	0.00019	0.00019	0.00111	0	4e-05
Eig As #14	2e-05	2e-05	0	0	0.00677
Eig As #15	0.00061	0.00061	0.00151	0	0.00098
Eig As #16	0.00021	0.00021	0.23865	0	0.00068
Eig As #17	0.00021	0.00021	0.23865	0	0.00068
Eig As #18	0.00025	0.00025	0.24347	0	0.0007
Eig As #19	0.00025	0.00025	0.24347	0	0.0007
Eig As #20	0	0	0.00265	0	0.00055
Eig As #21	0	0	0.00265	0	0.00055
Eig As #22	0.36234	0.36234	0	0	0.09334
Eig As #23	0.36234	0.36234	0	0	0.09334
Eig As #24	0	0	0	1	0

PARTICIPATION FACTORS (Euclidean norm)

	omega_Syn_2	e1q_Syn_2	e1d_Syn_2	delta_Syn_3	omega_Syn_3
Eig As # 1	0	0	0	0	0
Eig As # 2	0	0	0	0	0
Eig As # 3	0	0	0	0	0
Eig As # 4	0.08564	0.00492	0.0053	0.38243	0.38243
Eig As # 5	0.08564	0.00492	0.0053	0.38243	0.38243
Eig As # 6	0.30889	0.01325	0.00609	0.04918	0.04918
Eig As # 7	0.30889	0.01325	0.00609	0.04918	0.04918
Eig As # 8	0.00102	0.01366	0.0017	0.0003	0.0003
Eig As # 9	0.00102	0.01366	0.0017	0.0003	0.0003
Eig As #10	0.00014	0.00229	0.00117	0.00033	0.00033
Eig As #11	0.00014	0.00229	0.00117	0.00033	0.00033
Eig As #12	4e-05	0.00162	0.00055	0.00117	0.00117
Eig As #13	4e-05	0.00162	0.00055	0.00117	0.00117
Eig As #14	0.00677	0.00778	0.4859	0.0104	0.0104
Eig As #15	0.00098	0.00088	0.44865	0.00453	0.00453
Eig As #16	0.00068	0.16013	0.01106	0.00041	0.00041

Eig As #17	0.00068	0.16013	0.01106	0.00041	0.00041
Eig As #18	0.0007	0.18938	0.00948	0.00018	0.00018
Eig As #19	0.0007	0.18938	0.00948	0.00018	0.00018
Eig As #20	0.00055	0.13702	0.00902	0.00121	0.00121
Eig As #21	0.00055	0.13702	0.00902	0.00121	0.00121
Eig As #22	0.09334	0	0	0.04432	0.04432
Eig As #23	0.09334	0	0	0.04432	0.04432
Eig As #24	0	0	0	0	0

PARTICIPATION FACTORS (Euclidean norm)

	e1q_Syn_3	e1d_Syn_3	vm_Exc_1	vr1_Exc_1	vr2_Exc_1
Eig As # 1	0	0	0.41106	0	0
Eig As # 2	0	0	0.58338	0	0
Eig As # 3	0	0	0.00555	0	0
Eig As # 4	0.01186	0.03092	0	1e-05	0
Eig As # 5	0.01186	0.03092	0	1e-05	0
Eig As # 6	0.0018	0.00035	0	3e-05	1e-05
Eig As # 7	0.0018	0.00035	0	3e-05	1e-05
Eig As # 8	0.00073	0.00052	0	0.00687	0.00195
Eig As # 9	0.00073	0.00052	0	0.00687	0.00195
Eig As # 10	0.00383	0.00045	0.00017	0.34843	0.1066
Eig As # 11	0.00383	0.00045	0.00017	0.34843	0.1066
Eig As # 12	0.01081	0.00273	1e-05	0.06426	0.01874
Eig As # 13	0.01081	0.00273	1e-05	0.06426	0.01874
Eig As # 14	0.01498	0.44691	0	0	1e-05
Eig As # 15	0.00332	0.50149	1e-05	0	0.01284
Eig As # 16	0.09173	0.00622	0.00031	0.02484	0.17647
Eig As # 17	0.09173	0.00622	0.00031	0.02484	0.17647
Eig As # 18	0.0544	0.00512	0.00019	0.02227	0.18912
Eig As # 19	0.0544	0.00512	0.00019	0.02227	0.18912
Eig As # 20	0.32208	0.03651	0	0.00023	0.0021
Eig As # 21	0.32208	0.03651	0	0.00023	0.0021
Eig As # 22	0	0	0	0	0
Eig As # 23	0	0	0	0	0
Eig As # 24	0	0	0	0	0

PARTICIPATION FACTORS (Euclidean norm)

	vf_Exc_1	vm_Exc_2	vr1_Exc_2	vr2_Exc_2	vf_Exc_2
Eig As # 1	0	0.32142	0	0	0
Eig As # 2	0	0.28859	0	0	0
Eig As # 3	0	0.39	0	0	0
Eig As # 4	1e-05	0	0.00016	1e-05	0.00017
Eig As # 5	1e-05	0	0.00016	1e-05	0.00017
Eig As # 6	3e-05	1e-05	0.00113	0.00026	0.00119
Eig As # 7	3e-05	1e-05	0.00113	0.00026	0.00119
Eig As # 8	0.00665	0.00013	0.38766	0.11191	0.38126
Eig As # 9	0.00665	0.00013	0.38766	0.11191	0.38126
Eig As # 10	0.33911	2e-05	0.0209	0.00649	0.02064
Eig As # 11	0.33911	2e-05	0.0209	0.00649	0.02064
Eig As # 12	0.06243	2e-05	0.02231	0.0066	0.022
Eig As # 13	0.06243	2e-05	0.02231	0.0066	0.022
Eig As # 14	0	2e-05	0.00018	0.00441	3e-05
Eig As # 15	4e-05	1e-05	1e-05	0.00889	3e-05
Eig As # 16	0.03704	0.00021	0.01908	0.11572	0.02429
Eig As # 17	0.03704	0.00021	0.01908	0.11572	0.02429
Eig As # 18	0.03411	0.00016	0.02045	0.14627	0.02638
Eig As # 19	0.03411	0.00016	0.02045	0.14627	0.02638
Eig As # 20	0.00036	8e-05	0.01432	0.1072	0.01855
Eig As # 21	0.00036	8e-05	0.01432	0.1072	0.01855
Eig As # 22	0	0	0	0	0
Eig As # 23	0	0	0	0	0
Eig As # 24	0	0	0	0	0

PARTICIPATION FACTORS (Euclidean norm)

	vm_Exc_3	vr1_Exc_3	vr2_Exc_3	vf_Exc_3
Eig As # 1	0.26752	0	0	0
Eig As # 2	0.12803	0	0	0
Eig As # 3	0.60445	0	0	0
Eig As # 4	1e-05	0.0006	4e-05	0.00062
Eig As # 5	1e-05	0.0006	4e-05	0.00062
Eig As # 6	0	0.00031	7e-05	0.00033
Eig As # 7	0	0.00031	7e-05	0.00033
Eig As # 8	1e-05	0.03705	0.01058	0.03604
Eig As # 9	1e-05	0.03705	0.01058	0.03604
Eig As #10	4e-05	0.05722	0.01759	0.05595
Eig As #11	4e-05	0.05722	0.01759	0.05595
Eig As #12	0.0001	0.34544	0.10124	0.33724
Eig As #13	0.0001	0.34544	0.10124	0.33724
Eig As #14	2e-05	0.00027	0.00508	3e-05
Eig As #15	1e-05	0	0.01006	3e-05
Eig As #16	0.00012	0.00996	0.06741	0.01415
Eig As #17	0.00012	0.00996	0.06741	0.01415
Eig As #18	4e-05	0.00542	0.04361	0.00787
Eig As #19	4e-05	0.00542	0.04361	0.00787
Eig As #20	0.00018	0.03173	0.26806	0.04637
Eig As #21	0.00018	0.03173	0.26806	0.04637
Eig As #22	0	0	0	0
Eig As #23	0	0	0	0
Eig As #24	0	0	0	0

STATISTICS

DYNAMIC ORDER	24
# OF EIGS WITH $\text{Re}(\mu) < 0$	22
# OF EIGS WITH $\text{Re}(\mu) > 0$	0
# OF REAL EIGS	8
# OF COMPLEX PAIRS	8
# OF ZERO EIGS	2

8.2 Power Flow Sensitivity Analysis

For the power flow (or qv) sensitivity analysis, three matrices can be used:

1. \mathbf{J}_{LF} , which is obtained from the static equations (12.1) of power flows in transmission lines and transformers, and is generally defined as the *standard* power flow Jacobian matrix.
2. \mathbf{J}_{LFV} , which is the complete Jacobian matrix of the power flow equations of the system.
3. \mathbf{J}_{LFD} , which is computed from the complete matrix \mathbf{A}_C :

$$\mathbf{J}_{LFD} = \mathbf{J}_{LFV} - \mathbf{g}_x \mathbf{f}_x^{-1} \mathbf{f}_y \quad (8.12)$$

and can thus be considered a *dynamic* power flow Jacobian matrix.¹

Observe that in the previous definitions, it has been assumed that the algebraic variables are only bus voltage magnitudes and phases, i.e. $\mathbf{J}_{LFV} = \mathbf{g}_y$. If there are

¹If there are pure integrators in the system, \mathbf{f}_x can be singular. In this case, \mathbf{f}_x is conditioned by adding a small value on the diagonal before computing the inverse.

other algebraic variables, these are removed from the Jacobian matrix as follows:²

$$\mathbf{J}_{LFV} = \mathbf{g}_y^{(m_1, m_1)} - \mathbf{g}_y^{(m_1, m_2)} [\mathbf{g}_y^{(m_2, m_2)}]^{-1} \mathbf{g}_y^{(m_2, m_1)} \quad (8.13)$$

where m_1 is twice the number of the buses of the system and $m_2 = m - m_1$. Observe that the first m_1 rows of \mathbf{g}_y corresponds to the active and reactive power equation gradients. Thus \mathbf{g}_y is as follows:

$$\mathbf{g}_y = \begin{bmatrix} \mathbf{g}_y^{(m_1, m_1)} & \mathbf{g}_y^{(m_1, m_2)} \\ \mathbf{g}_y^{(m_2, m_1)} & \mathbf{g}_y^{(m_2, m_2)} \end{bmatrix} \quad (8.14)$$

The matrix \mathbf{J}_{LFD} can be defined in a similar way as \mathbf{J}_{LFV} in (8.13).

Once the power flow Jacobian matrix has been selected and computed, the eigenvalue analysis is performed on a reduced matrix, as follows. Let us assume that the power flow Jacobian matrix is divided into four sub-matrices:

$$\mathbf{J}_{LF} = \begin{bmatrix} \mathbf{J}_{p\theta} & \mathbf{J}_{pv} \\ \mathbf{J}_{q\theta} & \mathbf{J}_{qv} \end{bmatrix} \quad (8.15)$$

In case of the *standard* Jacobian matrix \mathbf{J}_{LF} , this has also a physical meaning, since it can be obtained by the linearization of the power flow equations with constant power injections:

$$\begin{bmatrix} \Delta p \\ \Delta q \end{bmatrix} = \begin{bmatrix} \mathbf{J}_{p\theta} & \mathbf{J}_{pv} \\ \mathbf{J}_{q\theta} & \mathbf{J}_{qv} \end{bmatrix} \begin{bmatrix} \Delta \theta \\ \Delta v \end{bmatrix} \quad (8.16)$$

Then, the reduced matrix is defined as follows:

$$\tilde{\mathbf{J}}_{LF} = \mathbf{J}_{qv} - \mathbf{J}_{q\theta} \mathbf{J}_{p\theta}^{-1} \mathbf{J}_{pv} \quad (8.17)$$

That can thus be used for a qv sensitivity analysis, if one assumes that $\Delta p = 0$ and that the sub-matrix $\mathbf{J}_{p\theta}$ is non-singular:

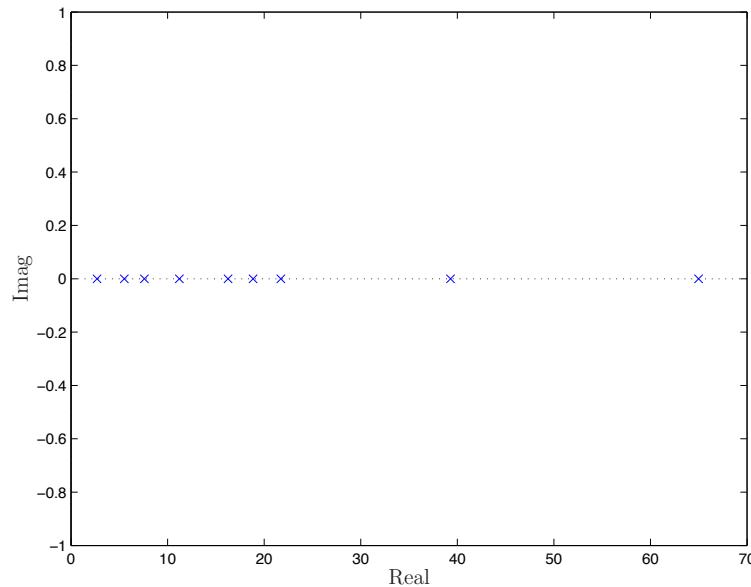
$$\Delta q = \tilde{\mathbf{J}}_{LF} \Delta v \quad (8.18)$$

For \mathbf{J}_{LFV} and \mathbf{J}_{LFD} , the reduced matrix is defined as in (8.17), even though it lacks the rigorous physical meaning of (8.18).

8.2.1 Example

The following Fig. 8.3 and qv sensitivity report depicts the qv sensitivity analysis for the IEEE 14-bus test system and have been generated with the GUI for Small Signal Stability Analysis available in the View menu of the main window. For the static data of the network, refer to Section F.4. The report refers to the \mathbf{J}_{LF} matrix, but since the system has only constant power loads and generators and there are no dynamic components, one has $\mathbf{J}_{LF} = \mathbf{J}_{LFV} = \mathbf{J}_{LFD}$. The report shows five high eigenvalues ($\mu = 999$), which represent the constant voltage buses of the five generators.

²A similar manipulation is done for the matrix \mathbf{f}_y .

Figure 8.3: Eigenvalue Analysis: QV sensitivity.

EIGENVALUE REPORT

P S A T 2.1.3

Author: Federico Milano, (c) 2002-2008
 e-mail: Federico.Milano@uclm.es
 website: <http://www.uclm.es/area/gsee/Web/Federico>

File: ~/Applications/psat2/tests/d_014.mdl
 Date: 05-Nov-2008 23:15:08

EIGENVALUES OF THE COMPLETE POWER JACOBIAN MATRIX

Eigenvalue	Most Associated Bus	Real part	Imaginary Part
Eig Jlfv # 1	Bus 04	63.5305	0
Eig Jlfv # 2	Bus 09	38.3793	0
Eig Jlfv # 3	Bus 07	21.0476	0
Eig Jlfv # 4	Bus 13	18.6295	0
Eig Jlfv # 5	Bus 05	15.6251	0
Eig Jlfv # 6	Bus 14	2.5838	0
Eig Jlfv # 7	Bus 12	5.4231	0
Eig Jlfv # 8	Bus 12	7.4764	0
Eig Jlfv # 9	Bus 11	10.9682	0
Eig Jlfv #10	Bus 01	999	0
Eig Jlfv #11	Bus 02	999	0
Eig Jlfv #12	Bus 03	999	0
Eig Jlfv #13	Bus 06	999	0
Eig Jlfv #14	Bus 08	999	0

PARTICIPATION FACTORS (Euclidean norm)

	Bus 01	Bus 02	Bus 03	Bus 04	Bus 05
Eig Jlfv # 1	0	0	0	0.54241	0.45142
Eig Jlfv # 2	0	0	0	3e-05	0.00067
Eig Jlfv # 3	0	0	0	0.06441	0.13198
Eig Jlfv # 4	0	0	0	0.0003	0.00038
Eig Jlfv # 5	0	0	0	0.28759	0.33383
Eig Jlfv # 6	0	0	0	0.00814	0.00412
Eig Jlfv # 7	0	0	0	0.00248	0.00143
Eig Jlfv # 8	0	0	0	1e-05	0
Eig Jlfv # 9	0	0	0	0.09464	0.07615
Eig Jlfv #10	1	0	0	0	0
Eig Jlfv #11	0	1	0	0	0
Eig Jlfv #12	0	0	1	0	0
Eig Jlfv #13	0	0	0	0	0
Eig Jlfv #14	0	0	0	0	0

PARTICIPATION FACTORS (Euclidean norm)

	Bus 06	Bus 07	Bus 08	Bus 09	Bus 10
Eig Jlfv # 1	0	0.00609	0	8e-05	1e-05
Eig Jlfv # 2	0	0.15558	0	0.6126	0.2147
Eig Jlfv # 3	0	0.50263	0	0.00244	0.23686
Eig Jlfv # 4	0	0.00017	0	0.00011	0.00379
Eig Jlfv # 5	0	0.01049	0	0.0516	0.14517
Eig Jlfv # 6	0	0.07032	0	0.20176	0.24
Eig Jlfv # 7	0	0.01726	0	0.03244	0.12213
Eig Jlfv # 8	0	4e-05	0	4e-05	0.03499
Eig Jlfv # 9	0	0.23743	0	0.09893	0.00236
Eig Jlfv #10	0	0	0	0	0
Eig Jlfv #11	0	0	0	0	0
Eig Jlfv #12	0	0	0	0	0
Eig Jlfv #13	1	0	0	0	0
Eig Jlfv #14	0	0	1	0	0

PARTICIPATION FACTORS (Euclidean norm)

	Bus 11	Bus 12	Bus 13	Bus 14
Eig Jlfv # 1	0	0	0	0
Eig Jlfv # 2	0.00783	0	0.00012	0.00846
Eig Jlfv # 3	0.06126	3e-05	0.00018	0.00023
Eig Jlfv # 4	0.00162	0.18175	0.76625	0.04563
Eig Jlfv # 5	0.16017	0.00212	0.00416	0.00486
Eig Jlfv # 6	0.10757	0.01767	0.03124	0.31918
Eig Jlfv # 7	0.1318	0.30824	0.15754	0.22668
Eig Jlfv # 8	0.10486	0.48061	0.03876	0.34069
Eig Jlfv # 9	0.42488	0.00958	0.00176	0.05426
Eig Jlfv #10	0	0	0	0
Eig Jlfv #11	0	0	0	0
Eig Jlfv #12	0	0	0	0
Eig Jlfv #13	0	0	0	0
Eig Jlfv #14	0	0	0	0

STATISTICS

NUMBER OF BUSES	14
# OF EIGS WITH Re(mu) < 0	0
# OF EIGS WITH Re(mu) > 0	14
# OF REAL EIGS	14
# OF COMPLEX PAIRS	0
# OF ZERO EIGS	0

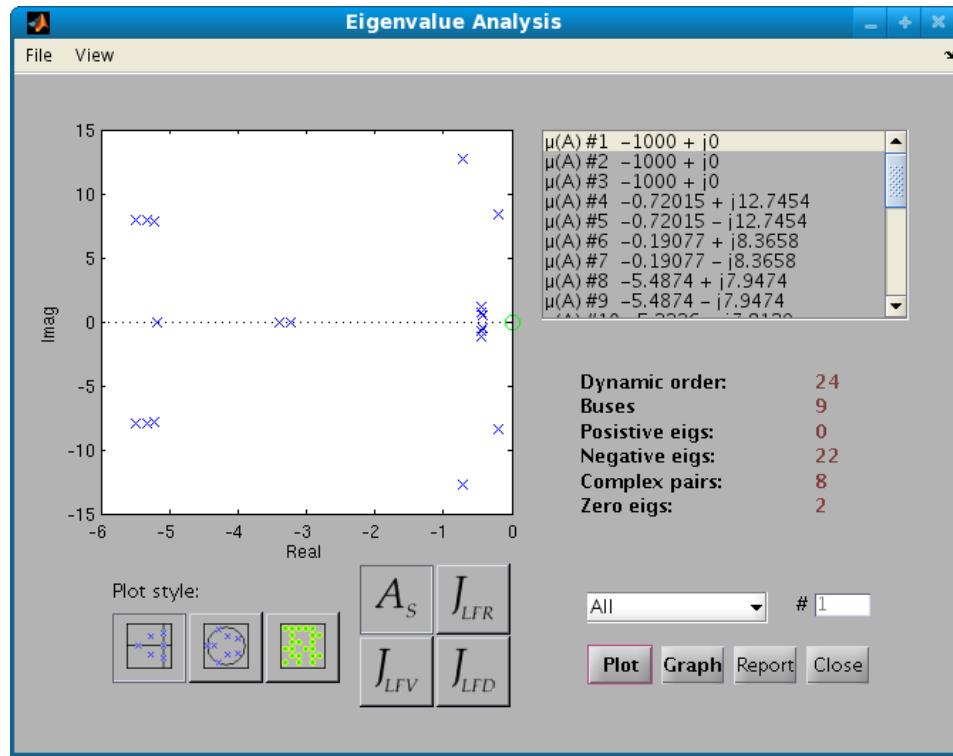


Figure 8.4: GUI for the small signal stability analysis.

8.3 Graphical User Interface

Figure 8.4 depicts the user interface for small signal stability analysis. Several options are available for adjusting the performance and the changing the output of the routine. It is possible to set the output map (S -map, Z -map or participation factor map); the Jacobian matrix (state matrix A_S or one of the power flow Jacobian matrices J_{LF} , J_{LFV} or J_{LFD}); and the number and the kind of eigenvalues to be computed. The “Graph” and the “Report” push-buttons will export the eigenvalue analysis in a new MATLAB figure and write the small signal stability analysis report, respectively.

A complete description of SSSA settings is reported in Appendix A.

Chapter 9

Time Domain Simulation

This chapter describes the time domain integration methods used in PSAT and their settings. A particular class of settings are the *snapshots* that allows computing and storing specific points during the time simulations. This chapter also describes how to include disturbances. Three phase faults and breaker operations are supported by means of specific functions and structures, while a generic disturbance can be created writing an user defined function. Finally, the plotting utilities for time domain simulations are briefly described by means of simple examples.

Several power system software packages make a distinction between power flow (static) data and dynamic ones. On the contrary, in PSAT, static and dynamic data can be defined in the same data file. PSAT makes use of static and/or dynamic data depending on the kind of simulation.

9.1 Integration Methods

Two integration methods are available, i.e. forward Euler's and trapezoidal rule. Both methods are implicit A -stable algorithms and use a complete Jacobian matrix to evaluate the algebraic and state variable directions at each step. These methods are well known and can be found in several books (e.g. [16]).

For a generic time t , and assumed a time step Δt , one has to solve the following problem:

$$\begin{aligned} \mathbf{0} &= \mathbf{f}_n(\mathbf{x}(t + \Delta t), \mathbf{y}(t + \Delta t), \mathbf{f}(t)) \\ \mathbf{0} &= \mathbf{g}(\mathbf{x}(t + \Delta t), \mathbf{y}(t + \Delta t)) \end{aligned} \tag{9.1}$$

where \mathbf{f} and \mathbf{g} are the differential and algebraic equations and \mathbf{f}_n is a function that depends on the integration method. Equations (9.1) are nonlinear and their solution is obtained by means of a Newton-Raphson's method which in turn consists of computing iteratively the increment $\Delta \mathbf{x}^i$ and $\Delta \mathbf{y}^i$ of the state and algebraic

variables and updating the actual variables:

$$\begin{aligned}\begin{bmatrix} \Delta \mathbf{x}^i \\ \Delta \mathbf{y}^i \end{bmatrix} &= -[\mathbf{A}_n^i]^{-1} \begin{bmatrix} \mathbf{f}_n^i \\ \mathbf{g}^i \end{bmatrix} \\ \begin{bmatrix} \mathbf{x}^{i+1} \\ \mathbf{y}^{i+1} \end{bmatrix} &= \begin{bmatrix} \mathbf{x}^i \\ \mathbf{y}^i \end{bmatrix} + \begin{bmatrix} \Delta \mathbf{x}^i \\ \Delta \mathbf{y}^i \end{bmatrix}\end{aligned}\quad (9.2)$$

where \mathbf{A}_n^i is a matrix depending on the algebraic and state Jacobian matrices of the system. The loop stops if the variable increment is below a certain fixed tolerance ϵ_0 or if the maximum number of iteration is reached. In the latter case the time step Δt is reduced and the Newton-Raphson's method repeated again. Figure 9.1 depicts the block diagram of the time domain integration. For sake of completeness, the following sections report the expressions of \mathbf{A}_n^i and \mathbf{f}_n^i for each method.

9.1.1 Forward Euler's Method

The forward Euler's method is a first order integration method. It is generally faster but less accurate than the trapezoidal method. At a generic iteration i , \mathbf{A}_n^i and \mathbf{f}_n^i are as follows:

$$\begin{aligned}\mathbf{A}_n^i &= \begin{bmatrix} \mathbf{I}_n - \Delta t \mathbf{f}_x^i & -\Delta t \mathbf{f}_y^i \\ \mathbf{g}_x^i & \mathbf{g}_y^i \end{bmatrix} \\ \mathbf{f}_n^i &= \mathbf{x}^i - \mathbf{x}(t) - \Delta t \mathbf{f}^i\end{aligned}\quad (9.3)$$

where \mathbf{I}_n is the identity matrix of the same dimension of the dynamic order of the DAE system.

9.1.2 Trapezoidal Method

The trapezoidal method is the workhorse solver for electro-mechanical DAE, and is widely used, in a variety of flavors, in most commercial and non-commercial power system software packages. The version implemented in PSAT is probably the simplest one, but proved to be very robust and reliable for several test cases. At a generic iteration i , \mathbf{A}_n^i and \mathbf{f}_n^i are as follows:

$$\begin{aligned}\mathbf{A}_n^i &= \begin{bmatrix} \mathbf{I}_n - 0.5 \Delta t \mathbf{f}_x^i & -0.5 \Delta t \mathbf{f}_y^i \\ \mathbf{g}_x^i & \mathbf{g}_y^i \end{bmatrix} \\ \mathbf{f}_n^i &= \mathbf{x}^i - \mathbf{x}(t) - 0.5 \Delta t (\mathbf{f}^i + \mathbf{f}(t))\end{aligned}\quad (9.4)$$

where the notation is the same as in (9.3).

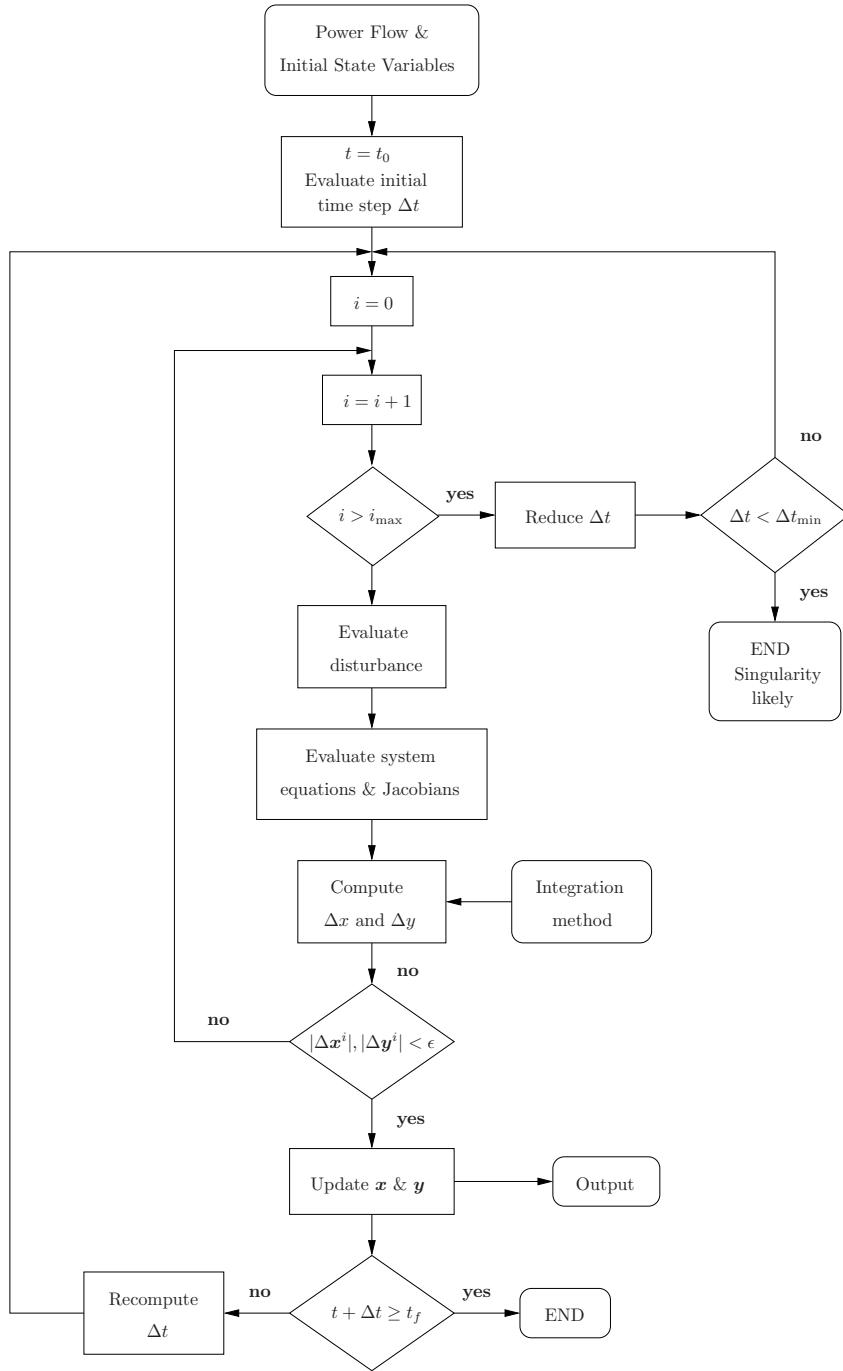


Figure 9.1: Time domain integration block diagram.

9.2 Settings

General settings for time domain simulations, i.e. the initial¹ and final times, convergence tolerance, and maximum number of iterations of the Newton-Raphson's method for each time step can be set in the main window. Other parameters can be customized in the GUI for general settings (menu *Edit/General Settings* or shortcut <Ctrl-k> in the main window), which is depicted in Fig. 9.2. The following options are available for time domain simulations:

Fixed Time Step (Settings.fixt): one can enable the use of a fixed time step.

This can be useful for “critical” simulations were the automatic time step flaws. If the option of the fixed time step is disabled, PSAT will computes a reasonable initial time step based on the eigenvalues of the system at the initial time.² Observe that this procedure can be time-consuming for systems with an high number of state variables.

Time Step [s] (Settings.tstep): the value of the time step in seconds. The default value is 0.001 s.

Integration Method (Settings.method): one can choose in between *Trapezoidal Rule* (default) and *Forward Euler*. Both methods are implicit and A-stable. The trapezoidal rule is the workhorse of time domain simulations of power electric systems.

Stop TDs at Max Delta (Settings.checkdelta): this option makes possible to stop the time domain simulation when the maximum machine angle difference is greater than a given $\Delta\delta^{\max}$.³

Max. Delta Diff. [deg] (Settings.deltadelta): the maximum machine angle difference in degree for which the time domain simulation will be stopped. The default value is 180°.

Use Center of Inertia (COI) (Settings.coi): enforce the use of the Center of Inertia for synchronous machines. See Section 17.1.9 for details.

Convert PQ bus to Z (Settings.pq2z): if this option is enabled, PQ buses are converted to constant impedances right before beginning the time domain simulation. Enforcing this option will help convergence if there are fault occurrences and/or breaker interventions during the time domain simulation. Refer to Sections 12.6 for details on the conversion of PQ loads to constant impedances.

Plot during Simulation (Settings.plot): to enforce this option will generate a graphic of selected variables (see item *Plotting Variables*) during time domain

¹Although the initial time could be assigned any value, it is recommended to use $t_0 = 0$ as other values have not been tested and could cause problems with time dependent devices. In any case it must be $t_0 \geq 0$.

²This operation is performed in the function `fm_tstep`.

³This option has been added by Laurent Lenoir, École Polytechnique de Montréal.

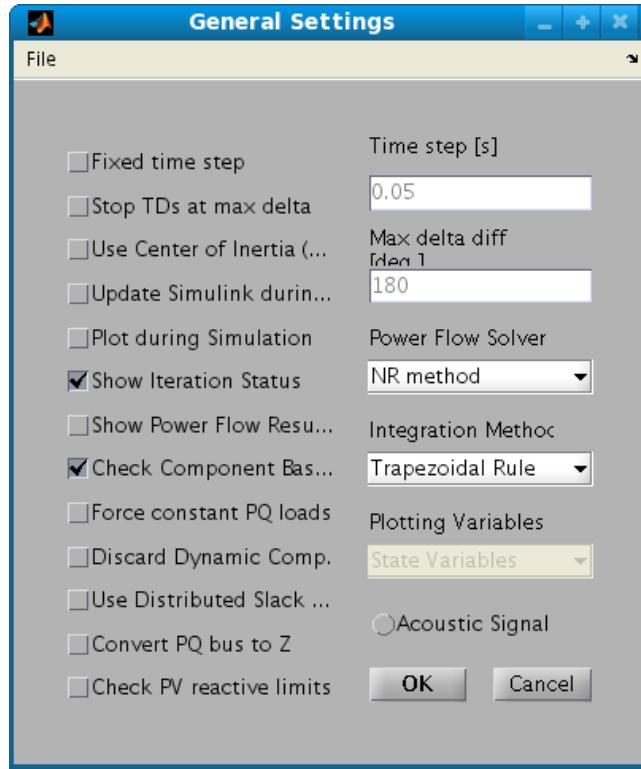


Figure 9.2: GUI for general settings.

simulations. This is typically a time consuming operation and is disabled by default.

Plotting Variables (Settings.plottype): this pop-up menu allows selecting plotting variables that will be displayed during time domain simulations. Observe that these are not the variables that will be stored in the output. See Section 9.3 for more details on output variable selection.

Update Simulink during TD (Settings.simtd): this option allows displaying and updating voltages in SIMULINK models during time domain simulations. This option only works if the data are loaded from a SIMULINK model. This option will not work on GNU OCTAVE and for the command line usage of PSAT.

Settings for time domain simulations are stored in the structure **Settings**, which contains also general settings and parameters for power flow computations. This structure is fully described in Appendix A.

9.3 Output Variable Selection

During time domain simulations (and also for CPF analysis), PSAT stores output variables in the structure `Varout`. Further details on this structure are given in the Appendix A.

By default, PSAT stores only state variables, bus voltage magnitudes and bus voltage angles. This behavior can be changed by means of the GUI for plot variable selection (see Fig. 9.3). By this GUI, the user can select any of the following variables:

1. State variables;
2. Bus voltage angles and magnitudes;
3. Other algebraic variables, such as generator mechanical powers and field voltages, AVR reference voltages, etc.
4. Active and reactive power injections at buses;
5. Currents and active, reactive and apparent power flows in transmission lines.

The GUI allows selecting variables one by one, by area, by region or by pre-defined sets. When using the command line version of PSAT, the selection has to be done manually by assigning a vector of indexes to `Varout.idx`. The variable indexes are as follows:

- from (1) to (`DAE.n`) state variables.
- from (`DAE.n + 1`) to (`DAE.n + Bus.n`) bus voltage angles.
- from (`DAE.n + Bus.n + 1`) to (`DAE.n + 2*Bus.n`) bus voltage magnitudes.
- from (`DAE.n + 2*Bus.n + 1`) to (`DAE.n + DAE.m`) all other algebraic variables, including generator field voltages and mechanical powers, AVR reference voltages, OXL field currents, etc.
- from (`DAE.n + DAE.m + 1`) to (`DAE.n + DAE.m + Bus.n`) active power injections at buses.
- from (`DAE.n + DAE.m + Bus.n + 1`) to (`DAE.n + DAE.m + 2*Bus.n`) reactive power injections at buses.
- from (`DAE.n + DAE.m + 2*Bus.n + 1`) to (`DAE.n + DAE.m + 2*Bus.n + nL`) active power flows $i-j$.
- from (`DAE.n + DAE.m + 2*Bus.n + nL + 1`) to (`DAE.n + DAE.m + 2*Bus.n + 2*nL`) active power flows $j-i$.
- from (`DAE.n + DAE.m + 2*Bus.n + 2*nL + 1`) to (`DAE.n + DAE.m + 2*Bus.n + 3*nL`) reactive power flows $i-j$.

- from $(DAE.n + DAE.m + 2*Bus.n + 3*nL + 1)$ to $(DAE.n + DAE.m + 4*nL)$ reactive power flows $j-i$.
- from $(DAE.n + DAE.m + 2*Bus.n + 4*nL + 1)$ to $(DAE.n + DAE.m + 5*Bus.n + 3*nL)$ current flows $i-j$.
- from $(DAE.n + DAE.m + 2*Bus.n + 5*nL + 1)$ to $(DAE.n + DAE.m + 6*nL)$ current flows $j-i$.
- from $(DAE.n + DAE.m + 2*Bus.n + 6*nL + 1)$ to $(DAE.n + DAE.m + 7*Bus.n + 3*nL)$ apparent power flows $i-j$.
- from $(DAE.n + DAE.m + 2*Bus.n + 7*nL + 1)$ to $(DAE.n + DAE.m + 8*nL)$ apparent power flows $j-i$.

where $nL = Line.n + Ltc.n + Phs.n + Hvdc.n + Lines.n$.

For example, if one wants to plot only the voltage magnitude of the third bus and the active power injections at the fourth bus, the index assignment will be as follows:

```
>> Varout.idx = [DAE.n + Bus.n + 3, DAE.n + DAE.m + 4];
```

The assignment must be done **after** running the power flow analysis and **before** running the time domain simulation. Any custom variable selection is lost after running the power flow analysis unless it is saved through the GUI for variable selection.⁴ If this assignment is done in a function, remember to declare as **global** all needed structures.

9.4 Snapshots

In PSAT, snapshots are used to store specific points of time domain simulations.⁵ Once the simulation is completed, the user can browse existing snapshots and display the values of state and algebraic variables in that points.

Figure 9.4 depicts the graphical user interface for setting the *snapshots* (menu *Tools/Snapshots* or shortcut <Ctrl-n> in the main window). This GUI is displayed only after solving the power flow and always contains a snapshot called *Power Flow Result*. The GUI allows to set any number of snapshots at desired times. When running the time domain simulation, the integration routine will compute a point for each time defined in the snapshots and store the system variables in the structure *Snapshot*. This option can be useful for being sure that the time domain simulation will compute a point for a determined time at which a disturbance is applied or for fitting the time steps in delimited regions of the simulation time interval.

The GUI allows also to set the currently selected snapshot as the “initial time” for the next time domain simulation. This option allows starting the time domain

⁴The custom variable list is saved as a vector in the data file. If the data file proceeds from a Simulink model, a block containing the list information is added to the model. Be aware that any changes in the model can make the variable list obsolete.

⁵Snapshots are also used during continuation power flow analysis for storing bifurcation points.

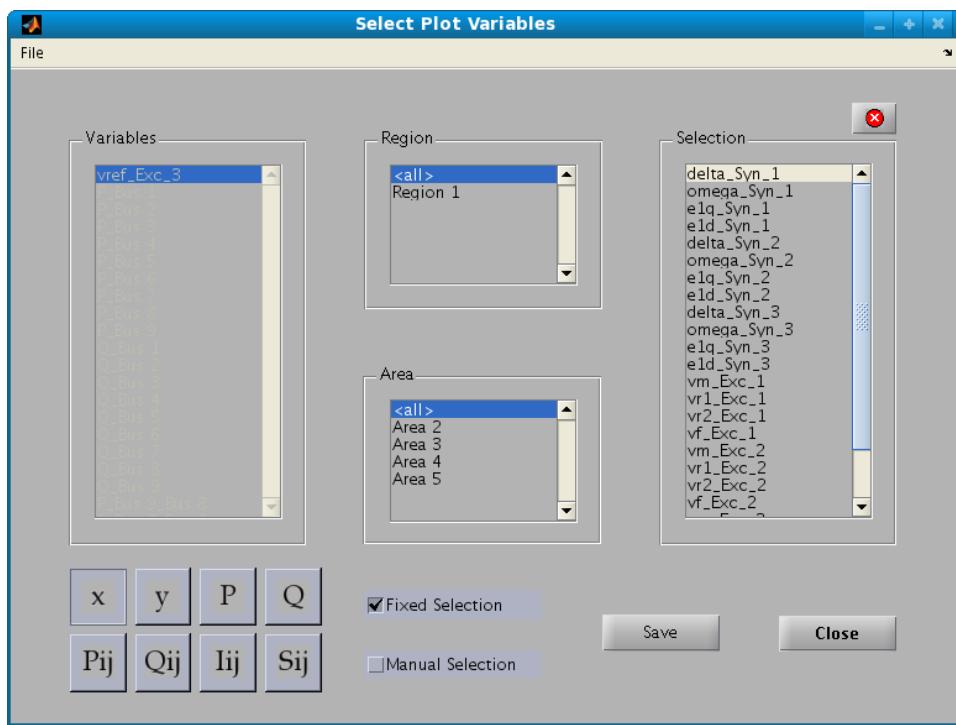


Figure 9.3: GUI for plot variable selection.

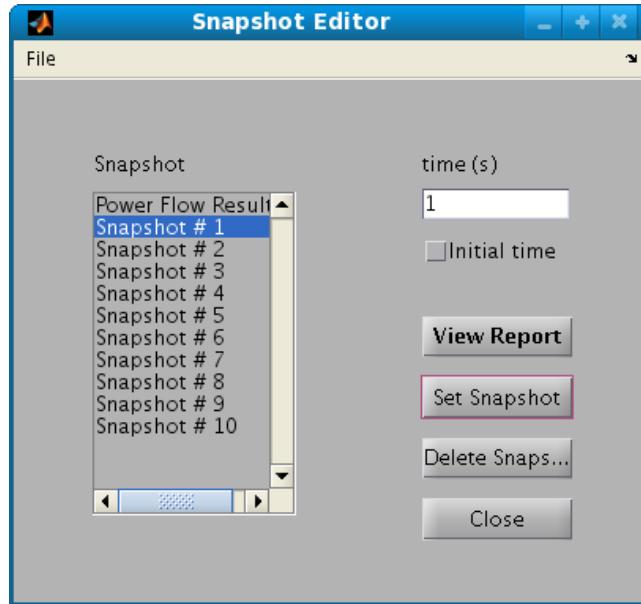


Figure 9.4: GUI for snapshot settings.

simulation without actually recomputing the power flow. A snapshot defined as initial time can be visualized by the GUI for power flow reports (menu *View/Static Report* or shortcut $<\text{Ctr-v}>$ in the main window). A sequence of snapshots can be also visualized in the GUI for plotting variables (menu *View/Plotting Utilities* or shortcut $<\text{Ctr-w}>$ in the main window).

9.5 Disturbances

Disturbances are fully supported in PSAT, although they might require some programming skill. The most common perturbations for transient stability analysis, i.e. faults and breaker interventions, are handled by means of special structures and functions, whereas a generic perturbation requires the definition of an user defined function. Fault and breaker models are described in Chapter 14. Observe that one does not need to load a perturbation file/function when using faults and/or breakers models.⁶

⁶One could run a time domain simulation just after the power flow analysis for any system. It does not matter if there is no perturbation file and no fault and breaker components loaded. It does not even matter if there is no dynamic component in the actual network. Of course in the latter cases, the time domain simulation will provide constant values for all variables. Observe that running a trivial time domain simulation could be useful to test the initialization of dynamic components and regulators. For the same reason, it is better set disturbance actions some time after the initial simulation time.

Generic disturbances are supported by means of user defined functions.⁷ Perturbation files are loaded in the main window as described in Section 2. Only one (or none) perturbation file at a time can be loaded. Their structure should be as follows:

```
function pert(t)

global global_variable_name1 global_variable_name2 ...

if ... % criterion

    % actions

elseif ... % criterion

    % actions

else

    % actions

end
```

The function must accept as an input the current simulation time (scalar value) and may include any global structure of the system for taking the desired actions.⁸ Observe that the time domain integration calls the disturbance file at each iteration, thus it may be convenient to reduce the number of operations within the disturbance function. In order to force the integration routine to evaluate a particular point, define the desired time in the `Snapshot` structure.

9.6 Examples

Figure 9.5 depicts the graphical user interface for plotting time domain simulation results. As an example the figure depicts the speeds ω for the three generators of the WSCC 9-bus test system. Generators are represented by means of a fourth order model with automatic voltage regulation (IEEE type I) [111]. The data for this system are reported in Appendix F.3. After solving the power flow, the rotor speed of one generator is set to 0.95 p.u. as follows:

⁷Step perturbations can also be obtained by changing parameter or variable values at the MATLAB prompt after solving the power flow computation and before starting the time domain simulation.

⁸Observe that it may be necessary to call other functions. For example, after modifying a transmission line impedance, one has to call `Line = build.y(Line);` in order to update the admittance matrix.

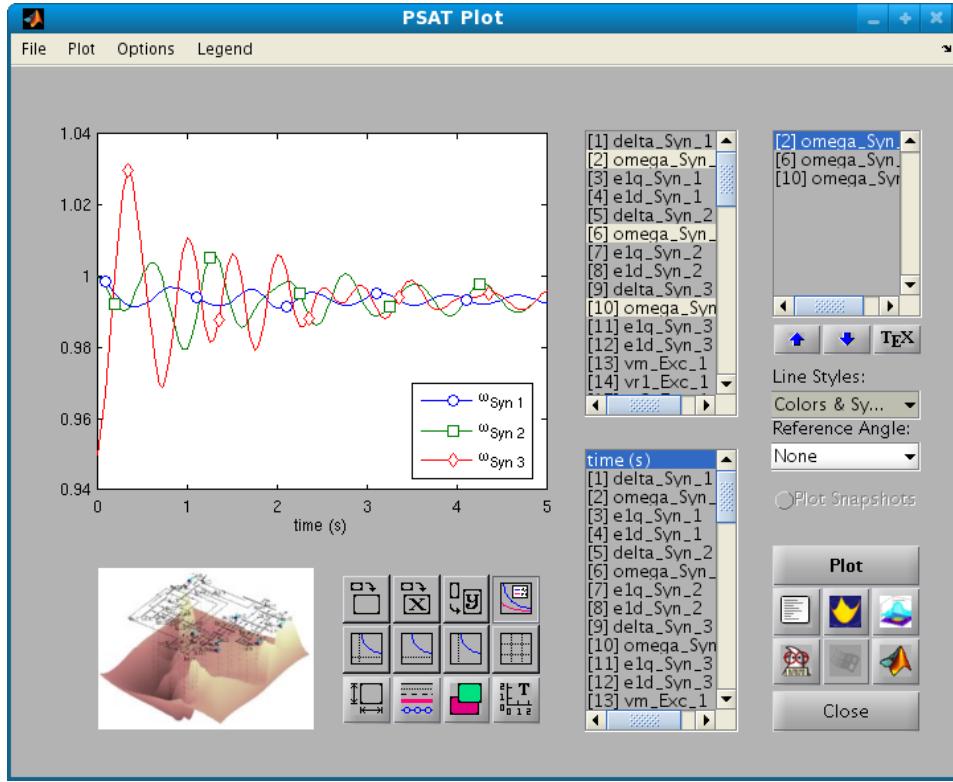


Figure 9.5: GUI for plotting time domain simulations. In this example, the speeds refer to the 9-bus test with IV order generator models and AVR type II. The disturbance is obtained by varying the speed of generator at bus 3 at the MATLAB prompt ($\omega_3(t_0) = 0.95$ p.u.).

```
>> DAE.x(Syn.omega(2)) = 0.95;
```

then the time domain simulation is performed. The GUI allows a variety of settings, such as choosing the variables to plot, setting in detail the graphical appearance, adding and modifying a legend of the plotted variables and saving the graph to a color .eps file, which is placed in the folder of the current data file and automatically named with a progressive number (from 00 to 99).

Figure 9.6 depicts generator rotor speeds and some relevant bus voltages, respectively, for the 9-bus test system with simplified synchronous machine models (δ, ω model), as described in the examples 2.6-2.7, pp. 41-46, “Power System Control and Stability”, by P. M. Anderson and A. A. Fouad [6]. A three phase fault occurs at $t = 1$ s, at bus 7. The fault is then cleared by opening the line 4-7 at $t = 1.083$ s. Finally the line 4-7 is re-closed at $t = 4$ s. The data for this system are reported as well in Appendix F.3.

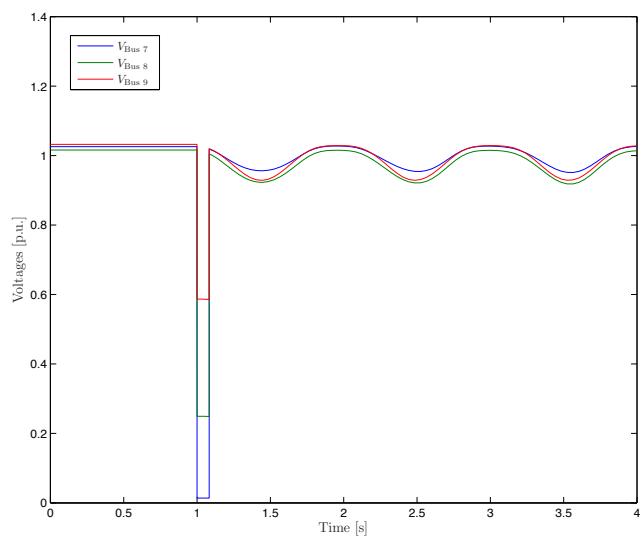
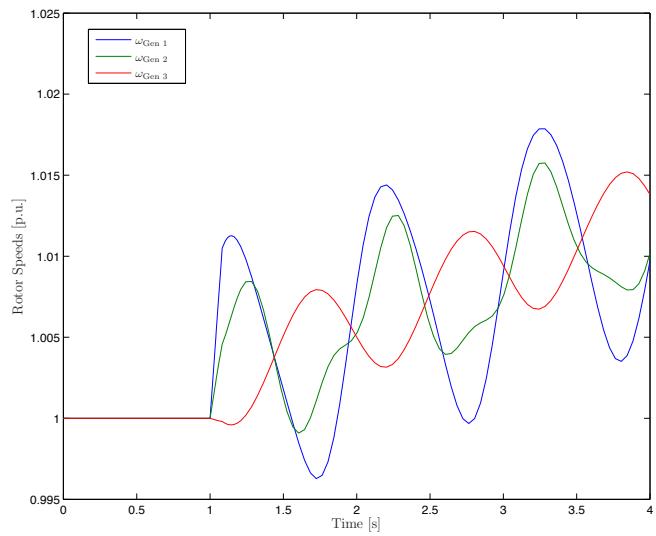


Figure 9.6: Generator speeds and bus voltages for the 9-bus test system with II order generator models and a fault applied at bus 7.

Chapter 10

PMU Placement

This chapter describes seven methods for Phasor Measurement Unit (PMU) placement with the aim of linear static state estimation of power system networks. These methods are depth first, graph theoretic procedures and bisecting search-simulated annealing which were proposed in [11], as well as recursive and single shot N security and recursive and single shot $N - 1$ security algorithms which were proposed in [43]. A description of the PMU placement GUI and an example of report file for the 14-bus test system are reported at the end of this chapter.

10.1 Linear Static State Estimation

This section briefly describes basic concepts of power system static state estimation based on what was proposed in [112] and [39].

The static state estimation problem is generally formulated as a non-linear set of equations, as follows:

$$\mathbf{z} = \mathbf{h}(\mathbf{x}) + \boldsymbol{\epsilon} \quad (10.1)$$

where:

\mathbf{z} ($\mathbf{z} \in \mathbb{R}^m$): measurement vector;

\mathbf{x} ($\mathbf{x} \in \mathbb{R}^n$): state vector;

$\boldsymbol{\epsilon}$ ($\boldsymbol{\epsilon} \in \mathbb{R}^m$): measurement errors vector;

\mathbf{h} ($\mathbf{h} : \mathbb{R}^n \rightarrow \mathbb{R}^m$): vector of the relationships between states and measurements;

Equation (10.1) is typically solved by means of a Newton-Raphson's method [112, 3, 94]. Using devices able to provide voltage and current phasors, such as PMUs, yields a linear relationship between state variables and measurements variables, as follows:

$$\mathbf{z} = \mathbf{H}\mathbf{x} + \boldsymbol{\epsilon} \quad (10.2)$$

where \mathbf{H} ($\mathbf{H} \in \mathbb{R}^{m \times n}$) is the *state* matrix of the system. Typically $m > n$, and the solution of (10.2) is obtained by a least mean square technique [126].

By splitting the vector \mathbf{z} into the $m_V \times 1$ voltage and $m_I \times 1$ current sub-vectors, \mathbf{z}_V and \mathbf{z}_I , and the vector \mathbf{x} into the $n_M \times 1$ and $n_C \times 1$ non-measured sub-vectors, \mathbf{v}_M and \mathbf{v}_C , relationship (10.2) becomes

$$\begin{bmatrix} \mathbf{z}_V \\ \mathbf{z}_I \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{Y}_{IM} & \mathbf{Y}_{IC} \end{bmatrix} \begin{bmatrix} \mathbf{v}_M \\ \mathbf{v}_C \end{bmatrix} + \begin{bmatrix} \mathbf{\epsilon}_V \\ \mathbf{\epsilon}_C \end{bmatrix} \quad (10.3)$$

where \mathbf{I} is the identity matrix, and \mathbf{Y}_{IM} , \mathbf{Y}_{IC} are sub-matrices whose elements are series and shunt admittances of the network branches. Neglecting shunts, the matrix \mathbf{H} is as follows:

$$\mathbf{H} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{M}_{IB}\mathbf{Y}_{BB}\mathbf{A}_{MB}^T & \mathbf{M}_{IB}\mathbf{Y}_{BB}\mathbf{A}_{CB}^T \end{bmatrix} \quad (10.4)$$

where \mathbf{M}_{IB} is the $m_I \times b$ measurement-to-branch incidence matrix associated with the current phasor measurements, \mathbf{Y}_{BB} is the $b \times b$ diagonal matrix of the branch admittances, and \mathbf{A}_{MB} and \mathbf{A}_{CB} are the $n_M \times b$ and $n_C \times b$ calculated node-to-branch incidence sub-matrices, respectively [11, 38].

10.2 PMU Placement Rules

The following PMU placement rules were proposed in [11]:

Rule 1: Assign one voltage measurement to a bus where a PMU has been placed, including one current measurement to each branch connected to the bus itself (Fig. 10.1.a).

Rule 2: Assign one voltage pseudo-measurement to each node reached by another equipped with a PMU.

Rule 3: Assign one current pseudo-measurement to each branch connecting two buses where voltages are known (Fig. 10.1.b). This allows interconnecting observed zones.

Rule 4: Assign one current pseudo-measurement to each branch where current can be indirectly calculated by the Kirchhoff current law (Fig. 10.1.c). This rule applies when the current balance at one node is known, i.e. if the node has no power injections (if $N - 1$ currents incident to the node are known, the last current can be computed by difference).

10.3 Algorithms

10.3.1 Depth First

This method uses only Rules from 1 to 3 (pure transit nodes are not considered). The first PMU is placed at the bus with the largest number of connected branches.

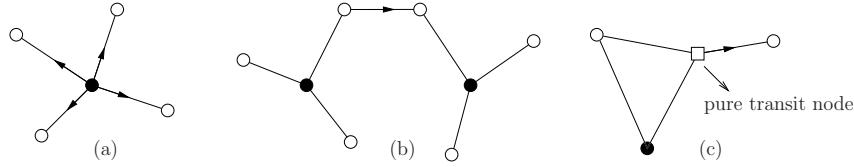


Figure 10.1: PMU placement rules.

If there is more than one bus with this characteristic, one is randomly chosen. Following PMUs are placed with the same criterion, until the complete network observability is obtained, as depicted in Fig. 10.2.¹

10.3.2 Graph Theoretic Procedure

This method was originally proposed in [11] and is similar to the depth first algorithm, except for taking into account pure transit nodes (Rule 4).

10.3.3 Bisection Search Method

Figures 10.3 and 10.4 depict the flowchart of the bisection search method and the pseudo-code of the simulated annealing procedure. Refer to [11] for the complete description of this method.

10.3.4 Recursive Security N Algorithm

This method is a modified depth first approach. The procedure can be subdivided into three main steps:

- a) **Generation of N minimum spanning trees:** Fig. 10.5 depicts the flow chart of the minimum spanning tree generation algorithm. The algorithm is performed N times (N being the number of buses), using as starting bus each bus of the network.
- b) **Search of alternative patterns:** The PMU sets obtained with the step (a) are reprocessed as follows: one at a time, each PMU of each set is replaced at the buses connected with the node where a PMU was originally set, as depicted in Fig. 10.6. PMU placements which lead to a complete observability are retained.
- c) **Reducing PMU number in case of pure transit nodes:** In this step it is verified if the network remains observable taking out one PMU at a time from each set, as depicted in Fig. 10.7. If the network does not present pure transit nodes, the procedure ends at step (b).

¹The depth first and the graph theoretic procedures do not ensure a minimum PMU placement.

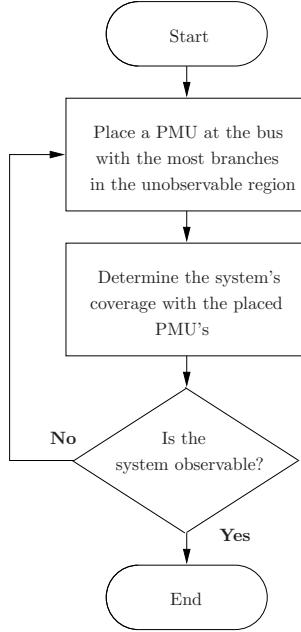


Figure 10.2: Flowchart of the Graph Theoretic Procedure.

The placement sets which present the minimum number of PMUs are finally selected.

10.3.5 Single Shot Security N Algorithm

This method was proposed in [43]. The algorithm is based only on topological rules, and determines a single spanning tree, as illustrated in Fig. 10.8.

10.3.6 Recursive and Single-Shot Security $N - 1$ Algorithms

The rules for minimal PMU placement assume a fixed network topology and a complete reliability of measurement devices. Simple criteria which yield a complete observability in case of line outages ($N - 1$ security) are proposed in [43] and are based on the following definition: A bus is said to be observable if at least one of the two following conditions applies:

Rule 1: a PMU is placed at the node;

Rule 2: the node is connected at least to two nodes equipped with a PMU.

Rule 2 is ignored if the bus is connected to single-end line. Figures 10.9 and 10.10 depict the algorithms for obtaining the $N - 1$ security placement proposed in [43]. The first method is a slightly different version of the recursive technique described

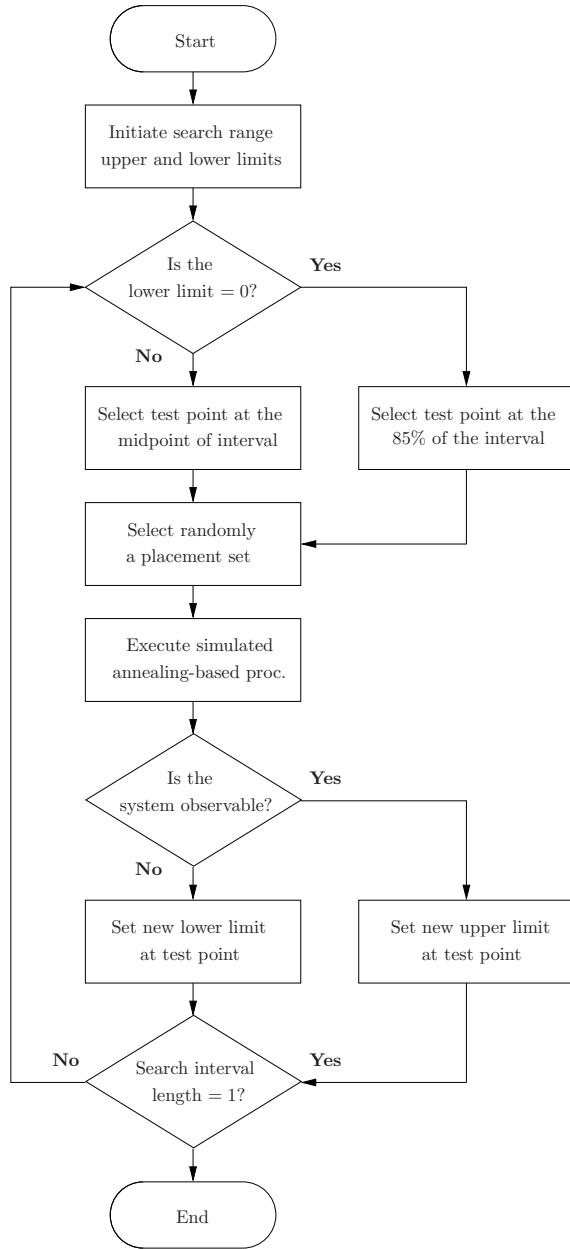


Figure 10.3: Flowchart of the Bisecting Search.

```

begin
    evaluate coverage of PMU placement set  $S$ 
     $E := N - \text{number of buses in the observed region}$ 
     $T := 15$ 
     $M := \min\{0.002\binom{N}{\nu_{\text{test}}}, M_{\max}\}$ 
    for  $i := 1$  to 40 do
        for  $j := 1$  to  $M$  do
            randomly select a PMU
            save the bus location of the selected PMU
            randomly select a non-PMU bus
            evaluate coverage of the modified placement set
             $E_{\text{new}} := N - \text{number of buses in the observed region}$ 
            if  $E_{\text{new}} = 0$  then
                return with ‘system observable’
                and the modified placement set
            fi
             $\Delta E := E_{\text{new}} - E$ 
            if  $\Delta E > 0$  then
                generate a random accept/reject value
                with a probability  $\exp(-\Delta ET)$ 
                if reject then
                    return selected PMU to
                    previous bus location
                fi
            fi
        od
         $T := 0.879T$ 
    od
    return with ‘system not observable’
end

```

Figure 10.4: Pseudo-code of the simulated Annealing Algorithm.

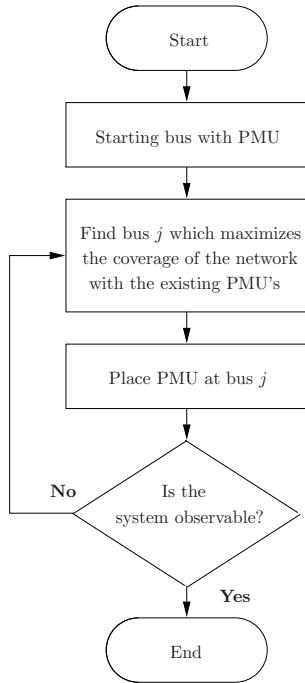


Figure 10.5: Recursive N Security Method.

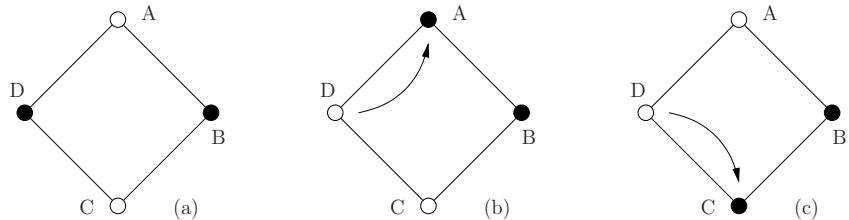


Figure 10.6: Search of alternative placement sets.

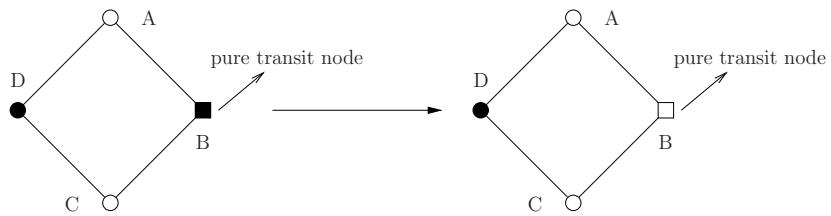


Figure 10.7: Pure transit node filtering.

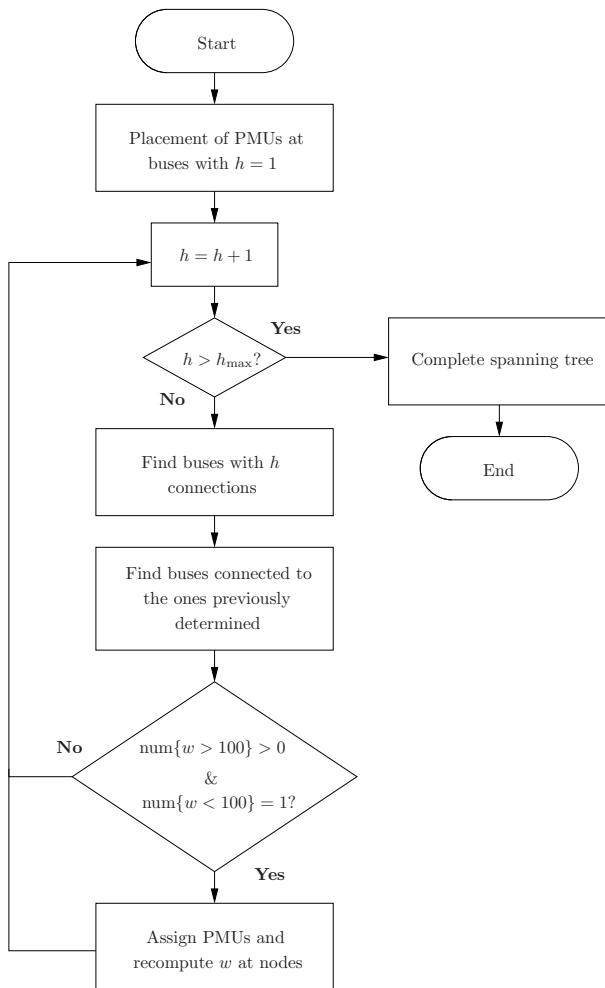


Figure 10.8: Single-Shot N Security Method.

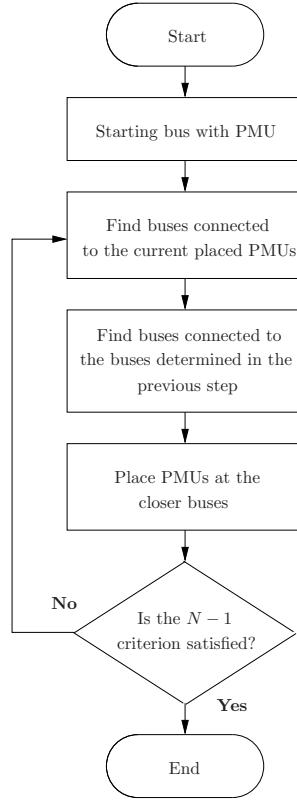


Figure 10.9: Recursive $N - 1$ Security Method.

in Section 10.3.4, whereas the second method is a variant of the algorithm described in Section 10.3.5.

10.4 PMU Placement GUI and Settings

Figure 10.11 depicts the GUI for PMU placement, which allows to select the PMU placement method and enable to write result in a report file. The list-boxes report the voltages obtained with the power flow and the ones determined with the linear static state estimation based on the current PMU set, as well as the position of the PMUs.

All PMU settings and results are set in the structure PMU. Refer to Appendix A for details.

10.4.1 Example

An example of report text file of PMU placement is as follows:

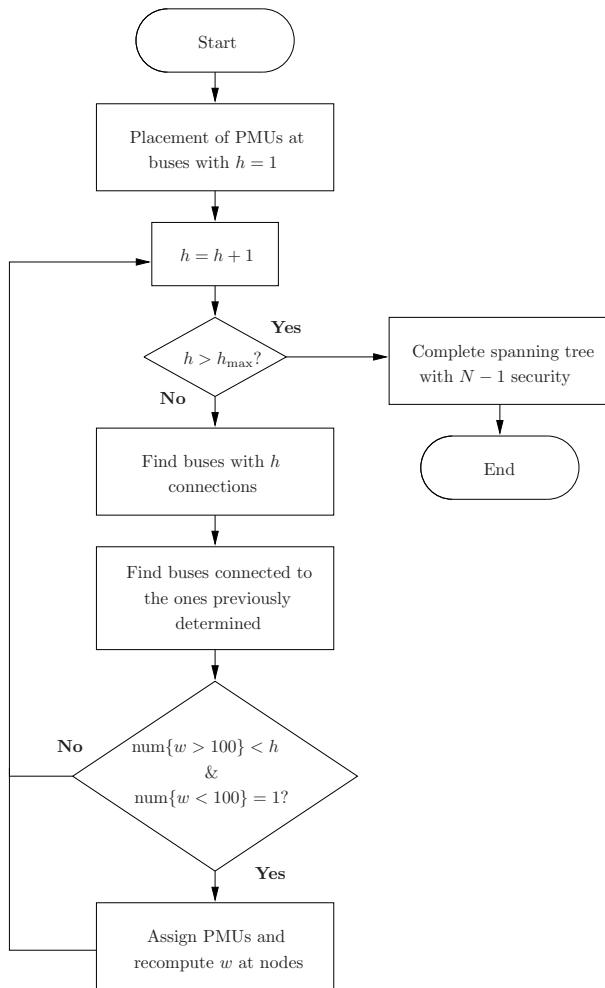


Figure 10.10: Single Shot $N - 1$ Security Method.

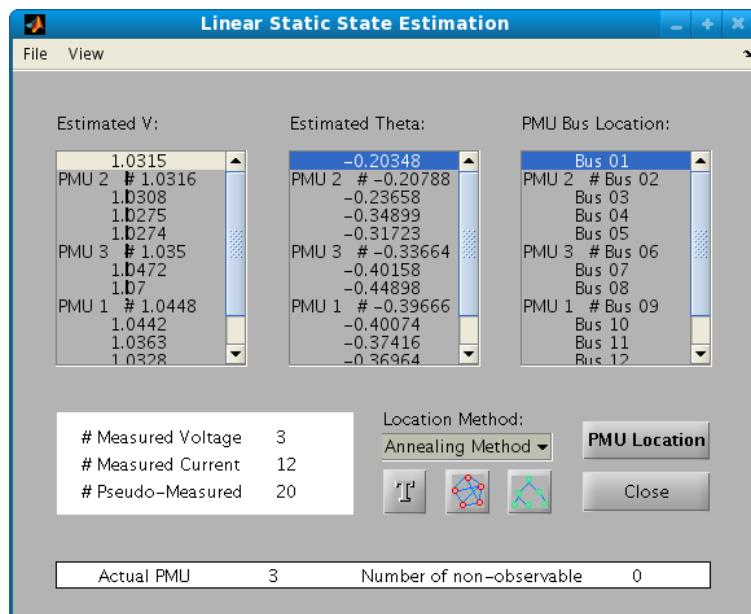


Figure 10.11: GUI for the PMU placement methods.

PMU PLACEMENT REPORT

P S A T 2.1.3

Author: Federico Milano, (c) 2002-2008
 e-mail: Federico.Milano@uclm.es
 website: <http://www.uclm.es/area/gsee/Web/Federico>

File: ~/Applications/psat2/tests/d_014.mdl
 Date: 05-Nov-2008 23:39:35

Placement Method: Annealing Method
 Elapsed Time: 0h 0m 0.82978s

STATISTICS

Buses	14
Lines	20
PMUs	3
PMU Sets	1
Meas. Currents	12
Pseudo-Meas. Currents	20

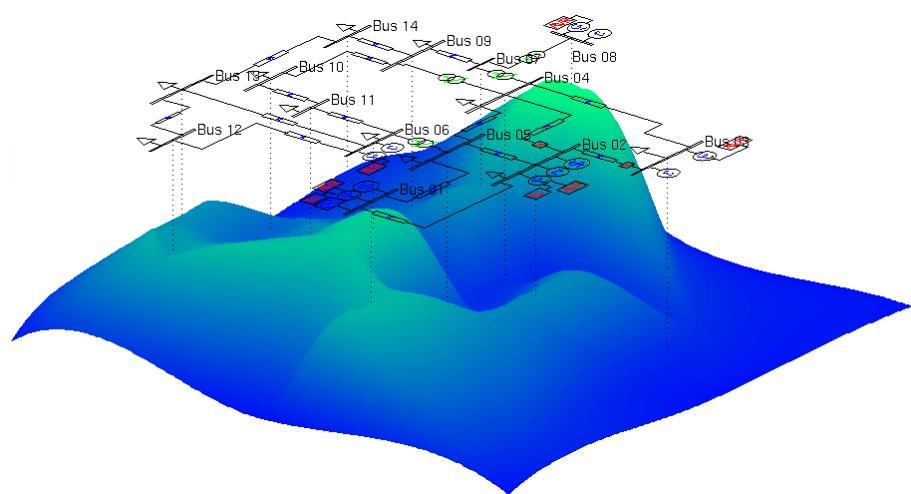
PMU PLACEMENT

Bus Name Set 1

Bus 01	0
Bus 02	1
Bus 03	0
Bus 04	0
Bus 05	0
Bus 06	1
Bus 07	0
Bus 08	0
Bus 09	1
Bus 10	0
Bus 11	0
Bus 12	0
Bus 13	0
Bus 14	0

Part III

Models



Chapter 11

Generalities

In PSAT, device models are defined by a class. Even though in MATLAB , classes are not as powerful as in other modern computer languages, the advantages of using classes are huge. Classes ease maintenance of the code, facilitate modularity and permits to reuse well assessed code. All these features can be hardly obtained without the use of classes.

A class is basically a complex type that contains a variety pf attributes, namely, properties (data) and methods (functions). In PSAT, some properties and methods are common to all devices, as described in the following two sections. Unless clearly stated, a device has at least the common properties described below. Other specific properties are given in the description of each device.

In this documentation only public properties are discussed, i.e. properties that can be accessed and modified from outside the class. Private properties and specific methods are not described. The user is invited to open the code and study the functions that compose each device.¹ Fortunately, the class modularity allows quickly mastering the code.

11.1 General Device Model

The general model for each device is as follows:

$$\begin{aligned}\dot{\mathbf{x}}_i &= \mathbf{f}_i(\mathbf{x}_i, \mathbf{y}_i, \mathbf{x}_e, \mathbf{y}_e, \mathbf{p}_i, \mathbf{u}_i) \\ \mathbf{0} &= \mathbf{g}_i(\mathbf{x}_i, \mathbf{y}_i, \mathbf{x}_e, \mathbf{y}_e, \mathbf{p}_i, \mathbf{u}_i)\end{aligned}\tag{11.1}$$

where the sub-index i indicates specific device internal variables and the sub-index e indicates external variables from other devices; \mathbf{f} are the differential equations, \mathbf{g} are the algebraic equations, \mathbf{x} are the state variables; \mathbf{y} are the algebraic variables; \mathbf{p} are device parameters assigned in the data file; and \mathbf{u} are controllable variables (such as reference signals in control loops).

¹The complete list of classes is given in Appendix A.

A static device is defined only by algebraic equations, while a dynamic one typically by both differential and algebraic equations. Algebraic equations of devices that inject powers into a node include at least two equations, for the active power and for the reactive one, respectively, as follows:

$$\begin{aligned} 0 &= p(\mathbf{x}_e, \mathbf{y}_e) - p_D(\mathbf{x}_i, \mathbf{y}_i, \mathbf{x}_e, \mathbf{y}_e, \mathbf{p}_i, \mathbf{u}_i) \\ 0 &= q(\mathbf{x}_e, \mathbf{y}_e) - q_D(\mathbf{x}_i, \mathbf{y}_i, \mathbf{x}_e, \mathbf{y}_e, \mathbf{p}_i, \mathbf{u}_i) \end{aligned} \quad (11.2)$$

where p_D and q_D are the power injections of the device at the bus; and p and q are the power injections of all other devices connected to the same bus. The powers p_D and q_D are the positive if injected into the bus (generator) and negative otherwise (load).

11.2 Common Properties

Properties that are common to all devices are:

1. **con**: device data in the form of a matrix. Each row defines a new instance of the device, while each column defines a parameter. The format of the matrix **con** is described in detail for each device. A common notation used in the tables that describe the matrix **con** is as follows:
 - (a) A dagger \dagger before the column number indicates that the column is optional and can be omitted in the data file.
 - (b) A double dagger \ddagger before the column number indicates that the column is set and used by the PSAT internally. Any value set by the user will not have any effect.
2. **format**: a string containing the format of each row of the **con** matrix. Used for printing PSAT data to files.
3. **n**: total number of devices. This is the number of rows in
4. **ncol**: maximum number of columns of the matrix **con**. Used in case some parameters are optional. the **con** matrix.
5. **store**: data backup. This is basically a copy of the **con** matrix. This attribute can be used for automatically modifying some input data and can be useful for command line usage. See also Chapter 28.
6. **u**: vector containing the status of the devices. 1 means on-line, 0 means off-line. Some devices can lack this attribute (i.e. buses).

11.3 Common Methods

Methods that are common to most devices are:

1. **add**: adds one or more instances of the device.
2. **base**: converts device parameters to system power and voltage bases.
3. **block**: defines special operations of the device mask (used only for SIMULINK models).
4. **display**: prints the class properties in a structure-like format.
5. **dynidx**: assigns indexes to the state and algebraic variables of the device.
6. **fcall**: computes differential equations \mathbf{f} of the device.
7. **Fxcall**: computes Jacobian matrices \mathbf{f}_x , \mathbf{f}_y and \mathbf{g}_x of the device.
8. **gcall**: computes algebraic equations \mathbf{g} of the device.
9. **getxy**: returns indexes of state and algebraic variables of the device.
10. **Gcall**: computes the Jacobian matrix \mathbf{g}_y of the device.
11. **init**: cleans up all device properties.
12. **mask**: coordinates of the black mask (used in SIMULINK models and for drawing system maps).
13. **remove**: removes one or more instances of the device.
14. **restore**: restores device properties as given in the data file.
15. **setup**: initializes the main device properties using the property **con**.
16. **setx0**: compute the initial value of state variables of the device after power flow analysis.
17. **subsasgn**: assigns device properties. Properties that are not listed in this function cannot be assigned from outside the class.
18. **subsref**: returns device properties. Properties that are not listed in this function cannot be get from outside the class.
19. **XXclass**: class constructor. XX is the device specific code.
20. **xfirst**: assigns an initial guess to state and algebraic variables of the device.
21. **warn**: prints some warning messages.
22. **windup**: applies the anti-windup limiter if necessary.

Chapter 12

Power Flow Data

This chapter describes basic static components for power flow analysis. These are: buses, transmission lines, transformers, slack buses, constant active power and constant voltage generators (PV), constant power loads (PQ), constant power generators, constant admittances, interchange areas, and regions.

12.1 Bus

The minimal network unit is the *bus*. Shunt devices are connected to buses and series devices connect buses. In PSAT, buses are intended as topological nodes, not as physical connections. Thus a bus cannot be composed by different bars, cannot be divided into sections, and cannot be assigned an impedance or a status.

The data format is depicted in Table 12.1. Bus numbers, which can be in any order, and voltage ratings V_b are mandatory. Voltage magnitudes v_0 and phases θ_0 can be optionally set if the power flow solution is known or if a custom initial guess is needed. If voltages are not specified, a flat start is used ($v = 1$ at all buses except for the PV and slack generator buses, and $\theta = 0$). Once the power flow has been solved, voltage values can be saved in the data file using the *File/Save/Append Voltages* menu in the main window. Data associated with area and region numbers are optional, and will be used in future PSAT versions.

Bus devices are defined in the global variable `Bus`, which is an instance of the class `BUclass`. Relevant properties are as follows:

1. `con`: bus data.
2. `n`: total number of buses.
3. `int`: bus indexes.
4. `Pg`: active power injected in the network by generators.
5. `Qg`: reactive power injected in the network by generators.
6. `P1`: active power absorbed from the network by loads.

Table 12.1: Bus Data Format (`Bus.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	V_b	Voltage base	kV
† 3	v_0	Voltage amplitude initial guess	p.u.
† 4	θ_0	Voltage phase initial guess	rad
† 5	-	Area number (<i>not used yet...</i>)	int
† 6	-	Region number (<i>not used yet...</i>)	int

7. `Q1`: reactive power absorbed from the network by loads.
8. `island`: indexes of island buses.
9. `names`: bus names.
10. `store`: data backup.

The fields `Pg`, `Qg`, `P1` and `Q1` are a byproduct of the power flow solution. In the fields `P1` and `Q1` shunt power consumptions are not included, since the shunt admittances are included in the admittance matrix. The field `island` depends on breaker interventions: if a bus is disconnected from the grid after one or more breaker interventions, the resulting island is properly handled by the time domain simulation routine. This means that only buses that would create convergence problems are included in the `island` vector. These are PV generators and PQ buses, since they fix a constant power injection or consumption at the island bus. The definition of the `island` vector is done in the method `connectivity` of the transmission line class `LNclass`.

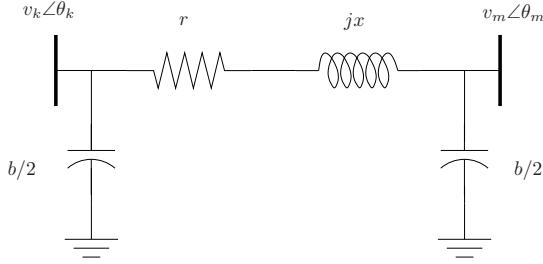
PSAT is device oriented, i.e. standard devices can be connected to any bus in any number and type. The only exception is the slack generator (`SW`) that has to be unique for each bus. Refer to Chapter 24 for a detailed description of component connection rules in PSAT.

12.2 Transmission Line

Fig. 12.1 depicts the circuit used for defining the transmission line lumped model, as described in many power system text books. The line equations are as follows:

$$\begin{aligned}
 p_k &= v_k^2(g_{km} + g_{k0}) - v_k v_m(g_{km} \cos(\theta_k - \theta_m) + b_{km} \sin(\theta_k - \theta_m)) \quad (12.1) \\
 q_k &= -v_k^2(b_{km} + b_{k0}) - v_k v_m(g_{km} \sin(\theta_k - \theta_m) - b_{km} \cos(\theta_k - \theta_m)) \\
 p_m &= v_m^2(g_{km} + g_{m0}) - v_k v_m(g_{km} \cos(\theta_k - \theta_m) - b_{km} \sin(\theta_k - \theta_m)) \\
 q_m &= -v_m^2(b_{km} + b_{m0}) + v_k v_m(g_{km} \sin(\theta_k - \theta_m) + b_{km} \cos(\theta_k - \theta_m))
 \end{aligned}$$

Transmission lines are defined in the structure `Line`, which is also used for transformers (see Section 12.3). The user can define data in absolute values or in

Figure 12.1: Transmission line π circuit.

p.u. In the latter case, the length ℓ of the line has to be $\ell = 0$. If $\ell \neq 0$, it is assumed that parameters are expressed in unit per km. Table 12.2 depicts the data format of transmission lines. I_{\max} , P_{\max} and S_{\max} define the limits for currents, active power flows and apparent power flows ($s = \sqrt{p^2 + q^2}$). These limits are not required in power flow analysis, but can be used for CPF and OPF analyses. Refer to Chapters 6 and 7 for details.

Transmission lines are defined in the global variable `Line`, which is an instance of the class `LNclass`. Relevant properties are as follows:

1. `con`: transmission line data.
2. `n`: total number of lines.
3. `Y`: admittance matrix of the network.
4. `Bp`, `Bpp`: admittance matrices for fast decoupled power flow analysis.
5. `fr`, `vfr`: indexes of voltages (from bus).
6. `to`, `vto`: indexes of voltages (to bus).
7. `p`, `q`: active and reactive power flows.
8. `no_build_y`: flag used to avoid computation of the admittance matrix. This flag has to set to 1 if only the admittance matrix is known, while line impedances are not known.
9. `store`: data backup.
10. `u`: connection status.

All lines included in the structure `Line` are used for building the network admittance matrix `Y`. It is also possible to define lines not to be included in the admittance matrix, by means of the structure `Lines`, whose data format is depicted in Table 12.3. These alternative transmission lines are defined in the global variable `Lines`, which is an instance of the class `LSclass`. Relevant properties are as follows:

Table 12.2: Line Data Format (`Line.con`)

Column	Variable	Description	Unit
1	k	From Bus	int
2	m	To Bus	int
3	S_n	Power rating	MVA
4	V_n	Voltage rating	kV
5	f_n	Frequency rating	Hz
6	ℓ	Line length	km
7	-	<i>not used</i>	-
8	r	Resistance	p.u. (Ω/km)
9	x	Reactance	p.u. (H/km)
10	b	Susceptance	p.u. (F/km)
† 11	-	<i>not used</i>	-
† 12	-	<i>not used</i>	-
† 13	I_{\max}	Current limit	p.u.
† 14	P_{\max}	Active power limit	p.u.
† 15	S_{\max}	Apparent power limit	p.u.
† 16	u	Connection status	{0, 1}

1. `con`: data chart of the `Lines` components.
2. `n`: total number of alternative lines.
3. `bus1`, `v1`: indexes of voltages (from bus).
4. `bus2`, `v2`: indexes of voltages (to bus).
5. `store`: data backup.
6. `u`: connection status.

12.3 Transformer

Two kinds of static transformers can be defined, i.e. two-winding transformers and three-winding transformers. Refer to Chapter 19 for models of regulating transformers.

12.3.1 Two-Winding Transformer

Two-winding transformers are modeled as series reactances without iron losses and their equations are included in (12.1). Table 12.4 depicts the transformer data format which is included in the global variable `Line` (an instance of the class `LNclass`). The primary and secondary voltage ratio k_T allows distinguishing between transmission lines and transformers: if $k_T = 0$, PSAT takes the component as a line, if

Table 12.3: Alternative Line Data Format (`Lines.con`)

Column	Variable	Description	Unit
1	k	From Bus	int
2	m	To Bus	int
3	S_n	Power rating	MVA
4	V_n	Voltage rating	kV
5	f_n	Frequency rating	Hz
6	r	Resistance	p.u.
7	x	Reactance	p.u.
8	b	Susceptance	p.u.
† 9	u	Connection status	{0, 1}

$k_T \neq 0$, the component is taken as a transformer. When $k_T \neq 0$, the line length ℓ is neglected, even if $\ell \neq 0$. The fixed tap ratio a and the fixed phase shift ratio ϕ are optional parameters. The fixed tap ratio unit is p.u./p.u. since it represents the ratio of the primary voltage in p.u. by the secondary voltage in p.u. For example, if the nominal voltages are $V_{1n} = 220$ kV and $V_{2n} = 128$ kV, and the actual tap positions of the transformer are $V_1 = 231$ kV and $V_2 = 128$ kV, the tap ratio a is as follows:

$$a = \frac{V_1}{V_{1n}} \cdot \frac{V_{2n}}{V_2} = \frac{231}{220} \cdot \frac{128}{128} = 1.05 \text{ p.u.}$$

12.3.2 Three-Winding Transformer

Three-winding transformers are internally modeled as three two-winding transformers in a Y connection, as depicted in Fig. 12.2. PSAT processes three-winding transformer data before running the power flow for the first time and adds one bus in the `Bus` structure and three new lines in the `Line` structure. Observe that the new bus will get same voltage rating, area and region as the primary winding bus.

The data format of three-winding transformers allows setting impedances of the triangle branches, whose relationships with the resulting star impedances are as follows:

$$\begin{aligned}\bar{z}_{12} &= \bar{z}_1 + \bar{z}_2 \\ \bar{z}_{13} &= \bar{z}_1 + \bar{z}_3 \\ \bar{z}_{23} &= \bar{z}_2 + \bar{z}_3\end{aligned}\tag{12.2}$$

Thus, one has:

$$\begin{aligned}\bar{z}_1 &= (\bar{z}_{12} + \bar{z}_{13} - \bar{z}_{23})/2 \\ \bar{z}_2 &= (\bar{z}_{12} + \bar{z}_{23} - \bar{z}_{13})/2 \\ \bar{z}_3 &= (\bar{z}_{13} + \bar{z}_{23} - \bar{z}_{12})/2\end{aligned}\tag{12.3}$$

Table 12.4: Transformer Data Format (Line.con)

Column	Variable	Description	Unit
1	k	From Bus	int
2	m	To Bus	int
3	S_n	Power rating	MVA
4	V_{n1}	Voltage rating of primary winding	kV
5	f_n	Frequency rating	Hz
6	-	<i>not used</i>	-
7	$k_T = \frac{V_{n1}}{V_{n2}}$	Nominal voltage ratio	kV/kV
8	r	Resistance	p.u.
9	x	Reactance	p.u.
10	-	<i>not used</i>	-
† 11	a	Fixed tap ratio	p.u./p.u.
† 12	ϕ	Fixed phase shift	deg
† 13	I_{\max}	Current limit	p.u.
† 14	P_{\max}	Active power limit	p.u.
† 15	S_{\max}	Apparent power limit	p.u.
† 16	u	Connection status	{0, 1}

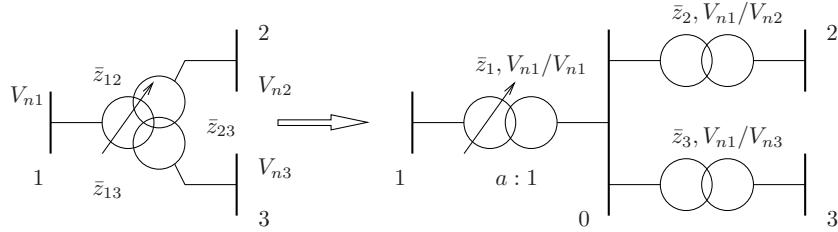


Figure 12.2: Three-winding transformer equivalent circuit.

Three-winding transformers are defined in the global variable `Twt`, which has only the `con` field and is an instance of the class `Twclass`. Table 12.5 depicts the three-winding transformer data format. Observe that PSAT clears the `Twt.con` matrix after processing it and that there is no function associated with three-winding transformer components.

12.4 Slack and $v\theta$ Generator

Slack generators are modeled as $v\theta$ buses, i.e. constant voltage magnitude and phase generators, as follows:

$$\begin{aligned} v &= v_G^0 \\ \theta &= \theta_G^0 \end{aligned} \tag{12.4}$$

Table 12.5: Three-Winding Transformer Data Format (`Twt.con`)

Column	Variable	Description	Unit
1	-	Bus number of the 1 th winding	int
2	-	Bus number of the 2 nd winding	int
3	-	Bus number of the 3 rd winding	int
4	S_n	Power rating	MVA
5	f_n	Frequency rating	Hz
6	V_{1n}	Voltage rating of the 1 th winding	kV
7	V_{2n}	Voltage rating of the 2 nd winding	kV
8	V_{3n}	Voltage rating of the 3 rd winding	kV
9	r_{12}	Resistance of the branch 1-2	p.u.
10	r_{13}	Resistance of the branch 1-3	p.u.
11	r_{23}	Resistance of the branch 2-3	p.u.
12	x_{12}	Reactance of the branch 1-2	p.u.
13	x_{13}	Reactance of the branch 1-3	p.u.
14	x_{23}	Reactance of the branch 2-3	p.u.
† 15	a	Fixed tap ratio	p.u./p.u.
† 16	I_{\max_1}	Current limit of the 1 th winding	p.u.
† 17	I_{\max_2}	Current limit of the 2 nd winding	p.u.
† 18	I_{\max_3}	Current limit of the 3 rd winding	p.u.
† 19	P_{\max_1}	Real power limit of the 1 th winding	p.u.
† 20	P_{\max_2}	Real power limit of the 2 nd winding	p.u.
† 21	P_{\max_3}	Real power limit of the 3 rd winding	p.u.
† 22	S_{\max_1}	Apparent power limit of the 1 th winding	p.u.
† 23	S_{\max_2}	Apparent power limit of the 2 nd winding	p.u.
† 24	S_{\max_3}	Apparent power limit of the 3 rd winding	p.u.
† 25	u	Connection status	{0, 1}

Table 12.6: Slack Generator Data Format (`SW.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	V_n	Voltage rating	kV
4	v_G^0	Voltage magnitude	p.u.
5	θ_g^0	Reference Angle	p.u.
† 6	q_G^{\max}	Maximum reactive power	p.u.
† 7	q_G^{\min}	Minimum reactive power	p.u.
† 8	v_G^{\max}	Maximum voltage	p.u.
† 9	v_G^{\min}	Minimum voltage	p.u.
† 10	p_G^0	Active power guess	p.u.
† 11	γ	Loss participation coefficient	-
† 12	z	Reference bus	{0, 1}
† 13	u	Connection status	{0, 1}

Each network must contain at least one slack generator. The angle θ_G^0 is assumed to be the reference angle of the system. If several slack generators are defined, only one can be chosen as the reference bus.¹ Table 12.6 depicts the slack generator data, which also contains data used in optimal power flow and continuation power flow analysis. In case of distributed slack bus model, the last two parameters p_G^0 and γ are mandatory and the following additional equation holds:

$$p = (1 + \gamma k_G) p_G^0 \quad (12.5)$$

where k_G is the distributed slack bus variable. If not specified, γ is assumed to be $\gamma = 1$. Slack generators are defined in the global variable `SW`, which is an instance of the class `SWclass`. Relevant properties are as follows:

1. `con`: slack generator data.
2. `n`: total number of slack generators.
3. `bus`, `vbus`: indexes of voltages of the slack buses.
4. `refbus`: indexes of buses used as phase reference.
5. `store`: data backup.
6. `u`: connection status.

¹Observe that PSAT allows defining several networks in the same data file. One slack bus must be defined for each network.

12.5 PV Generator

PV generators fix the voltage magnitude and the power injected at the buses where they are connected, as follows:

$$\begin{aligned} p &= p_G^0 \\ v &= v_G^0 \end{aligned} \quad (12.6)$$

In case of distributed slack bus model, the active power equation becomes:

$$p = (1 + \gamma k_G) p_G^0 \quad (12.7)$$

where k_G is the distributed slack bus variable and γ is the loss participation factor. Table 12.7 depicts PV generator data, which include reactive power and voltage limits needed for optimal power flow and continuation load flow analysis. Refer to Chapters 7 and 6 for details. If the check of PV reactive limits is enforced (see GUI for General Settings, which is depicted in Fig. 5.2), reactive power limits are used in power flow analysis as well. When a limit is violated, the PV generator is switched to a PQ bus, as follows:

$$\begin{aligned} p &= p_G^0 \\ q &= \{q_G^{\max}, q_G^{\min}\} \end{aligned} \quad (12.8)$$

After solving the power flow, the PQ buses are switched again to PV buses, assuming $v_G^0 = v$ at the bus where the PV generators are connected.

The user can define multiple PV generators at each bus. However, during the initialization step of the power flow analysis, PSAT defines a unique compound PV generator per bus. Inactive PV generators are discarded.

PV generators are defined in the global variable `PV`, which is an instance of the class `PVclass`. Relevant properties are as follows:

1. `con`: PV generator data.
2. `n`: total number of PV generators.
3. `bus`, `vbus`: indexes of voltages of the PV buses.
4. `u`: connection status.
5. `store`: data backup.

12.6 PQ Load

PQ loads are modeled as constant active and reactive powers:

$$\begin{aligned} p &= -p_L^0 \\ q &= -q_L^0 \end{aligned} \quad (12.9)$$

Table 12.7: PV Generator Data Format (PV.con)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	V_n	Voltage rating	kV
4	p_G^0	Active Power	p.u.
5	v_G^0	Voltage Magnitude	p.u.
† 6	q_G^{\max}	Maximum reactive power	p.u.
† 7	q_G^{\min}	Minimum reactive power	p.u.
† 8	v_G^{\max}	Maximum voltage	p.u.
† 9	v_G^{\min}	Minimum voltage	p.u.
† 10	γ	Loss participation coefficient	-
† 11	u	Connection status	{0, 1}

as long as voltages are within the specified limits. If a voltage limit is violated, PQ loads are converted into constant impedances, as follows:

$$\begin{aligned} p &= -p_L^0 v^2 / v_L^{\lim 2} \\ q &= -q_L^0 v^2 / v_L^{\lim 2} \end{aligned} \quad (12.10)$$

where v_L^{\lim} is v_L^{\max} or v_L^{\min} depending on the case. By default, maximum and minimum voltage limits are assumed to be 1.2 and 0.8 p.u. respectively. Table 12.8 depicts the PQ load data format. If $z = 0$, voltage limit control is disabled.

The user can define multiple PQ loads at each bus. However, during the initialization step of the power flow analysis, PSAT defines a unique compound PQ load per bus. Inactive PQ loads are discarded.

PQ loads can be converted to constant impedances after the power flow solution (see Section 9.2). If the option for changing the constant power loads into constant impedance is enabled, PQ loads are forced to switch to constant admittances, as follows:

$$\begin{aligned} p &= -p_L^0 v^2 / v_L^2 \\ q &= -q_L^0 v^2 / v_L^2 \end{aligned} \quad (12.11)$$

where v_L is the voltage value obtained with the power flow solution.

PQ loads are defined in the global variable `PQ`, which is an instance of the class `PQclass`. Relevant properties are as follows:

1. `con`: PQ load data.
2. `n`: total number of PQ loads.
3. `bus`, `vbus`: indexes of voltages of load buses.
4. `gen`: 1 if it is a PQ generator, 0 otherwise.

Table 12.8: PQ Load Data Format (PQ.con)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	V_n	Voltage rating	kV
4	p_L^0	Active Power	p.u.
5	q_L^0	Reactive Power	p.u.
† 6	v_L^{\max}	Maximum voltage	p.u.
† 7	v_L^{\min}	Minimum voltage	p.u.
† 8	z	Allow conversion to impedance	{0, 1}
† 9	u	Connection status	{0, 1}

5. P_0 : initial active power (used with non-conventional loads of Chapter 16).
6. Q_0 : initial reactive power (used with non-conventional loads of Chapter 16).
7. **store**: data backup
8. **u**: connection status.

Other static and dynamic load models are discussed in Chapter 16.

12.7 PQ Generator

PQ generators are modeled as constant active and reactive powers:

$$\begin{aligned} p &= p_G^0 \\ q &= q_G^0 \end{aligned} \tag{12.12}$$

as long as voltages are within the specified limits. If a voltage limit is violated, PQ generators are converted into constant impedances, as follows:

$$\begin{aligned} p &= p_G^0 v^2 / v_G^{\lim} \\ q &= q_G^0 v^2 / v_G^{\lim} \end{aligned} \tag{12.13}$$

where v_G^{\lim} is v_G^{\max} or v_G^{\min} depending on the case. By default, maximum and minimum voltage limits are assumed to be 1.2 and 0.8 p.u. respectively. Table 12.9 depicts the PQ generator data format. If $z = 0$, voltage limit control is disabled.

Internally, PSAT translates PQ generators into PQ loads with:

$$\begin{aligned} p_L^0 &= -p_G^0 \\ p_L^0 &= -p_G^0 \end{aligned} \tag{12.14}$$

The field **gen** of the PQ structure says if the load was converted from a PQ generator (see Section 12.6). Observe that PQ generators are NOT converted into constant

Table 12.9: PQ Generator Data Format (`PQgen.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	V_n	Voltage rating	kV
4	p_G^0	Active Power	p.u.
5	q_G^0	Reactive Power	p.u.
† 6	v_G^{\max}	Maximum voltage	p.u.
† 7	v_G^{\min}	Minimum voltage	p.u.
† 8	z	Allow conversion to impedance	{0, 1}
† 9	u	Connection status	{0, 1}

impedances after the power flow solution. Observe also that, since PQ generators are internally treated as negative PQ loads, it is not allowed connecting a PQ load at the same bus as a PQ generator.

PQ generators are defined in the global variable `PQgen` (an instance of the class `PQclass`), with the same properties as PQ loads.

12.8 Shunt

Shunt impedances are described by the following equations:

$$\begin{aligned} p &= -gv^2 \\ q &= bv^2 \end{aligned} \tag{12.15}$$

and are included in the network admittance matrix \mathbf{Y} . The susceptance b is negative for inductive charges, positive for capacitive ones. Shunts are defined in the global variable `Shunt`, which is an instance of the class `SHclass`. Relevant properties are as follows:

1. `con`: shunt impedance data.
2. `bus`, `vbus`: indexes of voltages of shunt buses.
3. `store`: data backup.
4. `u`: connection status.

12.9 Area & Region

PSAT allows defining areas and regions. These are currently used only for grouping variables to be plotted (see Section 9.3 for more details). Each area (region) is defined by a unique number, which has to correspond to one of the numbers defined

Table 12.10: Shunt Admittance Data Format (`Shunt.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	V_n	Voltage rating	kV
4	f_n	Frequency rating	Hz
5	g	Conductance	p.u.
6	b	Susceptance	p.u.
† 7	u	Connection status	{0, 1}

Table 12.11: Area & Regions Data Format (`Areas.con` and `Regions.con`)

Column	Variable	Description	Unit
1	-	Area/region number	int
2	-	Slack bus number for the area/region	int
3	S_n	Power rate	MVA
† 4	p_{ex}	Interchange export ($> 0 = \text{out}$)	p.u.
† 5	p_{tol}	Interchange tolerance	p.u.
† 6	$\Delta p\%$	Annual growth rate	%
† 7	p_{net}	Actual real power net interchange	p.u.
† 8	q_{net}	Actual reactive power net interchange	p.u.

in the column 5 (6) of the `Bus.con` data. Areas correspond to *loss zone data*, while regions correspond to the *interchange area data* in the IEEE common data format [137].

A slack bus can be defined for each area and region. This slack bus is just a “suggestion” and does not affect or redefine the `SW.con` data. The slack bus number can be zero.

Areas and regions are defined in the global variables `Areas` and `Regions`, respectively, which are instances of the class `ARclass`. Relevant properties are as follows:

1. `con`: area/region data.
2. `n`: number of areas/regions.
3. `bus`: cell array of bus indexes within each area/region.
4. `int`: area/region indexes.
5. `slack`: indexes of slack buses within each area/region.
6. `names`: area/region names.

Chapter 13

CPF and OPF Data

This section describes the components needed for the OPF analysis.

The basic components are the slack generator, the PV generator and the PQ load. As defined in Tables 12.6, 12.7 and 12.8, the user can define reactive power and voltage limits for the generation, and voltage limits for the loads. Furthermore, in the definition of transmission lines and transformers, the user can set the limits for maximum current, active power and apparent power flows.¹

For the voltages at all the N network buses, one has:

$$\mathbf{v}^{\min} \leq \mathbf{v} \leq \mathbf{v}^{\max} \quad (13.1)$$

whereas the limits for the generation (PV and slack generators) are the following:

$$\mathbf{q}_G^{\min} \leq \mathbf{q}_G \leq \mathbf{q}_G^{\max} \quad (13.2)$$

Finally, flows constraints are:

$$|\phi| \leq \phi^{\max} \quad (13.3)$$

If no constraint is defined for lines or transformers, i.e. the 13th, 14th or 15th element is left blank or set to zero in the `Line.con` chart, a “huge” limit for the flow is used.

The components `SW`, `PV`, `PQ` and `Line` allow to define only some security limits. Cost parameters and additional market constraints are defined in other structures, described below, that are specifically used for the OPF routines. For the generation three global variables can be defined:

1. `Supply`: power bids, generator power directions and limits (instance of the class `SUclass`).
2. `Rsrv`: power reserve data (instance of the class `RSclass`).
3. `Rmpg`: power ramp data (instance of the class `RGclass`).

whereas for the load side the following two global variables are available:

¹Which flow type has to be used in the OPF analysis can be specified in a pop-up menu of the OPF GUI.

1. **Demand**: power bids, load power directions and limits (instance of the class `DMclass`).
2. **Rmpl**: power ramp data (instance of the class `RLclass`).

Each global variable is composed of at least the following fields:

1. **con**: data.
2. **n**: total number of elements.
3. **bus**: indexes of buses at which elements are connected.
4. **store**: data backup.
5. **u**: connection status.

13.1 Generator Supply

The `Supply` structure defines the basic data for generations bids and costs, as depicted in Table 13.1. This structure is always required for running the OPF. The user has to define cost parameters that can be both for active and reactive power generation, as defined by the following equations:²

$$\begin{aligned} c(\mathbf{p}_S) &= \sum c_{p_0} + c_{p_1} p_S + c_{p_2} p_S^2 \\ c(\mathbf{q}_G) &= \sum c_{q_0} + c_{q_1} q_G + c_{q_2} q_G^2 \end{aligned} \quad (13.4)$$

Power supply blocks are:

$$p_S^{\min} \leq p_S \leq p_S^{\max} \quad (13.5)$$

Once the OPF analysis has been completed, the resulting p_S^* is set up in the matrix `Supply.con`. While setting up data, the user can just put zeros in the 6th column of the matrix `Supply.con`.

It is also possible to set an unit commitment initial status u_C . The unit commitment problem is currently available only for the GAMS interface. See Chapter 30 for details.

Finally, the user can set a tie breaking cost k_{TB} . The tie breaking involves a penalty cost k_{TB} prorated by the amount scheduled over the maximum amount that could be scheduled for the generator by means of a quadratic function added to the objective function:

$$c_{TB} = k_{TB} \frac{p_S^2}{p_S^{\max}} \quad (13.6)$$

²The difference in the notation of costs C in Table 13.1 and c in the equations is due to the power unit. C is used for absolute units, while c for p.u. The relation is as follows:

$$\begin{aligned} c_{p_0} &= C_{p_0}/S_n \\ c_{p_1} &= C_{p_1} \\ c_{p_2} &= C_{p_2} S_n \end{aligned}$$

where S_n is the system power base. Similar transformations applies for the reactive power costs.

Table 13.1: Power Supply Data Format (Supply.con)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3 ^a	p_S^0	Active power direction	p.u.
4	p_S^{\max}	Maximum power bid	p.u.
5	p_S^{\min}	Minimum power bid	p.u.
‡ 6 ^b	p_S^*	Actual active power bid	p.u.
7	C_{p_0}	Fixed cost (active power)	\$/h
8	C_{p_1}	Proportional cost (active power)	\$/MWh
9	C_{p_2}	Quadratic cost (active power)	\$/MW ² h
10	C_{q_0}	Fixed cost (reactive power)	\$/h
11	C_{q_1}	Proportional cost (reactive power)	\$/MVArh
12	C_{q_2}	Quadratic cost (reactive power)	\$/MVAr ² h
13	u_C	Unit commitment initial status	{0, 1}
14	k_{TB}	Tie breaking cost	\$/MWh
15	γ	Loss participation factor	-
16	q_G^{\max}	Maximum reactive power	p.u.
17	q_G^{\min}	Minimum reactive power	p.u.
† 18	C_S^{up}	Congestion up cost	\$/h
† 19	C_S^{dw}	Congestion down cost	\$/h
† 20	u	Connection status	{0, 1}

a: This field is used only for the CPF analysis.

b: This field is an output of the OPF analysis.

If the generator does not supply power, this cost is zero, whereas if p_S is close to the maximum power the tie breaking cost increases quadratically and penalizes the generator. Thus two otherwise tied energy offers will be scheduled to the point where their modified costs are identical, effectively achieving a prorated result. Generally the value of k_{TB} should be small (e.g. 0.0005). The default value is zero.

In case there are more than one supply block for bus, the reactive power limits are shared among the suppliers using q_G^{\max} and q_G^{\min} data. In case of $q_G^{\max} = 0$ and $q_G^{\min} = 0$, slack and PV generator reactive power limits will be used.

Congestion up and down costs are used in the congestion management. C^{up}_S and C^{dw}_S are the costs for increasing and decreasing the current accepted supply bid.

The structure `Supply` is also used in the continuation power flow (see Chapter 6) for defining the pattern of generator increase with respect to the base case. In this case the active power direction P_{S_0} has to be set. For optimal power flow computation this value can be left zero.

Table 13.2: Power Reserve Data Format (`Rsrv.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	p_R^{\max}	Maximum power reserve	p.u.
4	p_R^{\min}	Minimum power reserve	p.u.
5	C_r	Reserve offer price	\$/MWh
† 6	u	Connection status	{0, 1}

13.2 Generator Reserve

The operating reserve of a system is associated with the power that is not directly used by loads but can be requested and generators have to provide quickly. The power reserve has an associated cost:

$$c(\mathbf{p}_R) = \sum c_r p_R \quad (13.7)$$

and limits:

$$p_R^{\min} \leq p_R \leq p_R^{\max} \quad (13.8)$$

along with the inequalities for ensuring that the sum of the power supply and the power reserve is less than the total available power supply p_S^{\max} and that the total power reserve must be less than the total power demand:

$$\begin{aligned} p_S + p_R &\leq p_S^{\max} \\ \sum p_R &\leq \sum p_D \end{aligned} \quad (13.9)$$

The structure `Rsrv` defines these data, as reported in Table 13.2. The power reserve vector \mathbf{p}_R as obtained by OPF routines are stored in the field `Pr`.

13.3 Generator Power Ramp

Generation facilities have limits on their ability to move from one level of production to another, and these limits are generally taken in account by the so called ramp constraints. The structure `Rmpg` defines the generator ramp data, as reported in Table 13.3.³

The parameters used in the optimization routine are the up and down ramp rates, i.e. r^{up} and r^{dw} . These quantities express the amount of power that can be moved each minute up or down by the generator and are associated to technical limits of the generation plants. The constraints are the following:

$$\begin{aligned} p_S^t - p_S^{t-1} &\leq p_S^{\max} r^{\text{up}} \Delta t \\ -p_S^t + p_S^{t-1} &\leq p_S^{\max} r^{\text{dw}} \Delta t \end{aligned} \quad (13.10)$$

³These constraints have not been implemented yet but will be included soon in future versions.

Table 13.3: Generator Power Ramp Data Format (`Rmpg.con`)

Column	Variable	Description	Unit
1	-	Supply number	int
2	S_n	Power rating	MVA
3	r^{up}	Ramp rate up	p.u./h
4	r^{dw}	Ramp rate down	p.u./h
5	UT	Minimum # of period up	h
6	DT	Minimum # of period down	h
7	UT_i	Initial # of period up	int
8	DT_i	Initial # of period down	int
9	C_{SU}	Start up cost	\$
† 10	u	Connection status	{0, 1}

Along with these ramp limits, the user can define also a maximum reserve ramp rate r_r^{\max} , that multiplied by the time interval Δt , expresses the maximum amount of power that can be dedicated to the reserve, thus:

$$p_R \leq r_r^{\max} \Delta t \quad (13.11)$$

If the generator output is low, also the operating reserve can decrease, and the operating reserve loading point p_{RLP} allows to reduce the power reserve for low outputs:

$$p_R \leq p_S \frac{r_r^{\max} \Delta t}{p_{RLP}} \quad (13.12)$$

Thus, the power reserve p_R will be the minimum between (13.11) and (13.12).

13.4 Load Demand

The **Demand** structure defines the basic data for load demand bids and costs, as presented in Table 13.4. The user has to define the maximum and minimum power bids, as well as the cost coefficients that can be both for active and reactive powers. The cost functions, similar to (13.4) for the power supplies, are:

$$\begin{aligned} c_1(\mathbf{p}_D) &= \sum c_{p_0} + c_{p_1} p_D + c_{p_2} p_D^2 \\ c_2(\mathbf{p}_D) &= \sum c_{q_{0i}} + c_{q_1} p_D \tan(\phi) + c_{q_2} p_D^2 \tan(\phi)^2 \end{aligned} \quad (13.13)$$

where

$$\tan \phi = \frac{q_D^0}{p_D^0} \quad (13.14)$$

As for the constraints, one has:

$$p_R^{\min} \leq p_R \leq p_R^{\max} \quad (13.15)$$

The power factor angle ϕ is used for computing reactive power costs. The power factor angle is obtained using power directions p_D^0 and q_D^0 . Observe that, their ratio can be different from the power factor of the base case load defined in the PQ structure .

Once the OPF analysis has been completed, the resulting p_D^* is set up in the matrix `Demand.con`. While setting up data, the user can just put zeros in the 7th column of the matrix `Demand.con`.

It is also possible to set an unit commitment initial status u_C . The unit commitment problem is currently available only for the GAMS interface. See Chapter 30 for details.

Finally, the user can set a tie breaking cost k_{TB} . The tie breaking involves a penalty cost k_{TB} prorated by the amount scheduled over the maximum amount that could be scheduled for the load by means of a quadratic function added to the objective function:

$$c_{TB} = k_{TB} \frac{p_D^2}{p_D^{\max}} \quad (13.16)$$

If the load does not consume power, this cost is zero, whereas if p_D is close to the maximum power the tie breaking cost increases quadratically and penalizes the load. Thus two otherwise tied energy demands will be scheduled to the point where their modified costs are identical, effectively achieving a prorated result. Generally the value of k_{TB} should be small (e.g. 0.0005). The default value is zero.

Congestion up and down costs are used in the congestion management. c_D^{up} and c_D^{dw} are the costs for increasing and decreasing the current accepted demand bid.

The structure `Demand` is also used in the continuation load flow (see Chapter 6) for defining the pattern of load increase with respect to the base case. The CPF routine uses p_D^0 and q_D^0 to define the active and reactive power directions, respectively.

13.5 Demand Profile

The structure `Ypdp` defines the demand profile for multi-period market clearing models.⁴ The user can input data using two different formats, namely free format and yearly profile.

The free format is simply a vector (of any length $\neq 206$) of numbers representing the percentage of power demand for the period t . Each element of the vector defines a period t , and the whole vector length represent the time horizon of the market clearing model. For example:

```
Ypdp.con = [80 90 100 110 100];
```

defines 5 time periods, for which the percentage ξ^t of the demand is 80, 90, 100, 110 and 100%, respectively. The percentage multiplies the base case powers p_L^0 (if

⁴Multi-period market clearing models are currently available only for the PSAT-GAMS interface.

Table 13.4: Power Demand Data Format (`Demand.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3 ^a	p_D^0	Active power direction	p.u.
4 ^a	q_D^0	Reactive power direction	p.u.
5	p_D^{\max}	Maximum power bid	p.u.
6	p_D^{\min}	Minimum power bid	p.u.
‡ 7 ^b	p_D^*	Optimal active power bid	p.u.
8	C_{p_0}	Fixed cost (active power)	\$/h
9	C_{p_1}	Proportional cost (active power)	\$/MWh
10	C_{p_2}	Quadratic cost (active power)	\$/MW ² h
11	C_{q_0}	Fixed cost (reactive power)	\$/h
12	C_{q_1}	Proportional cost (reactive power)	\$/MVArh
13	C_{q_2}	Quadratic cost (reactive power)	\$/MVAr ² h
14	u_C	Unit commitment initial status	{0, 1}
15	k_{TB}	Tie breaking cost	\$/MWh
† 16	C_D^{up}	Congestion up cost	\$/h
† 17	C_D^{dw}	Congestion down cost	\$/h
† 18	u	Connection status	{0, 1}

a: These fields are used for both the CPF analysis and the OPF analysis.

b: This field is an output of the OPF analysis.

used) and demand bid limits p_D^{\max} and p_D^{\min} , so that:

$$p_L^0(t) = \frac{\xi^t}{100} p_L^0 \quad \forall t \in \mathcal{T} \quad (13.17)$$

$$p_D^{\max}(t) = \frac{\xi^t}{100} p_D^{\max} \quad \forall t \in \mathcal{T} \quad (13.18)$$

$$p_D^{\min}(t) = \frac{\xi^t}{100} p_D^{\min} \quad \forall t \in \mathcal{T} \quad (13.19)$$

where $\mathcal{T} = \{1, 2, 3, 4, 5\}$ for this example.

The yearly demand profile is used to define a database for a one day long time horizon ($\mathcal{T} = 24$ hours). The user can tune the coefficients by choosing the season and the day of the week. Thus, the 24 coefficients ξ^t are computed as follows:

$$\xi^t = \frac{k_\alpha^t(\alpha)}{100} \cdot \frac{k_\beta(\beta)}{100} \cdot \frac{k_\gamma(\gamma)}{100} 100 \quad (13.20)$$

where k_α^t (24 elements), k_β (scalar) and k_γ (scalar) are in % and represent the kind of the day, the day of the week and the week of the year, respectively, and the indexes α , β and γ are as follows:

α : index of the kind of the day in the range from 1 to 6, with the following notation:

- 1: winter working day
- 2: winter weekend
- 3: summer working day
- 4: summer weekend
- 5: spring/fall working day
- 6: spring/fall weekend

β : day of the week in the range from 1 (Monday) to 7 (Sunday).

γ : week of the year in the range from 1 to 52.

13.6 Load Ramp

Although less commonly used than the generation ramp rate, it is possible to define load ramp rates. These take in account the load ability to move from one level of consumption to another. The structure `Rmp1` defines the load ramp data, as reported in Table 13.6.⁵

The parameters used in the optimization routine are the up and down ramp rates, i.e. r^{up} and r^{dw} . These quantities express the amount of power that can be

⁵These constraints have not been implemented yet but will be included in future versions.

Table 13.5: Demand Profile Data Format (`Ypdp.con`)

Column	Variable	Description	Unit
1-24	$k_\alpha^t(1)$	Daily profile for a winter working day	%
25-48	$k_\alpha^t(2)$	Daily profile for a winter weekend	%
49-72	$k_\alpha^t(3)$	Daily profile for a summer working day	%
73-96	$k_\alpha^t(4)$	Daily profile for a summer weekend	%
97-127	$k_\alpha^t(5)$	Daily profile for a spring/fall working day	%
121-144	$k_\alpha^t(6)$	Daily profile for a spring/fall weekend	%
145-151	k_β	Profile for the days of the week	%
152-203	k_γ	Profile for the weeks of the year	%
204	α	Kind of the day	$\{1, \dots, 6\}$
205	β	Day of the week	$\{1, \dots, 7\}$
206	γ	Week of the year	$\{1, \dots, 52\}$

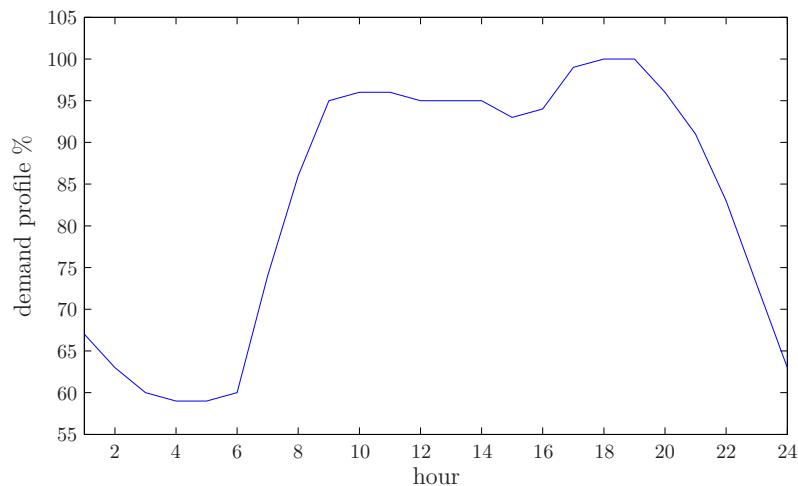


Figure 13.1: Example of daily demand profile.

Table 13.6: Load Ramp Data Format (`Rmpl.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	r^{up}	Ramp rate up	p.u./min
4	r^{dw}	Ramp rate down	p.u./min
5	T_{up}	Minimum up time	min
6	T_{down}	Minimum down time	min
7	n_{up}	Number of period up	int
8	n_{down}	Number of period down	int
† 9	u	Connection status	{0, 1}

moved each minute up or down by the load and are associated to technical limits in load facilities as follows:

$$\begin{aligned} p_D^t - p_D^{t-1} &\leq p_D^{\max} r^{\text{up}} \Delta t \\ -p_D^t + p_D^{t-1} &\leq p_D^{\max} r^{\text{dw}} \Delta t \end{aligned} \quad (13.21)$$

These equations are conceptually similar to (13.10) for the generation ramp rate, and uses the same time interval Δt defined in the OPF window.

Chapter 14

Faults & Breakers

This chapter describes symmetrical three phase faults and breakers.

14.1 Fault

Table 14.1 depicts data for three phase faults, i.e. the time t_f of occurrence of the fault, the clearance time t_c and the internal impedance of the fault r_f and x_f .

During time domain simulations, a time vector for computing a point slightly before and slightly after the fault occurrences is created. When the faults or the fault clearances occur, the shunt admittances of the network are modified and the admittance matrix is recomputed. Fault clearing can lead, in some idiosyncratic cases, to convergence issue in restoring bus voltages. If this is the case, one can try to disable the option “Reset pre-fault bus angles after fault clearing” in the Advanced Settings GUI (available in the menu Edit of the main window).

Three phase faults are defined in the global variable **Fault**, which is an instance of the class **FTclass**. Relevant properties are as follows:

1. **con**: Fault data.
2. **n**: total number of faults.
3. **bus**, **vbus**: indexes of voltages of buses to which faults are connected.
4. **dat**: internal data.
5. **V**: vector of pre-fault voltages.
6. **ang**: vector of pre-fault angles.
7. **delta**: mean of synchronous machine rotor angles.
8. **store**: data backup.

Table 14.1: Fault Data Format (`Fault.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	V_n	Voltage rating	kV
4	f_n	Frequency rating	Hz
5	t_f	Fault time	s
6	t_c	Clearance time	s
7	r_f	Fault resistance	p.u.
8	x_f	Fault reactance	p.u.

14.2 Breaker

Table 14.2 depicts the data format for transmission line breakers. Besides MVA, kV and Hz ratings, the user can set up two intervention times and a status for the breaker.

The status u is used for the construction of the network admittance matrix Y . If $u = 1$, transmission line parameters are used as they are. If $u = 0$, the line status is set to open.¹ Each intervention toggles the status of the breaker. The position of the breaker on a line (i.e. at the beginning or at the end of the line) is not relevant.

During time domain simulations, the integration method computes one point slightly before and one slightly after ($t = 10^{-6}$ s) each breaker intervention. The admittance matrix is rebuilt after each breaker intervention. First and second breaker interventions can be disabled using Boolean data u_1 and u_2 , respectively. For example, if $u_1 = 1$, the first intervention is applied; if $u_1 = 0$, the first intervention is skipped.

Breakers are defined in the global variable `Breaker`, which is an instance of the class `BKclass`. Relevant properties are as follows:

1. `con`: Breaker data.
2. `n`: total number of breakers.
3. `line`: vector of line numbers to which breakers are connected.
4. `bus`: vector of bus numbers to which breakers are connected.
5. `u`: Boolean vector indicating the status of the breaker.
6. `t1`: vector of first intervention times.
7. `t2`: vector of second intervention times.
8. `store`: data backup.

Table 14.2: Breaker Data Format (`Breaker.con`)

Column	Variable	Description	Unit
1	-	Line number	int
2	-	Bus number	int
3	S_n	Power rating	MVA
4	V_n	Voltage rating	kV
5	f_n	Frequency rating	Hz
6	u	Connection status	{0, 1}
7	t_1	First intervention time	s
8	t_2	Second intervention time	s
† 9	u_1	Apply first intervention	{0, 1}
† 10	u_2	Apply second intervention	{0, 1}

Breakers works properly **only** if connected to transmission lines. Connecting breakers to other components (e.g. synchronous machines) is not allowed. Refer to Chapter 24 for more details. To simulate the disconnection of a single device, use the status property `u` of that device.

¹Observe that disabling single-end lines leads to islanded buses and can result in convergence problems.

Chapter 15

Measurements

This chapter describes components intended for measurements of non-standard quantities during time domain simulations. Measurement devices currently implemented in PSAT are the bus frequency measurement and the Phasor Measurement Unit (PMU).

15.1 Bus Frequency Measurement

The bus frequency measurement is obtained by means of a high-pass and a low-pass filter, as depicted in Fig. 15.1. Differential equations are as follows:

$$\begin{aligned}\dot{x} &= \frac{1}{T_f} \left(\frac{1}{2\pi f_0} \frac{1}{T_f} (\theta - \theta_0) - x \right) \\ \dot{\omega} &= (\Delta\omega + 1 - \omega)/T_\omega\end{aligned}\tag{15.1}$$

where $\Delta\omega$ is:

$$\Delta\omega = -x + \frac{1}{2\pi f_0} \frac{1}{T_f} (\theta - \theta_0)\tag{15.2}$$

Bus frequency measurement data are stored in the global variable `Busfreq`, which is an instance of the class `BFclass`. Relevant properties are as follows:

1. `con`: Bus frequency measurement data.
2. `n`: total number of components.

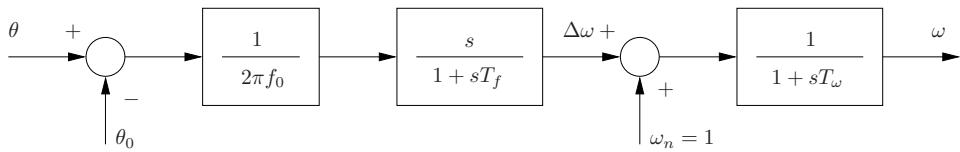


Figure 15.1: Bus frequency measurement filter.

Table 15.1: Bus Frequency Measurement Data Format (`Busfreq.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	T_f	Time constant of the high-pass filter	s
3	T_ω	Time constant of the low-pass filter	s
† 4	u	Connection status	{0, 1}

- 3. `bus`: indexes of measured buses.
- 4. `dat`: Bus frequency measurement parameters.
- 5. `x`: indexes of state variables x .
- 6. `w`: indexes of state variables ω .
- 7. `store`: data backup.
- 8. `u`: connection status.

Table 15.1 depicts the data format for the bus frequency measurements.

15.2 Phasor Measurement Unit (PMU)

A Phasor Measurement Unit (PMU) is a device able to measure the magnitude and the angle of a phasor. Basic concepts, definitions and applications about PMUs can be found in [136].

Let us define a sinusoidal quantity:

$$x(t) = X_M \cos(\omega t + \phi) \quad (15.3)$$

its phasor representation is:

$$X = \frac{X_M}{\sqrt{2}} e^{j\phi} \quad (15.4)$$

The phasor is defined for a pure constant sinusoid, but it can also be used for transients, assuming that the phasor is the fundamental frequency component of a waveform over a finite interval (observation window).

PMUs works on sampled measures (see Fig. 15.2). In the case of $x(t)$, we can define the samples signal x_k at $t = k\tau$, where τ is the sampling interval. Using a Discrete Fourier Transform (DFT), the phasor X is given by:

$$X = \frac{1}{\sqrt{2}} \frac{2}{N} (X_c - jX_s) \quad (15.5)$$

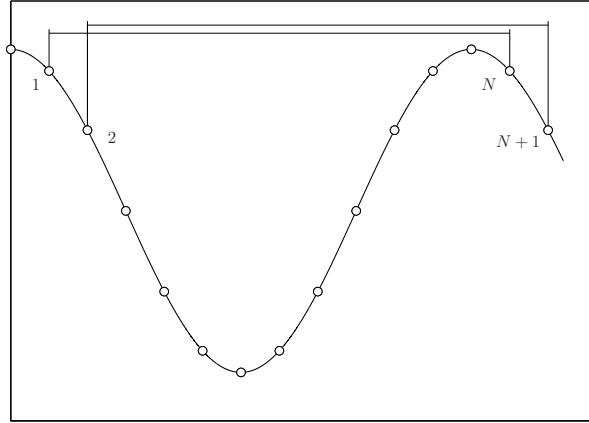


Figure 15.2: Phasors from sample data.

where N is the number of samples in one period of the nominal fundamental frequency f_0 , and:

$$X_c = \sum_{k=1}^N x_k \cos k\theta \quad (15.6)$$

$$X_s = \sum_{k=1}^N x_k \sin k\theta \quad (15.7)$$

and θ is the sampling angle associated with the sampling interval τ , as follows:

$$\theta = \frac{2\pi}{N} = 2\pi f_0 \tau \quad (15.8)$$

A typical sampling rate in many relaying and measurements functions is 12 times the power system frequency (e.g. 720 Hz for a 60 Hz power system).

Equation (15.5) represents a non-recursive DFT calculation. A recursive calculation is an efficient method for time varying phasors. Let X^r be the phasor corresponding to the data set $x\{k = r, r + 1, \dots, N + r - 1\}$, and let a new data sample be obtained to produce a new data set $x\{k = r + 1, r + 2, \dots, N + r\}$. The recursive phasor corresponding to the new data window X^{r+1} is as follows:

$$X^{r+1} = X^r + \frac{1}{\sqrt{2}} \frac{2}{N} (x_{N+r} - x_r) e^{-jr\theta} \quad (15.9)$$

A recursive calculation through a moving window data sample is faster than a non-recursive one, needs only two sample data at each calculation (x_{N+r} and x_r) and provides a stationary phasor.

If the quantity $x(t)$ undergoes a transient, the moving window detects the amplitude and angle variations with a delay which depends on the time sample rate

τ . If the system frequency f_0 undergoes a variation Δf , the positive sequence of the phasor undergoes the following change, at each r^{th} time sampling:

$$X^r(f_0 + \Delta f) = X e^{-j(N-1)\pi\Delta f \Delta t} \frac{\sin(N\Delta f \Delta t)}{N \sin(\Delta f \Delta t)} e^{j2\pi r \Delta f \Delta t} \quad (15.10)$$

thus, the rate of change of the phasor angle is as follows:

$$\frac{d\psi}{dt} = 2\pi\Delta f \quad (15.11)$$

The PMU model implemented in PSAT is used for bus voltage magnitude and phase measurements. The measurement is modeled as a simple low pass filter, as follows:

$$\dot{v}_m = (v - v_m)/T_m \quad (15.12)$$

$$\dot{\theta}_m = (\theta - \theta_m)/T_\theta \quad (15.13)$$

where v and θ are the voltage magnitude and phase, respectively.

PMU data in the global variable `Pmu`, which is an instance of the class `PMclass`.¹ Relevant properties are as follows:

1. `con`: PMU data.
2. `n`: total number of components.
3. `dat`: PMU parameters.
4. `vm`: indexes of state variables v_m .
5. `thetam`: indexes of state variables θ_m .
6. `store`: data backup.
7. `u`: connection status.

Table 15.2 depicts the data format for the PMU devices.

¹Do not confuse the global variable `Pmu` for PMU devices models with the global structure `PMU` which is used in the PMU placement algorithms illustrated in Chapter 10.

Table 15.2: Phasor Measurement Unit Data Format (`Pmu.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	V_n	Voltage rate	kV
3	f_n	Frequency rate	Hz
4	T_v	Voltage magnitude time constant	s
5	T_θ	Voltage phase time constant	s
† 6	u	Connection status	{0, 1}

Chapter 16

Loads

This chapter describes static and dynamic nonlinear loads. They are voltage dependent load, ZIP load, frequency dependent load, exponential recovery load, thermostatically controlled load,, Jimma's load , and mixed load. These models requires a PQ load in order to initialize parameters and state variables. Voltage dependent and ZIP loads can be optionally included in the power flow analysis.

Note: in this chapter, in order to simplify the notation, active and reactive load powers p and q are positive if absorbed from the network.

16.1 Voltage Dependent Load

Voltage dependent loads (VDL) are loads whose powers are monomial functions of the bus voltage, as follows:

$$\begin{aligned} p &= p^0(v/v^0)^{\alpha_p} \\ q &= q^0(v/v^0)^{\alpha_q} \end{aligned} \quad (16.1)$$

where v^0 is the initial voltage at the load bus as obtained by the power flow solution. Other parameters are defined in Table 16.1, which illustrates the VDL data. The p.u. values of p^0 and q^0 are set by the user if the parameter $z = 0$, i.e., if the VDL is included in the power flow analysis. If $z = 1$, the VDL is initialized after the power flow analysis, and p^0 and q^0 are computed based on the PQ load powers p_L^0 and q_L^0 :

$$\begin{aligned} p^0 &= \frac{k_p}{100} p_L^0 \\ q^0 &= \frac{k_q}{100} q_L^0 \end{aligned} \quad (16.2)$$

Thus, if $z = 1$, a PQ load must be connected at the VDL bus. Otherwise, if $z = 0$, a PQ load is not necessary.

Table 16.1: Voltage Dependent Load Data Format (`Mn.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	V_n	Voltage rating	kV
4	k_p or p^0	Active power rating	% or p.u.
5	k_q or q^0	Reactive power rating	% or p.u.
6	α_p	Active power exponent	-
7	α_q	Reactive power exponent	-
8	z	Initialize after power flow	{1, 0}
† 9	u	Connection status	{1, 0}

Observe that equations (16.1) are a simplification of the nonlinear general exponential voltage frequency dependent load described in Section 16.3.

Voltage dependent loads are defined in the global variable `Mn`, which is an instance of the class `MNclass`. Relevant properties are as follows:

1. `con`: voltage dependent load data.
2. `n`: total number of voltage dependent loads.
3. `bus`, `vbus`: indexes of voltages of buses to which voltage dependent loads are connected.
4. `init`: status for power flow computations.
5. `u`: connection status.
6. `store`: data backup.

16.2 ZIP Load

The polynomial or ZIP loads are loads whose powers are a quadratic expression of the bus voltage, as follows:

$$\begin{aligned} p &= p_z^0 \left(\frac{v}{v_0} \right)^2 + p_i^0 \frac{v}{v_0} + p_p^0 \\ q &= q_z^0 \left(\frac{v}{v_0} \right)^2 + q_i^0 \frac{v}{v_0} + q_p^0 \end{aligned} \quad (16.3)$$

where v^0 is the initial voltage at the load bus as obtained by the power flow solution. Other parameters are defined in Table 16.2, which illustrates the ZIP load data format. The values of p_z^0 , p_i^0 , p_p^0 , q_z^0 , q_i^0 and q_p^0 are set by the user if the parameter $z = 0$, i.e., if the ZIP load is included in the power flow analysis. If $z = 1$, the ZIP

Table 16.2: ZIP Load Data Format (P1.con)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	V_n	Voltage rating	kV
4	f_n	Frequency rating	Hz
5	k_{pz} or p_z^0	Conductance	% or p.u.
6	k_{pi} or p_i^0	Active current	% or p.u.
7	k_{pp} or p_p^0	Active power	% or p.u.
8	k_{qz} or q_z^0	Susceptance	% or p.u.
9	k_{qi} or q_i^0	Reactive current	% or p.u.
10	k_{qp} or q_p^0	Reactive power	% or p.u.
11	z	Initialize after power flow	{1,0}
† 12	u	Connection status	{1,0}

load is initialized after the power flow analysis, and the parameters in (16.3) are defined based on the PQ load powers p_L^0 and q_L^0 , as follows:

$$\begin{aligned} p_z^0 &= \frac{k_{pz}}{100} \frac{p_L^0}{v^0}, & p_i^0 &= \frac{k_{pi}}{100} \frac{p_L^0}{v^0}, & p_p^0 &= \frac{k_{pp}}{100} p_L^0; \\ q_z^0 &= \frac{k_{qz}}{100} \frac{q_L^0}{v^0}, & q_i^0 &= \frac{k_{qi}}{100} \frac{q_L^0}{v^0}, & q_p^0 &= \frac{k_{qp}}{100} q_L^0. \end{aligned} \quad (16.4)$$

Thus, if $z = 1$, a PQ load must be connected at the ZIP load bus. Otherwise, if $z = 0$, a PQ load is not necessary.

Voltage dependent loads are defined in the global variable P1, which is an instance of the class PLclass. Relevant properties are as follows:

1. **con**: ZIP load data.
2. **n**: total number of ZIP loads.
3. **bus**, **vbus**: indexes of voltages of buses to which ZIP loads are connected.
4. **init**: status for power flow computations.
5. **store**: data backup.
6. **u**: connection status.

16.3 Frequency Dependent Load

A generalized exponential voltage frequency dependent load is modeled by the following set of DAE [57]:

$$\begin{aligned}\dot{x} &= -\frac{\Delta\omega}{T_F} \\ 0 &= x + \frac{1}{2\pi f_n} \frac{1}{T_F} (\theta - \theta^0) - \Delta\omega \\ p &= p^0 \left(\frac{v}{v^0} \right)^{\alpha_p} (1 + \Delta\omega)^{\beta_p} \\ q &= q^0 \left(\frac{v}{v^0} \right)^{\alpha_q} (1 + \Delta\omega)^{\beta_q}\end{aligned}\tag{16.5}$$

where the frequency deviation $\Delta\omega$ is approximated by filtering and differentiating the bus voltage phase angle θ (see Fig. 16.1). The parameters p^0 and q^0 are the initial active and reactive powers, respectively, computed after the power flow solution and based on the PQ load active and reactive powers p_L^0 and q_L^0 as defined in (16.2), and v^0 and θ^0 are the voltage magnitude and phase angle determined by the power flow analysis.

Frequency dependent loads are initialized after the power flow analysis. A PQ load must be connected to the same bus, and its power and voltage ratings will be inherited by the frequency dependent load. Table 16.3 reports the data format for the component whereas Table 16.4 depicts some typical coefficients for characteristic loads [13].

Frequency dependent loads are defined in the global variable `F1`, which is an instance of the class `FLclass`. Relevant properties are as follows:

1. `con`: frequency dependent load data.
2. `n`: total number of polynomial power loads.
3. `bus`, `vbus`: indexes of voltages of buses to which frequency dependent loads are connected.
4. `a0`: initial bus voltage phase angles.
5. `dw`: indexes of the algebraic variables $\Delta\omega$.
6. `x`: indexes of filter state variables x .
7. `store`: data backup.
8. `u`: connection status.

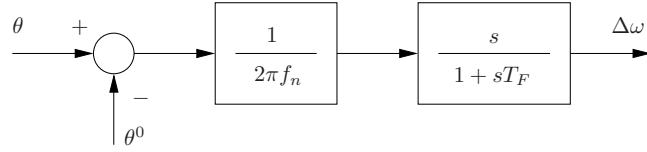


Figure 16.1: Measure of frequency deviation.

Table 16.3: Frequency Dependent Load Data Format (F1.con)

Column	Variable	Description	Unit
1	-	Bus number	int
2	k_p	Active power percentage	%
3	α_p	Active power voltage coefficient	-
4	β_p	Active power frequency coefficient	-
5	k_q	Reactive power percentage	%
6	α_q	Reactive power voltage coefficient	-
7	β_q	Reactive power frequency coefficient	-
8	T_F	Filter time constant	s
† 9	u	Connection status	{1, 0}

16.4 Voltage Dependent Load with Dynamic Tap Changer

Figure 16.2 depicts a simplified model of voltage dependent load with embedded dynamic tap changer.¹ The transformer model consists of an ideal circuit with tap ratio m and the voltage on the secondary winding is modeled as $v_s = v/m$. The voltage control is obtained by means of a quasi-integral anti-windup regulator. The data format is reported in Table 16.5.

The algebraic equations of the component are as follows:

$$\begin{aligned} p &= p^0 \left(\frac{v}{m} \right)^\alpha \\ q &= q^0 \left(\frac{v}{m} \right)^\beta \end{aligned} \quad (16.6)$$

and the differential equation is:

$$\dot{m} = -Hm + K \left(\frac{v}{m} - v_{\text{ref}} \right) \quad (16.7)$$

Voltage dependent loads with embedded dynamic tap changers are defined in the

¹A more detailed model can be obtained using ULTCs (Section 19.1) and voltage dependent loads (Section 16.1).

Table 16.4: Typical load coefficients [13]

Load	α_p	α_q	β_p	β_q
Filament lamp	1.6	0	0	0
Fluorescent lamp	1.2	3.0	-0.1	2.8
Heater	2.0	0	0	0
Induction motor (half load)	0.2	1.6	1.5	-0.3
Induction motor (full load)	0.1	0.6	2.8	1.8
Reduction furnace	1.9	2.1	-0.5	0
Aluminum plant	1.8	2.2	-0.3	0.6

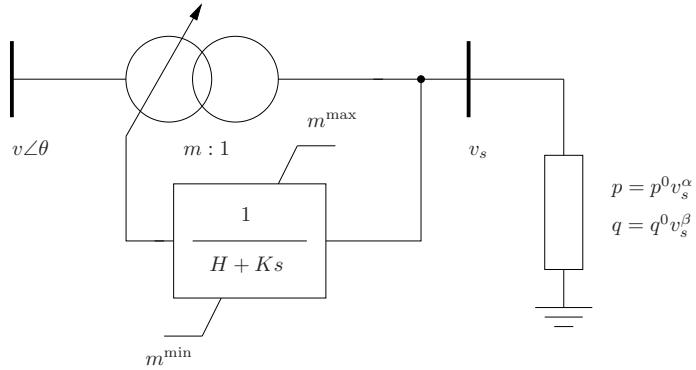


Figure 16.2: Voltage dependent load with dynamic tap changer.

global variable `Tap`, which is an instance of the class `OTPclass`. Relevant properties are as follows:

1. `con`: voltage dependent load with dynamic tap changer data.
2. `bus`, `vbus`: indexes of voltages of buses to which the ULTCs are connected.
3. `n`: total number of loads.
4. `m`: indexes of the state variable m .
5. `store`: data backup.
6. `u`: connection status.

Table 16.5: Voltage Dependent Load with Dynamic Tap Changer Data Format (Tap.con)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	V_n	Voltage rating	kV
4	H	Deviation from integral behavior	p.u.
5	K	Inverse of time constant	1/s
6	m^{\min}	Maximum tap ratio	p.u./p.u.
7	m^{\max}	Minimum tap ratio	p.u./p.u.
8	v_{ref}	Reference voltage	p.u.
9	p^0	Load active power	p.u.
10	q^0	Load reactive power	p.u.
11	α	Voltage exponent (active power)	p.u.
12	β	Voltage exponent (reactive power)	p.u.
† 13	u	Connection status	{0, 1}

16.5 Exponential Recovery Load

An exponential recovery load is included in PSAT based on what was proposed in [55, 66]. Equations are as follows:

$$\begin{aligned}\dot{x}_p &= -x_p/T_p + p_s - p_t \\ p &= x_p/T_p + p_t\end{aligned}\tag{16.8}$$

where p_s and p_t are the static and transient real power absorptions, which depend on the load voltage:

$$\begin{aligned}p_s &= p^0(v/v^0)^{\alpha_s} \\ p_t &= p^0(v/v^0)^{\alpha_t}\end{aligned}\tag{16.9}$$

Similar equations hold for the reactive power:

$$\begin{aligned}\dot{x}_q &= -x_q/T_q + q_s - q_t \\ q &= x_q/T_q + q_t\end{aligned}\tag{16.10}$$

and:

$$\begin{aligned}q_s &= q^0(v/v^0)^{\beta_s} \\ q_t &= q^0(v/v^0)^{\beta_t}\end{aligned}\tag{16.11}$$

The power flow solution and the PQ load data are used for determining the values of p^0 , q^0 and v^0 . In particular, p^0 and q^0 are determined as in (16.2). A PQ load has to be connected to the exponential recovery load bus. The data format is depicted in Table 16.6.

Table 16.6: Exponential Recovery Load Data Format (`Exload.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	k_p	Active power percentage	%
4	k_q	Reactive power percentage	%
5	T_p	Active power time constant	s
6	T_q	Reactive power time constant	s
7	α_s	Static active power exponent	-
8	α_t	Dynamic active power exponent	-
9	β_s	Static reactive power exponent	-
10	β_t	Dynamic reactive power exponent	-
† 11	u	Connection status	{1, 0}

Exponential recovery loads are defined in the global variable `Exload`, which is an instance of the class `ELclass`. Relevant properties are as follows:

1. `con`: Exponential recovery load data.
2. `bus`, `vbus`: indexes of voltages of buses to which exponential recovery loads are connected.
3. `dat`: initial powers and voltages (p^0 , q^0 and v^0).
4. `n`: total number of exponential recovery loads.
5. `xp`: indexes of the state variable x_p .
6. `xq`: indexes of the state variable x_q .
7. `store`: data backup.
8. `u`: connection status.

16.6 Thermostatically Controlled Load

The `Thload` structure defines a dynamic load with temperature control [57]. This component is initialized after the power flow solution and needs a PQ load connected at the same bus. The control diagram block is depicted in Fig. 16.3 which represents the following equations:

$$\begin{aligned}
 \dot{\Theta} &= (\Theta_a - \Theta + K_1 p) / T_1 & (16.12) \\
 \dot{x} &= K_i (\Theta_{\text{ref}} - \Theta) / T_i \\
 g &= K_p (\Theta_{\text{ref}} - \Theta) + x \\
 p &= g v^2 \\
 q &= 0
 \end{aligned}$$

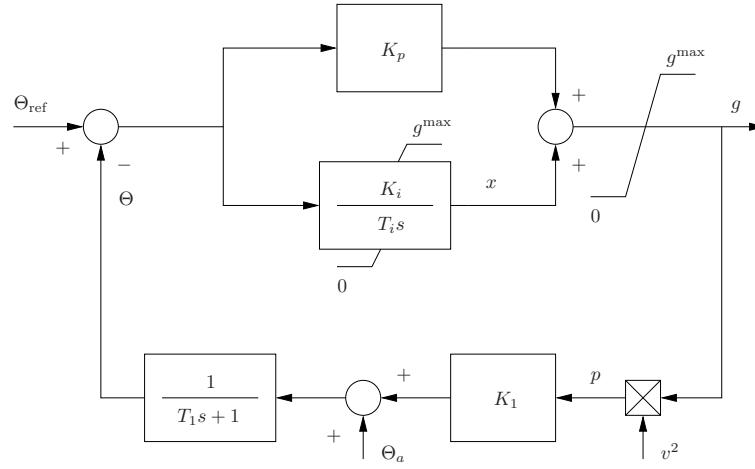


Figure 16.3: Thermostatically controlled load.

where the state variable x undergoes an anti-windup limiter and the algebraic variable G undergoes a windup limiter.

The power flow solution provides the initial voltage v^0 and active power p^0 , which are used for determining the gain K_1 and the maximum conductance g^{\max} , as follows:

$$\begin{aligned} K_1 &= \frac{\Theta_{\text{ref}} - \Theta^0}{p^0} \\ g^{\max} &= K_L g^0 \end{aligned} \quad (16.13)$$

where $g^0 = p^0/v^{0^2}$ and K_L ($K_L > 1$) is the ceiling conductance output ratio.

Exponential recovery loads are defined in the global variable `Thxload`, which is an instance of the class `THclass`. Relevant properties are as follows:

1. `con`: data of the `Thload` components.
2. `bus`, `vbus`: indexes of voltages of buses to which thermostatically controlled loads are connected.
3. `n`: total number of components.
4. `T`: indexes of the state variable Θ .
5. `G`: indexes of the state variable g .
6. `store`: data backup.
7. `u`: connection status.

Table 16.7: Thermostatically Controlled Load Data Format (Thload.con)

Column	Variable	Description	Unit
1	-	Bus number	int
2	k_p	Percentage of active power	%
3	K_p	Gain of proportional controller	p.u./p.u.
4	K_i	Gain of integral controller	p.u./p.u.
5	T_i	Time constant of integral controller	s
6	T_1	Time constant of thermal load	s
7	Θ_a	Ambient temperature	$^{\circ}\text{F}$ or $^{\circ}\text{C}$
8	Θ_{ref}	Reference temperature	$^{\circ}\text{F}$ or $^{\circ}\text{C}$
‡ 9	g^{\max}	Maximum conductance	p.u.
‡ 10	K_1	Active power gain	$(^{\circ}\text{F}$ or $^{\circ}\text{C})/\text{p.u.}$
11	K_L	Ceiling conductance output	p.u./p.u.
‡ 12	u	Connection status	{0, 1}

Table 16.7 depicts the data format for this component. When computing the active power, only PQ components are considered. If no constant PQ load is connected at the same bus of the thermostatically controlled load a warning message is displayed and $p^0 = 0$ is used. The ambient and reference temperatures must be expressed in the same units. The parameters g^{\max} and K_1 are computed and stored in the data matrix during the initialization step.

16.7 Jimma's Load

The Jimma structure defines a load similar to a ZIP model. In addition, the reactive power depends on the time derivative of the bus voltage [64, 133]. This component is initialized after the power flow solution and needs a PQ load connected at the same bus. Since PSAT does not allow defining bus voltages as state variables, the time derivative is defined using a service state variable x and a high-pass filter (see Fig. 16.4). The differential equation is as follows:

$$\begin{aligned} \dot{x} &= (-v/T_f - x)/T_f \\ \frac{dv}{dt} &= x + v/T_f \end{aligned} \tag{16.14}$$

The power injections are defined as follows:

$$p = p_z^0 \left(\frac{v}{v^0} \right)^2 + p_i^0 \left(\frac{v}{v^0} \right) + p_p^0 \tag{16.15}$$

$$q = q_z^0 \left(\frac{v}{v^0} \right)^2 + q_i^0 \left(\frac{v}{v^0} \right) + q_p^0 + K_v \frac{dv}{dt} \tag{16.16}$$

where the load parameters p_z^0 , p_i^0 , p_p^0 , q_z^0 , q_i^0 and q_p^0 are computed as in (16.4).

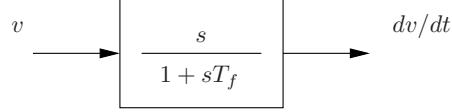


Figure 16.4: Jimma's load.

The power flow solution provides the initial voltage v^0 that is needed for computing the power injections.

Jimmma's loads are defined in the global variable `Jimma`, which is an instance of the class `JIClass`. Relevant properties are as follows:

1. `con`: data of the Jimma components.
2. `dat`: vector of initial voltages v^0 .
3. `bus`, `vbus`: indexes of voltages of buses to which Jimma's loads are connected.
4. `n`: total number of components.
5. `x`: indexes of the state variable x .
6. `store`: data backup.
7. `u`: connection status.

Table 16.8 depicts the data format for this component. When initializing the load, only PQ components are considered. If no constant PQ load is connected at the same bus of the Jimma's load a warning message is displayed and it is assumed that $p = 0$ and $q = 0$.

16.8 Mixed Load

The `Mixload` structure defines a load similar to a frequency dependent load. In addition, the active and the reactive powers depend on the time derivative of the bus voltage. This component is initialized after the power flow solution and needs a PQ load connected at the same bus. Since PSAT do not allow to define bus voltages as state variables, the time derivatives of the voltage magnitude and angle are defined through two service state variables x and y and high-pass filters (see Figs. 16.4 and 16.1). The differential equations are as follows:

$$\begin{aligned} \dot{x} &= (-v/T_{fv} - x)/T_{fv} \\ \Rightarrow \frac{dv}{dt} &= x + v/T_{fv} \end{aligned} \tag{16.17}$$

$$\begin{aligned} \dot{y} &= -\frac{1}{T_{ft}} \left(\frac{1}{2\pi f_n} \frac{1}{T_{ft}} (\theta - \theta^0) + y \right) \\ \Rightarrow \Delta\omega &= y + \frac{1}{2\pi f_n} \frac{1}{T_{ft}} (\theta - \theta^0) \end{aligned} \tag{16.18}$$

Table 16.8: Jimma's Data Format (`Jimma.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rate	MVA
3	V_n	Voltage rate	kV
4	f_n	Frequency rate	Hz
5	T_f	Time constant of the high-pass filter	s
6	k_{pz}	Percentage of active power $\propto v^2$	%
7	k_{pi}	Percentage of active power $\propto v$	%
8	k_{pp}	Percentage of constant active power	%
9	k_{qz}	Percentage of reactive power $\propto v^2$	%
10	k_{qi}	Percentage of reactive power $\propto v$	%
11	k_{qp}	Percentage of constant reactive power	%
12	K_v	Coefficient of the voltage time derivative	1/s
† 13	u	Connection status	{0, 1}

The bus power injections p and q are defined as follows:

$$p = K_{pf} \Delta\omega + p^0 \left[\left(\frac{v}{v^0} \right)^\alpha + T_{pv} \frac{dv}{dt} \right] \quad (16.19)$$

$$q = K_{qf} \Delta\omega + q^0 \left[\left(\frac{v}{v^0} \right)^\beta + T_{qv} \frac{dv}{dt} \right] \quad (16.20)$$

where p^0 and q^0 are computed based on the PQ load active and reactive powers p_L^0 and q_L^0 as defined in (16.2). The power flow solution provides the initial voltage v^0 that is needed for computing the power injections.

Mixed loads are defined in the global variable `Mixload`, which is an instance of the class `MXclass`. Relevant properties are as follows:

1. `con`: data of the `Mixload` components.
2. `dat`: vector of initial voltages v^0 and θ^0 .
3. `bus`, `vbus`: indexes of voltages of buses to which mixed loads are connected.
4. `n`: total number of components.
5. `x`: indexes of the state variable x .
6. `y`: indexes of the state variable y .
7. `store`: data backup.
8. `u`: connection status.

Table 16.9 depicts the data format for this component. When initializing the load, only PQ components are considered. If no constant PQ load is connected at the same bus of the mixed load a warning message is displayed and it is assumed that $p = 0$ and $q = 0$.

Table 16.9: Mixed Data Format (`Mixload.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rate	MVA
3	V_n	Voltage rate	kV
4	f_n	Frequency rate	Hz
5	K_{pf}	Frequency coefficient for the active power	p.u.
6	k_p	Percentage of active power	%
7	α	Voltage exponent for the active power	-
8	T_{pv}	Time constant of dV/dt for the active power	s
9	K_{qf}	Frequency coefficient for the reactive power	p.u.
10	k_q	Percentage of reactive power	%
11	β	Voltage exponent for the reactive power	-
12	T_{qv}	Time constant of dV/dt for the reactive power	s
13	T_{fv}	Time constant of voltage magnitude filter	s
14	T_{ft}	Time constant of voltage angle filter	s
† 15	u	Connection status	{0, 1}

16.9 Remarks on Non-conventional Loads

In general, the non-conventional loads described in this chapter are initialized after the power flow analysis and thus need a PQ load connected at the same bus. Voltage dependent and ZIP loads can be used alone if the “Initialize after power flow” parameter is set to zero. Finally, the voltage dependent load with embedded dynamic tap changer, are included in the power flow analysis and does not need a PQ load for being initialized. Table 16.10 summarizes the usage of non-conventional loads.

The powers used for initializing non-conventional loads are a percentage of the PQ load powers. If the sum of all percentages is 100%, the PQ load is removed from the data. PSAT does not check if the total sum of non-conventional load percentages is greater than 100%. If this is the case, data are inconsistent and the resulting PQ load will show negative powers. Refer to Section 24.3.7 for details on the usage of non-conventional load within SIMULINK models.

Table 16.10: Non-conventional Load Usage

Load	Embedded in power flow	Initialized after power flow	Need of a PQ load	Dynamic
Exload	No	Yes	Yes	Yes
F1	No	Yes	Yes	Yes
Jimma	No	Yes	Yes	Yes
Mixload	No	Yes	Yes	Yes
Mn	Yes if $z = 0$	Yes if $z = 1$	Yes if $z = 1$	No
P1	Yes if $z = 0$	Yes if $z = 1$	Yes if $z = 1$	No
Tap	Yes	No	No	Yes
Thload	No	Yes	Yes	Yes

Chapter 17

Machines

This chapter describes the synchronous generator and the induction motor models.

The dynamic model of induction motors are directly included in the power flow analysis, while synchronous machines are initialized after power flow computations. A PV or a slack generator are required to impose the desired voltage and active power at the synchronous machine bus. Once the power flow solution has been determined, v^0 , θ^0 , p^0 and q^0 at the generation bus are used for initializing the state variables, the field voltage v_f and the mechanical power p_m (see, for example, [111] for details on the initialization of synchronous machines). At the end of the initialization procedure, the PV, PQ or slack generators connected at the machine buses are removed.

Synchronous machine controls such as AVR or Turbine Governors are not included in the machine model. Refer to Chapter 18 for a detailed description of generator control systems.

17.1 Synchronous Machine

The Park-Concordia model is used for synchronous machine equations, whose scheme is depicted in Fig. 17.1. Various simplification levels are applied, from the classical swing equations to an eight order model with field saturation. Fig. 17.2 depicts the d and q -axis block diagrams of stator fluxes for the VI order model while Fig. 17.3 illustrates the field saturation characteristic of the synchronous machine. The link between the network phasors and the machine voltage is as follows:

$$\begin{aligned} v_d &= v \sin(\delta - \theta) \\ v_q &= v \cos(\delta - \theta) \end{aligned} \tag{17.1}$$

The expressions of d and q -axis currents depend on the model, and in general terms are implicitly defined as follows:

$$\begin{aligned} 0 &= g_d(\mathbf{x}, i_d, i_q, v, \theta) \\ 0 &= g_q(\mathbf{x}, i_d, i_q, v, \theta) \end{aligned} \tag{17.2}$$

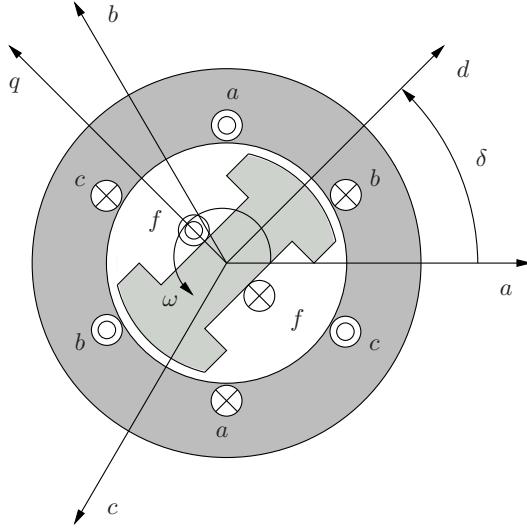


Figure 17.1: Synchronous machine scheme.

Each machine model includes 6 algebraic variables, namely active power p , reactive power q , bus voltage magnitude v and angle θ , mechanical power p_m , and field voltage v_f . Each machine model has also 6 algebraic equations: two are the power injection p and q at the network buses, and the other four equations are:

$$0 = v_d i_d + v_q i_q - p \quad (17.3)$$

$$0 = v_q i_d - v_d i_q - q \quad (17.4)$$

$$0 = p_m^0 - p_m \quad (17.5)$$

$$0 = v_f^0 - v_f \quad (17.6)$$

where v_d and v_q are defined in (17.1), and p_m^0 and v_f^0 are the mechanical power and the field voltage algebraic variables, respectively. In the following models, it is assumed that the speed variations are small, thus, the mechanical power in p.u. is approximately equal to the mechanical torque in p.u. ($p_m = \omega \tau_m \approx \tau_m$).

For models III, IV, V.1, V.2 and VI, the field voltage includes a feedback of the rotor speed and the active power produced by the machine:

$$v_f^* = v_f + K_\omega (\omega - 1) - K_P (p(x, v, \theta) - p^0) \quad (17.7)$$

where p^0 is the initial electric power generated by the machine. Equation (17.7) implements a simple oscillation stabilizer and is implied where the notation v_f^* is used.

Table 17.1 depicts the complete synchronous machine data format. Coefficients γ_p and γ_q are used in case of multiple generators connected to the same bus. In this case the amount of active and reactive power that each machine has to provide

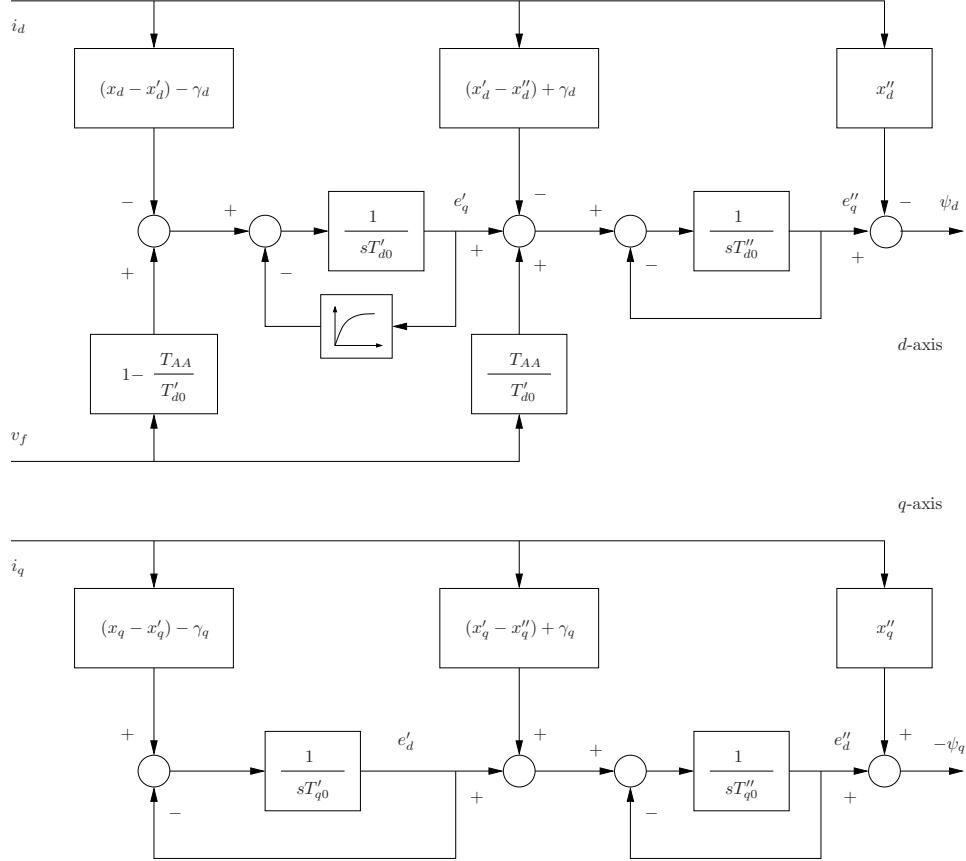


Figure 17.2: *d* and *q*-axis block diagrams of the stator fluxes for the most detailed synchronous machine model. Coefficients γ_d and γ_q are defined as follows:

$$\gamma_d = \frac{T''_{d0}}{T'_{d0}} \frac{x''_d}{x'_d} (x_d - x'_d)$$

$$\gamma_q = \frac{T''_{q0}}{T'_{q0}} \frac{x''_q}{x'_q} (x_q - x'_q)$$

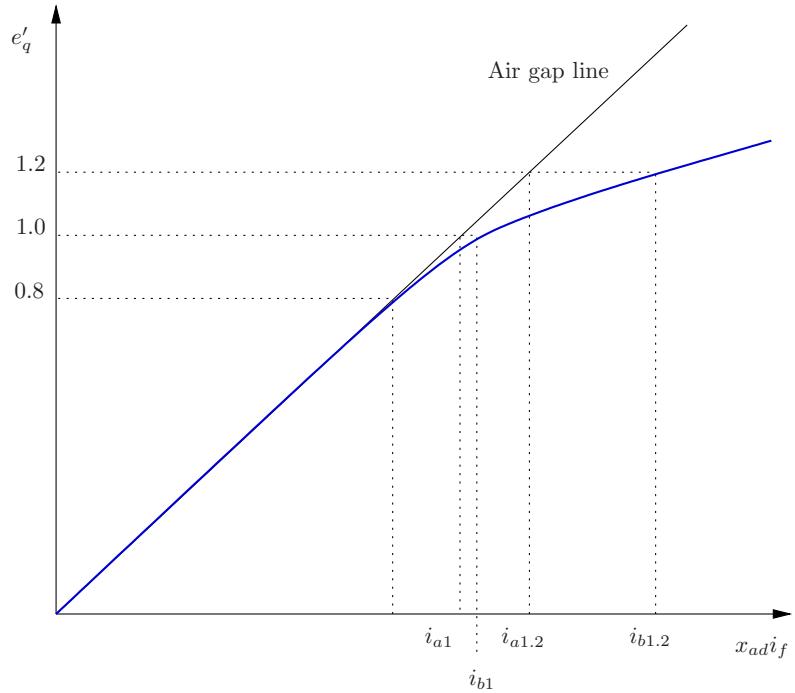


Figure 17.3: Field saturation characteristic of synchronous machines. Saturation factors are defined as follows:

$$S(1.0) = 1 - \frac{i_{a1}}{i_{b1}}$$

$$S(1.2) = 1 - \frac{i_{a1.2}}{i_{b1.2}}$$

Note: the saturation curve is linear for $e'_q < 0.8$, whereas it is approximated by means of a quadratic interpolation for $e'_q \geq 0.8$. $S(1.0) < S(1.2)$ should hold to ensure the right convexity of the saturation curve. Observe that if the saturation factors $S(1.0)$ and/or $S(1.2)$ are given, the d -axis additional leakage time constant is assumed $T_{AA} = 0$.

should be specified. The sum of these coefficients for the machines connected to the same bus has to be one. PSAT does not check the consistency of these coefficients. By default, γ_p and γ_q are set to 1. If the d -axis additional leakage time constant T_{AA} is omitted, it is assumed $T_{AA} = 0$. Table 17.2 depicts a quick reference card for the usage of time constants and reactances within synchronous machine models. When a time constant or a reactance is not used, it can be zero. PSAT checks time constants and reactances when initializing machine state variables. If some time constants and/or reactances are negative or zero, PSAT will automatically set default values and display warning messages.

The synchronous machine is defined in the global variable `Syn`, which is an instance of the class `SYclass`. Relevant properties are as follows:

1. `con`: Synchronous machine data.
2. `n`: total number of synchronous machines.
3. `bus`, `vbus`: indexes of voltages of buses to which synchronous machines are connected.
4. `Id`, `Iq`: direct and quadrature currents.
5. `J11`, `J12`, `J21`, `J22`: Jacobians of algebraic equations.
6. `delta`: rotor angle δ indexes.
7. `omega`: rotor speed ω indexes.
8. `e1q`: q -axis transient voltage e'_q indexes.
9. `e1d`: d -axis transient voltage e'_d indexes.
10. `e2q`: q -axis sub-transient voltage e''_q indexes.
11. `e2d`: d -axis sub-transient voltage e''_d indexes.
12. `psiq`: q -axis flux ψ_q indexes.
13. `psid`: d -axis flux ψ_d indexes.
14. `p`: active power p indexes.
15. `q`: reactive power q indexes.
16. `pm`: mechanical power p_m indexes.
17. `vf`: field voltage v_f indexes.
18. `Pg0`: initial active power generated by the machine.
19. `pm0`: initial mechanical power p_m^0 .
20. `vf0`: initial field voltage v_f^0 .
21. `store`: data backup.
22. `u`: connection status.

Table 17.1: Synchronous Machine Data Format (Syn.con)

Column	Variable	Description	Unit	Model
1	-	Bus number	int	all
2	S_n	Power rating	MVA	all
3	V_n	Voltage rating	kV	all
4	f_n	Frequency rating	Hz	all
5	-	Machine model	-	all
6	x_ℓ	Leakage reactance (<i>not used</i>)	p.u.	all
7	r_a	Armature resistance	p.u.	all
8	x_d	d -axis synchronous reactance	p.u.	all but II
9	x'_d	d -axis transient reactance	p.u.	all
10	x''_d	d -axis sub-transient reactance	p.u.	V.2, VI, VIII
11	T'_{d0}	d -axis open circuit transient time constant	s	all but II
12	T''_{d0}	d -axis open circuit sub-transient time constant	s	V.2, VI, VIII
13	x_q	q -axis synchronous reactance	p.u.	all but II
14	x'_q	q -axis transient reactance	p.u.	IV, V.1, VI, VIII
15	x''_q	q -axis sub-transient reactance	p.u.	V.2, VI, VIII
16	T'_{q0}	q -axis open circuit transient time constant	s	IV, V.1, VI, VIII
17	T''_{q0}	q -axis open circuit sub-transient time constant	s	V.1, V.2, VI, VIII
18	$M = 2H$	Mechanical starting time ($2 \times$ inertia constant)	kWs/kVA	all
19	D	Damping coefficient	-	all
† 20	K_ω	Speed feedback gain	gain	all but V.3 and VIII
† 21	K_P	Active power feedback gain	gain	all but V.3 and VIII
† 22	γ_p	Active power ratio at node	[0,1]	all
† 23	γ_q	Reactive power ratio at node	[0,1]	all
† 24	T_{AA}	d -axis additional leakage time constant	s	V.2, VI, VIII
† 25	$S(1.0)$	First saturation factor	-	all but II and V.3
† 26	$S(1.2)$	Second saturation factor	-	all but II and V.3
† 27	n_{COI}	Center of inertia number	int	all
† 28	u	Connection status	{0, 1}	all

Table 17.2: Reference table for synchronous machine time constants and reactances.

Order	T'_{d0}	T'_{q0}	T''_{d0}	T''_{q0}	x_d	x'_d	x''_d	x_q	x'_q	x''_q
II						✓				
III	✓				✓	✓		✓		
IV	✓	✓			✓	✓		✓	✓	
V.1	✓	✓		✓	✓	✓		✓	✓	
V.2	✓		✓	✓	✓	✓	✓	✓		✓
V.3	✓				✓	✓		✓		
VI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
VIII	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

17.1.1 Order II

This is the classic electro-mechanical model, with constant amplitude e.m.f. e'_q . The state variables are δ and ω . The effects of the leakage reactance and the armature resistance can be included. The differential equations are as follows:

$$\begin{aligned}\dot{\delta} &= \Omega_b(\omega - 1) \\ \dot{\omega} &= (p_m - p_e - D(\omega - 1))/M\end{aligned}\tag{17.8}$$

where Ω_b is the base frequency in rad/s and the electrical power p_e is defined as follows:

$$p_e = (v_q + r_a i_q) i_q + (v_d + r_a i_d) i_d\tag{17.9}$$

Finally, the following relationships between voltages and currents hold:

$$\begin{aligned}0 &= v_q + r_a i_q - e'_q + x'_d i_d \\ 0 &= v_d + r_a i_d - x'_d i_q\end{aligned}\tag{17.10}$$

The q -axis transient voltage e'_q is constant and is stored in the field `vf` of the Syn structure as if it were a field voltage. Automatic Voltage Regulators should not be connected to order II synchronous machines (PSAT will not claim though).

17.1.2 Order III

In this model all the q -axis electromagnetic circuits are neglected, whereas a lead-lag transfer function is used for the d -axis inductance. The three state variables δ , ω and e'_q are described by the following differential equations:

$$\begin{aligned}\dot{\delta} &= \Omega_b(\omega - 1) \\ \dot{\omega} &= (p_m - p_e - D(\omega - 1))/M \\ \dot{e}'_q &= (-f_s(e'_q) - (x_d - x'_d)i_d + v_f^*)/T'_{d0}\end{aligned}\tag{17.11}$$

where the electrical power p_e is (17.9) and the voltage and current link is described by the equations:

$$\begin{aligned} 0 &= v_q + r_a i_q - e'_q + x'_d i_d \\ 0 &= v_d + r_a i_d - x_q i_q \end{aligned} \quad (17.12)$$

This model is the simplest one to which an Automatic Voltage Regulator can be connected.

17.1.3 Order IV

In this model, lead-lag transfer functions are used for modeling the d and q -axis inductances, thus leading to a fourth order system in the state variables δ , ω , e'_q and e'_d :

$$\begin{aligned} \dot{\delta} &= \Omega_b(\omega - 1) \\ \dot{\omega} &= (p_m - p_e - D(\omega - 1))/M \\ \dot{e}'_q &= (-f_s(e'_q) - (x_d - x'_d)i_d + v_f^*)/T'_{d0} \\ \dot{e}'_d &= (-e'_d + (x_q - x'_q)i_q)/T'_{q0} \end{aligned} \quad (17.13)$$

where the electrical power p_e is (17.9) and the voltage and current link is described by the equations:

$$\begin{aligned} 0 &= v_q + r_a i_q - e'_q + x'_d i_d \\ 0 &= v_d + r_a i_d - e'_d - x'_q i_q \end{aligned} \quad (17.14)$$

A similar fourth order model can be formulated using the sub-transient d -axis voltage e''_d instead of e'_d . The corresponding differential equation is:

$$\dot{e}''_d = (-e''_d + (x_q - x'_q)i_q)/T''_{q0} \quad (17.15)$$

17.1.4 Order V, Type 1

In this model, it is assumed:

$$x'_d \approx x''_d \approx x''_q \quad (17.16)$$

which leads to a single d -axis equation for the variable e'_q . The d -axis transient and sub-transient dynamics are used. The model is a fifth order in the variables δ , ω , e'_q , e'_d and e''_d and is described by the equations:

$$\begin{aligned} \dot{\delta} &= \Omega_b(\omega - 1) \\ \dot{\omega} &= (p_m - p_e - D(\omega - 1))/M \\ \dot{e}'_q &= (-f_s(e'_q) - (x_d - x'_d)i_d + v_f^*)/T'_{d0} \\ \dot{e}'_d &= (-e'_d + (x_q - x'_q - \frac{T''_{q0}}{T'_{q0}} \frac{x'_d}{x'_q} (x_q - x'_q))i_q)/T'_{q0} \\ \dot{e}''_d &= (-e''_d + e'_d + (x'_q - x'_d + \frac{T''_{q0}}{T'_{q0}} \frac{x'_d}{x'_q} (x_q - x'_q))i_q)/T''_{q0} \end{aligned} \quad (17.17)$$

where the electrical power p_e is (17.9) and the voltage and current link is as follows:

$$\begin{aligned} 0 &= v_q + r_a i_q - e'_q + x'_d i_d \\ 0 &= v_d + r_a i_d - e''_d - x'_q i_q \end{aligned} \quad (17.18)$$

17.1.5 Order V, Type 2

A second type of fifth order model can be obtained assuming only one additional circuit on the d -axis. The resulting model has five state variables δ , ω , e'_q , e''_q and e''_d and the following differential equations:

$$\begin{aligned} \dot{\delta} &= \Omega_b(\omega - 1) \\ \dot{\omega} &= (p_m - p_e - D(\omega - 1))/M \\ \dot{e}'_q &= (-f_s(e'_q) - (x_d - x'_d - \frac{T''_{d0}}{T'_{d0}} \frac{x''_d}{x'_d} (x_d - x'_d)) i_d + (1 - \frac{T_{AA}}{T'_{d0}}) v_f^*) / T'_{d0} \\ \dot{e}''_q &= (-e''_q + e'_q - (x'_d - x''_d + \frac{T''_{d0}}{T'_{d0}} \frac{x''_d}{x'_d} (x_d - x'_d)) i_d + \frac{T_{AA}}{T'_{d0}} v_f^*) / T''_{d0} \\ \dot{e}''_d &= (-e''_d + (x_q - x''_q) i_q) / T''_{q0} \end{aligned} \quad (17.19)$$

where the electrical power p_e is (17.9) and the voltage and current link is as follows:

$$\begin{aligned} 0 &= v_q + r_a i_q - e''_q + x''_d i_d \\ 0 &= v_d + r_a i_d - e''_d - x''_q i_q \end{aligned} \quad (17.20)$$

17.1.6 Order V, Type 3

This model is the basic model for electromechanical and electromagnetic studies. The effects of speed variation on fluxes are considered along with the field flux dynamic. Thus, the system presents five state variables δ , ω , ψ_f , ψ_q and ψ_d and the differential equations:

$$\begin{aligned} \dot{\delta} &= \Omega_b(\omega - 1) \\ \dot{\omega} &= (p_m - p_e - D(\omega - 1))/M \\ \dot{\psi}_f &= (v_f - e'_q) / T'_{d0} \\ \dot{\psi}_q &= \Omega_b(v_q + r_a i_q - \omega \psi_d) \\ \dot{\psi}_d &= \Omega_b(v_d + r_a i_d + \omega \psi_q) \end{aligned} \quad (17.21)$$

where the electrical power p_e is:

$$p_e = \psi_d i_q - \psi_q i_d \quad (17.22)$$

To complete the model, three algebraic constraints are needed:

$$\begin{aligned} \psi_f &= e'_q - (x_d - x'_d) i_d \\ \psi_d &= e'_q - x_d i_d \\ \psi_q &= -x_q i_q \end{aligned} \quad (17.23)$$

These equations can be rewritten in order to obtain a differential equation for e'_q , thus eliminating from the system the field flux ψ_f :

$$\dot{e}'_q = \frac{x_d}{x'_d} \left(\frac{1}{T'_{d0}} (v_f - e'_q) - \frac{x_d - x'_d}{x_d} \dot{\psi}_d \right) \quad (17.24)$$

The state variables used in PSAT are δ , ω , e'_q , ψ_q and ψ_d . In order to compute correct eigenvalues for the small signal stability analysis, this model should be used in networks where a slack bus is present.

17.1.7 Order VI

The sixth order model is obtained assuming the presence of a field circuit and an additional circuit along the d -axis and two additional circuits along the q -axis. The system has six state variables (δ , ω , e'_q , e'_d , e''_q and e''_d) and the following equations:

$$\begin{aligned} \dot{\delta} &= \Omega_b(\omega - 1) \\ \dot{\omega} &= (p_m - p_e - D(\omega - 1))/M \\ \dot{e}'_q &= (-f_s(e'_q) - (x_d - x'_d - \frac{T''_{d0}}{T'_{d0}} \frac{x''_d}{x'_d} (x_d - x'_d)) i_d + (1 - \frac{T_{AA}}{T'_{d0}}) v_f^*) / T'_{d0} \\ \dot{e}'_d &= (-e'_d + (x_q - x'_q - \frac{T''_{q0}}{T'_{q0}} \frac{x''_q}{x'_q} (x_q - x'_q)) i_q) / T'_{q0} \\ \dot{e}''_q &= (-e''_q + e'_q - (x'_d - x''_d + \frac{T''_{d0}}{T'_{d0}} \frac{x''_d}{x'_d} (x_d - x'_d)) i_d + \frac{T_{AA}}{T'_{d0}} v_f^*) / T''_{d0} \\ \dot{e}''_d &= (-e''_d + e'_d + (x'_q - x''_q + \frac{T''_{q0}}{T'_{q0}} \frac{x''_q}{x'_q} (x_q - x'_q)) i_q) / T''_{q0} \end{aligned} \quad (17.25)$$

where the electrical power p_e is (17.9) and the algebraic constraints are as follows:

$$\begin{aligned} 0 &= v_q + r_a i_q - e''_q + x''_d i_d \\ 0 &= v_d + r_a i_d - e''_d - x''_q i_q \end{aligned} \quad (17.26)$$

This model is basically the same of the VIII order one, but with the assumptions $\dot{\psi}_d = \dot{\psi}_q = 0$, $\omega \psi_d \approx \psi_d$ and $\omega \psi_q \approx \psi_q$.

17.1.8 Order VIII

This model is obtained with the same assumption of model VI, but including electromagnetic flux dynamics. The state variables are δ , ω , e'_q , e'_d , e''_q , e''_d , ψ_d and ψ_q ,

with the following equations:

$$\begin{aligned}
 \dot{\delta} &= \Omega_b(\omega - 1) \\
 \dot{\omega} &= (p_m - p_e - D(\omega - 1))/M \\
 \dot{e}'_q &= (-f_s(e'_q) - (x_d - x'_d - \frac{T''_{d0}}{T'_{d0}} \frac{x''_d}{x'_d} (x_d - x'_d))i_d + (1 - \frac{T_{AA}}{T'_{d0}})v_f)/T'_{d0} \\
 \dot{e}'_d &= (-e'_d + (x_q - x'_q - \frac{T''_{q0}}{T'_{q0}} \frac{x''_q}{x'_q} (x_q - x'_q))i_q)/T'_{q0} \\
 \dot{e}''_q &= (-e''_q + e'_q - (x'_d - x''_d + \frac{T''_{d0}}{T'_{d0}} \frac{x''_d}{x'_d} (x_d - x'_d))i_d + \frac{T_{AA}}{T'_{d0}}v_f)/T''_{d0} \\
 \dot{e}''_d &= (-e''_d + e'_d + (x'_q - x''_q + \frac{T''_{q0}}{T'_{q0}} \frac{x''_q}{x'_q} (x_q - x'_q))i_q)/T''_{q0} \\
 \dot{\psi}_q &= \Omega_b(v_q + r_a i_q - \omega \psi_d) \\
 \dot{\psi}_d &= \Omega_b(v_d + r_a i_d + \omega \psi_q)
 \end{aligned} \tag{17.27}$$

where the electrical power p_e is (17.22). The following algebraic relationship complete the model:

$$\begin{aligned}
 \psi_d &= e''_q - x''_d i_d \\
 \psi_q &= -e''_d - x''_q i_q
 \end{aligned} \tag{17.28}$$

In order to compute correct eigenvalues for the small signal stability analysis, this model should be used in networks where a slack bus is present.

17.1.9 Center of Inertia

In the previous models, the rotor angle and speed are relative to reference angle and speed of a hypothetical machine with infinite inertia, as follows:

$$\dot{\delta} = \Omega_b(\omega - 1) \tag{17.29}$$

In some applications, it is useful to refer machine angles and speeds to the *Center of Inertia* (COI), which is a weighted sum of all machine angles and speeds:

$$\delta_{\text{COI}} = \frac{\sum M\delta}{\sum M} \tag{17.30}$$

$$\omega_{\text{COI}} = \frac{\sum M\omega}{\sum M} \tag{17.31}$$

Then, (17.29) becomes:

$$\dot{\delta} = \Omega_b(\omega - \omega_{\text{COI}}) \tag{17.32}$$

To enable the usage of the COI, one has to set

```
>> Setting.coi = 1;
```

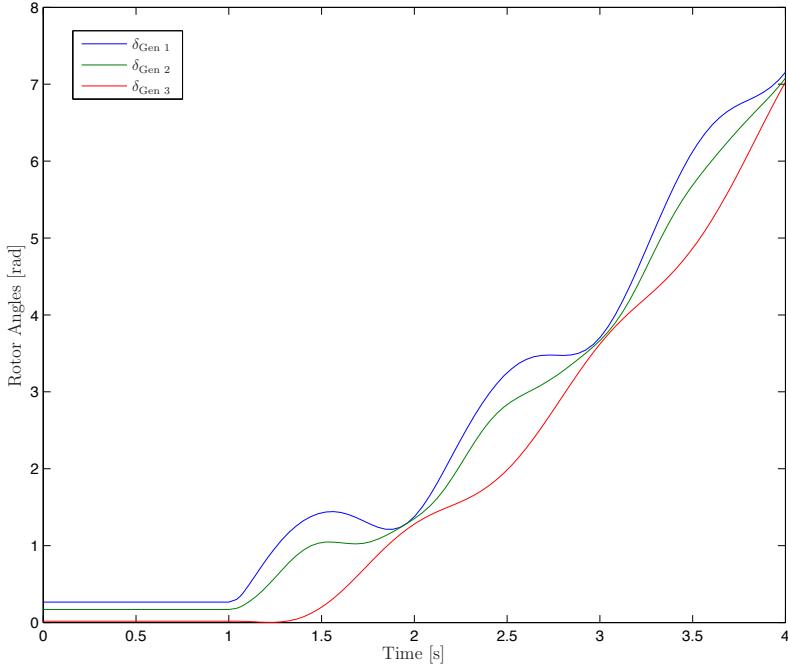


Figure 17.4: Generator rotor angles for the 9-bus test system: ideal synchronous speed reference.

or checking the box “Use Center of Inertia (COI)” in the general settings GUI.

Figures 17.4 and 17.5 show the difference between rotor angles referred to the ideal synchronous machine (default model) and the same angles but using a COI reference. The example represents a three-phase fault at bus 7 for the WSCC 9-bus system. At a first glance, the first plot could induce to think that the system is losing synchronism. Actually, the relative differences among rotor angles remain bounded, thus the system is stable. This conclusion is quite evident if using the COI.

In PSAT, the COI is defined by the global variable `COI`, which is an instance of the class `CIclass`. By default, all machines belong to the same COI. However, PSAT allows defining any number of COIs. To define several COIs, one has to set COI numbers n_{COI} in the definition of synchronous machines (see Table 17.1). Each number defines a COI. Use same COI numbers to group synchronous machines in the same COI.

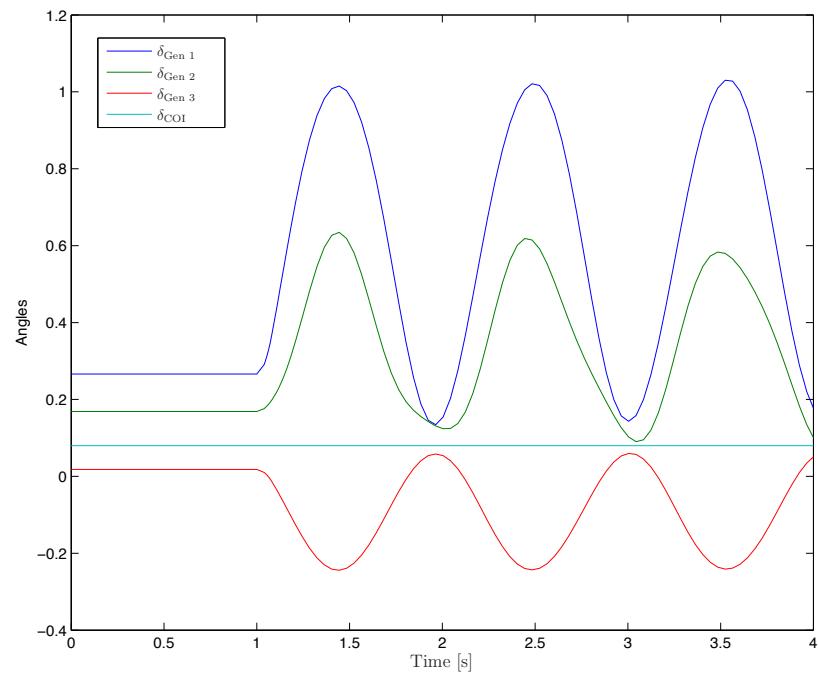


Figure 17.5: Generator rotor angles for the 9-bus test system: COI reference.

17.2 Induction Machine

The models used for the induction motors are defined with an approach similar to what was described for the synchronous machine. Three models are defined for the induction motor. These are pure mechanical model, single cage rotor model, and double cage rotor model. The expression used for the torque/speed characteristic is a composite load model:

$$\tau_m = a + b\omega + c\omega^2 \quad (17.33)$$

and given the relationship between the slip σ and the speed ω in p.u., e.g. $\sigma = 1 - \omega$, the torque/slip characteristic becomes:

$$\tau_m = \alpha + \beta\sigma + \gamma\sigma^2 \quad (17.34)$$

where

$$\begin{aligned} \alpha &= a + b + c \\ \beta &= -b - 2c \\ \gamma &= c \end{aligned}$$

Table 17.3 depicts the data format of the induction machine. The user can decide if connecting the induction machine directly in the power flow ($s_{up} = 0$) or start up the machine at a given time t_{up} ($s_{up} = 1$). If the machine is marked for start up, $\sigma = 1$ for $t \leq t_{up}$.

The induction machine is defined in the global variable `Ind`, which is an instance of the class `IMclass`. Relevant properties are as follows:

1. `con`: induction motor data.
2. `n`: total number of induction motors.
3. `bus`, `vbus`: indexes of voltages of buses to which induction motors are connected.
4. `dat`: induction motor parameters.
5. `slip`: slip σ indexes.
6. `e1r`: real part of 1st cage voltage e'_r indexes.
7. `e1m`: imaginary part of 1st cage voltage e'_m indexes.
8. `e2r`: real part of 2nd cage voltage e''_r indexes.
9. `e2m`: imaginary part of 2nd cage voltage e''_m indexes.
10. `store`: data backup.
11. `u`: connection status.

Following subsections describe the detailed models of the three induction motor models.

Table 17.3: Induction Machine Data Format (`Ind.con`)

Column	Variable	Description	Unit	
1	-	Bus number	int	all
2	S_n	Power rating	MVA	all
3	V_n	Voltage rating	kV	all
4	f_n	Frequency rating	Hz	all
5	-	Model order	int	all
6	s_{up}	Start-up control	{0, 1}	all
7	r_S	Stator resistance	p.u.	all
8	x_S	Stator reactance	p.u.	all
9	r_{R1}	1 st cage rotor resistance	p.u.	all
10	x_{R1}	1 st cage rotor reactance	p.u.	all
11	r_{R2}	2 nd cage rotor resistance	p.u.	V
12	x_{R2}	2 nd cage rotor reactance	p.u.	V
13	x_m	Magnetization reactance	p.u.	all
14	H_m	Inertia constant	kWs/kVA	all
15	a	1 st coeff. of $\tau_m(\omega)$	p.u.	all
16	b	2 nd coeff. of $\tau_m(\omega)$	p.u.	all
17	c	3 rd coeff. of $\tau_m(\omega)$	p.u.	all
18	t_{up}	Start up time	s	all
† 19	\aleph	Allow working as brake	{0, 1}	all
† 20	u	Connection status	{0, 1}	all

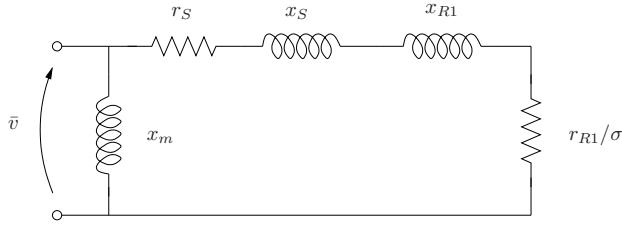


Figure 17.6: Order I induction motor: electrical circuit.

17.2.1 Order I

The electrical circuit used for the first order induction motor is depicted in Fig. 17.6. Only the mechanical state variable is considered, being the circuit in steady-state condition. The differential equation is as follows:

$$\dot{\sigma} = \frac{1}{2H_m} \left(\tau_m(\sigma) - \frac{r_{R1}v^2/\sigma}{(r_s + r_{R1}/\sigma)^2 + (x_s + x_{R1})^2} \right) \quad (17.35)$$

whereas the power injections are:

$$\begin{aligned} p &= -\frac{r_{R1}v^2/\sigma}{(r_s + r_{R1}/\sigma)^2 + (x_s + x_{R1})^2} \\ q &= -\frac{v^2}{x_m} - \frac{(x_s + x_{R1})v^2}{(r_s + r_{R1}/\sigma)^2 + (x_s + x_{R1})^2} \end{aligned} \quad (17.36)$$

17.2.2 Order III (single cage)

The simplified electrical circuit used for the single-cage induction motor is depicted in Fig. 17.7. Equations are formulated in terms of the real (r) and imaginary (m) axis, with respect to the network reference angle. In a synchronously rotating reference frame, the link between the network and the stator machine voltages is as follows:

$$\begin{aligned} v_r &= -v \sin \theta \\ v_m &= v \cos \theta \end{aligned} \quad (17.37)$$

Using the notation of Fig. 17.7, the power absorptions are:

$$\begin{aligned} p &= -(v_r i_r + v_m i_m) \\ q &= -(v_m i_r - v_r i_m) \end{aligned} \quad (17.38)$$

The differential equations in terms of the voltage behind the the stator resistance r_s are:

$$\begin{aligned} \dot{e}'_r &= \Omega_b \sigma e'_m - (e'_r + (x_0 - x')i_m)/T'_0 \\ \dot{e}'_m &= -\Omega_b \sigma e'_r - (e'_m - (x_0 - x')i_r)/T'_0 \end{aligned} \quad (17.39)$$

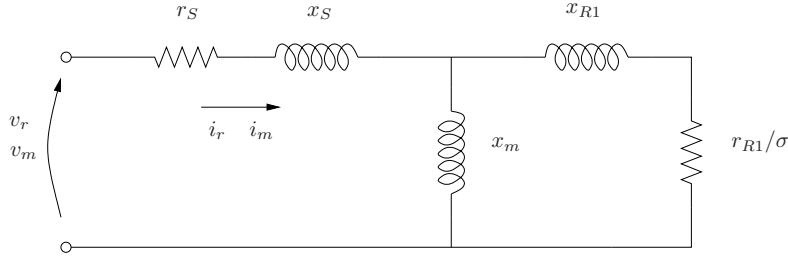


Figure 17.7: Order III induction motor: electrical circuit.

whereas the link between voltages, currents and state variables is as follows:

$$\begin{aligned} v_r - e'_r &= r_S i_r - x' i_m \\ v_m - e'_m &= r_S i_m + x' i_r \end{aligned} \quad (17.40)$$

where x_0 , x' and T_0' can be obtained from the motor parameters:

$$\begin{aligned} x_0 &= x_S + x_m \\ x' &= x_S + \frac{x_{R1} x_m}{x_{R1} + x_m} \\ T_0' &= \frac{x_{R1} + x_m}{\Omega_b r_{R1}} \end{aligned} \quad (17.41)$$

Finally, the mechanical equation is as follows:

$$\dot{\sigma} = (\tau_m(\sigma) - \tau_e)/(2H_m) \quad (17.42)$$

where the electrical torque is:

$$\tau_e \approx e'_r i_r + e'_m i_m \quad (17.43)$$

17.2.3 Order V (double cage)

The electrical circuit for the double-cage induction machine model is depicted in Fig. 17.8. As for the single-cage model, real and imaginary axis are defined with respect to the network reference angle, and (17.37) and (17.38) apply. Two voltages behind the stator resistance r_S model the cage dynamics, as follows:

$$\begin{aligned} \dot{e}'_r &= \Omega_b \sigma e'_m - (e'_r + (x_0 - x') i_m)/T_0' \\ \dot{e}'_m &= -\Omega_b \sigma e'_r - (e'_m - (x_0 - x') i_r)/T_0' \\ \dot{e}''_r &= -\Omega_b \sigma (e'_m - e''_m) + \dot{e}'_r - (e'_r - e''_m - (x' - x'') i_m)/T_0'' \\ \dot{e}''_m &= \Omega_b \sigma (e'_r - e''_r) + \dot{e}'_m - (e'_m - e''_r + (x' - x'') i_r)/T_0'' \end{aligned} \quad (17.44)$$

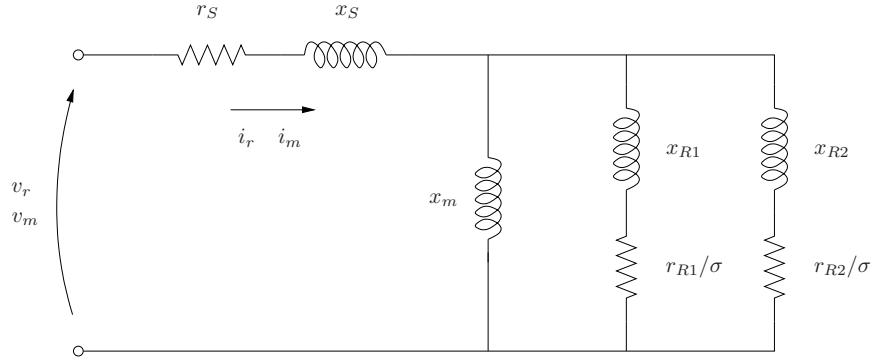


Figure 17.8: Order V induction motor: electrical circuit.

and the links between voltages and currents are:

$$\begin{aligned} v_r - e_r'' &= r_s i_r - x'' i_m \\ v_m - e_m'' &= r_s i_m + x'' i_r \end{aligned} \quad (17.45)$$

where the parameters are determined from the circuit resistances and reactances and are given by equations (17.41) and:

$$\begin{aligned} x'' &= x_s + \frac{x_{R1}x_{R2}x_m}{x_{R1}x_{R2} + x_{R1}x_m + x_{R2}x_m} \\ T_0'' &= \frac{x_{R2} + x_{R1}x_m / (x_{R1} + x_m)}{\Omega_b r_{R2}} \end{aligned} \quad (17.46)$$

The differential equation for the slip is the (17.42), while the electrical torque is defined as follows:

$$\tau_e \approx e_r'' i_r + e_m'' i_m \quad (17.47)$$

Chapter 18

Controls

This chapter describes regulators and controllers included in PSAT. These are: Turbine Governor (TG), Automatic Voltage Regulator (AVR), Power System Stabilizer (PSS), Over Excitation Limiter (OXL), Secondary Voltage Control system which includes Central Area Controllers (CACs) and Cluster Controllers (CCs) for coordinating AVR and SVCs; and Power Oscillation Damper. All regulators are described by means of a set of differential and algebraic equations, as follows:

$$\begin{aligned}\dot{\boldsymbol{x}} &= \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}) \\ \mathbf{0} &= \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u})\end{aligned}\tag{18.1}$$

where \boldsymbol{x} are the controller state variables, \boldsymbol{y} are the algebraic variables, and \boldsymbol{u} are controllable and/or specified signals (e.g., voltage or speed references).

The vector of algebraic variables \boldsymbol{y} includes: system variables (e.g., bus voltage magnitudes); internal variables (e.g., internal regulator input and output signals); and output variables (e.g., synchronous machine field voltages and mechanical torques).

18.1 Turbine Governor

Turbine Governors (TGs) define the primary frequency control of synchronous machines. Figure 18.1 shows the basic functioning of the primary frequency control. In particular, Fig. 18.1.a depicts the linearized control loop that includes the turbine governor transfer function $G(s)$ and a simplified machine model. The transfer function $G(s)$ depends on the type of the turbine and on the control (see for example Figs. 18.2 and 18.3). In steady-state conditions:

$$\lim_{s \rightarrow 0} G(s) = \frac{1}{R}\tag{18.2}$$

thus, one has:

$$\Delta p_m = \frac{1}{R}(\Delta \omega_{\text{ref}} - \Delta \omega)\tag{18.3}$$

where all quantities are in p.u. with respect to machine bases. Figure 18.1.b shows the effect of the droop R on the regulation: (i) $R \neq 0$ is the normal situation; (ii) $R = 0$ implies that $G(s)$ contains a pure integrator and thus a constant frequency control; and (iii) $R \rightarrow \infty$ means constant power control (the primary frequency control loop is open). In general, $R \neq 0$ and $R < \infty$, so that the machine regulates the frequency proportionally to its power rate. Only in islanded systems with one or few machines it makes sense to set $R = 0$. Finally, Fig. 18.1.c shows the effect of the variation of the system frequency $\Delta\omega$ on a multi-machine system. If $\Delta\omega < 0$ (which implies that the power absorbed by the load has increased) then, each machine increase its power production by Δp_{ri} proportionally to $1/R_i$.

The droop R is a measure of the participation of each machine to system losses and load power variations. Thus, one can define the loss participation coefficient γ_i in the static generator models (see Sections 12.4 and 12.5) based on R_i , as follows:

$$\gamma_i = \frac{1/R_i}{\sum_j^n 1/R_j} \quad (18.4)$$

where n is the number of synchronous machines with primary frequency control. For example, if a system has three machines with $1/R_1 = 3\%$, $1/R_2 = 5\%$, and $1/R_3 = 4\%$, then:

$$\begin{aligned}\gamma_1 &= \frac{0.03}{0.03 + 0.05 + 0.04} = 0.25 \\ \gamma_2 &= \frac{0.05}{0.03 + 0.05 + 0.04} = 0.42 \\ \gamma_3 &= \frac{0.04}{0.03 + 0.05 + 0.04} = 0.33\end{aligned}$$

Only the relative values of the coefficients γ_i are relevant, not the absolute ones. Two relevant remarks are as follows:

1. If $R_i \rightarrow \infty$, $\gamma_i = 0$ for the machine i .
2. If $R_i = 0$, $\gamma_i = 1$ for the machine i , being all other loss participation coefficients $\gamma_j = 0$, $\forall j = 1, \dots, n, j \neq i$.

When defining the TG data, the droop R and mechanical power limits must be given in p.u. with respect to the synchronous machine power rating. During initialization, the droops are converted to the system power base, as follows:

$$R_{\text{system}} = \frac{S_{\text{system}}}{S_{\text{machine}}} R_{\text{machine}} \quad (18.5)$$

Mechanical power limits are checked at the initialization step. If a limit is violated, an error message is displayed and the associated state variables are not properly initialized.

Each turbine governor model has two algebraic equations, as follows:

$$0 = p_m - p_m^{\text{syn}} \quad (18.6)$$

$$0 = \omega_{\text{ref}}^0 - \omega_{\text{ref}} \quad (18.7)$$

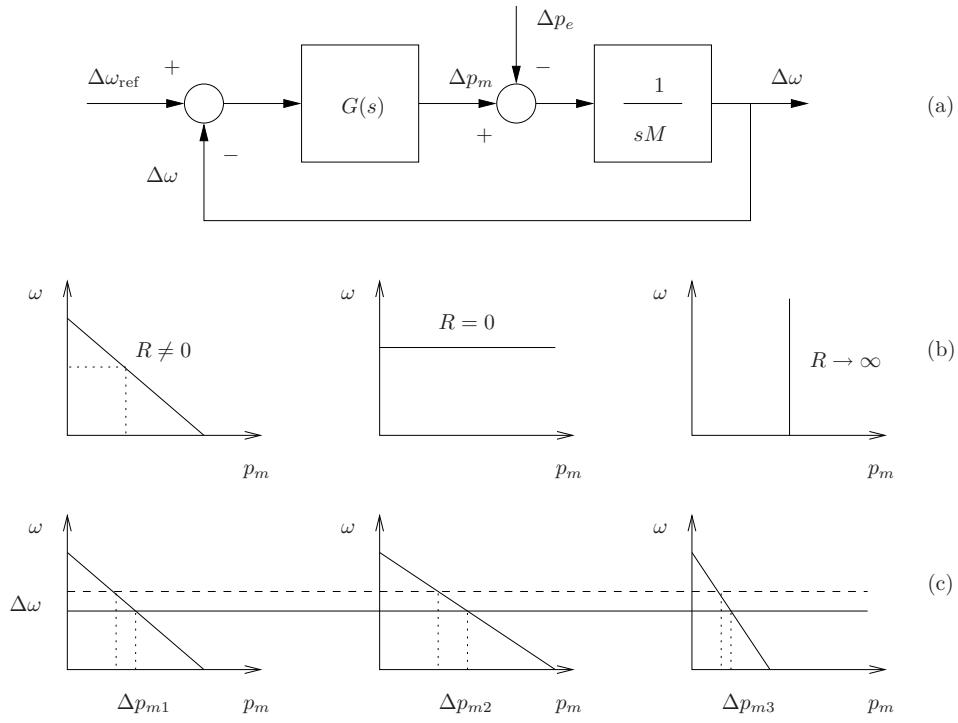


Figure 18.1: Basic functioning of the primary frequency control: (a) linearized control loop; (b) effect of the droop R on the control loop; and (c) Effect of a variation of the rotor speed $\Delta\omega$ on a multi-machine system with $R_i \neq 0$, $\forall i = 1, \dots, n$.

where (18.6) represents the link between the turbine governor and the synchronous machines, being p_m^{syn} the input mechanical power variable used for synchronous machines (see also (17.5)); and (18.7) defines the turbine governor reference rotor speed. The reference signal ω_{ref}^0 can be used, in the future, for AGC controllers.

The TG output is actually the torque, not the mechanical power. However, since we consider small rotor speed variation, $p_m = \omega\tau_m \approx \tau_m$, where τ_m is the neat torque of the machine shaft.

Turbine governors are stored in the global variable `Tg`, which is an instance of the class `TGclass`. Relevant properties are as follows:

1. `con`: Turbine Governor data.
2. `n`: total number of TGs.
3. `syn`: generator numbers.
4. `dat1`: computed parameters for TG type 1.

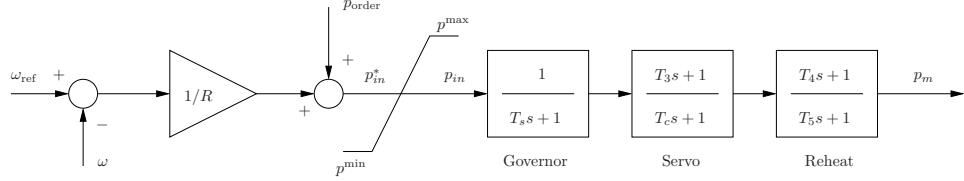


Figure 18.2: Turbine governor type I.

5. **dat2**: computed parameters for TG type 2.
6. **tg**: indexes of state variable x_g .
7. **tg1**: indexes of state variable x_{g1} .
8. **tg2**: indexes of state variable x_{g2} .
9. **tg3**: indexes of state variable x_{g3} .
10. **pm**: indexes of algebraic variable p_m .
11. **wref**: indexes of algebraic variable ω_{ref} .
12. **store**: data backup.
13. **u**: connection status.

18.1.1 TG Type I

The TG type I is depicted in Fig. 18.2 and described by the following equations:

$$\begin{aligned}
 p_{in}^* &= p_{\text{order}} + \frac{1}{R}(\omega_{\text{ref}} - \omega) \\
 p_{in} &= \begin{cases} p_{in}^* & \text{if } p^{\min} \leq p_{in}^* \leq p^{\max} \\ p^{\max} & \text{if } p_{in}^* > p^{\max} \\ p^{\min} & \text{if } p_{in}^* < p^{\min} \end{cases} \\
 \dot{x}_{g1} &= (p_{in} - x_{g1})/T_s \\
 \dot{x}_{g2} &= ((1 - \frac{T_3}{T_c})x_{g1} - x_{g2})/T_c \\
 \dot{x}_{g3} &= ((1 - \frac{T_4}{T_5})(x_{g2} + \frac{T_3}{T_c}x_{g1}) - x_{g3})/T_5 \\
 p_m &= x_{g3} + \frac{T_4}{T_5}(x_{g2} + \frac{T_3}{T_c}x_{g1})
 \end{aligned} \tag{18.8}$$

Table 18.1 depicts the data format of the TG type I.

Table 18.1: Turbine Governor Type I Data Format (Tg.con)

Column	Variable	Description	Unit
1	-	Generator number	int
2	1	Turbine governor type	int
3	ω_{ref}^0	Reference speed	p.u.
4	R	Droop	p.u.
5	p^{\max}	Maximum turbine output	p.u.
6	p^{\min}	Minimum turbine output	p.u.
7	T_s	Governor time constant	s
8	T_c	Servo time constant	s
9	T_3	Transient gain time constant	s
10	T_4	Power fraction time constant	s
11	T_5	Reheat time constant	s
† 12	u	Connection status	{0, 1}

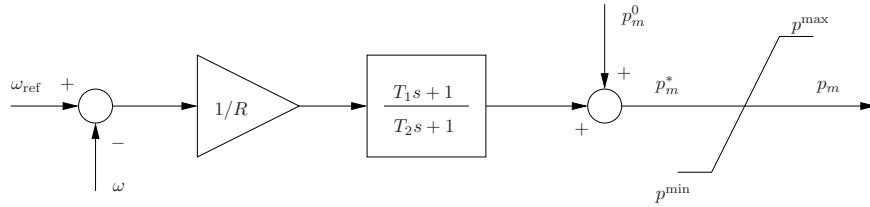


Figure 18.3: Turbine governor type II.

18.1.2 TG Type II

The TG type II is depicted in Fig. 18.3 and described by the following equations:

$$\begin{aligned}\dot{x}_g &= \left(\frac{1}{R}(1 - \frac{T_1}{T_2})(\omega_{\text{ref}} - \omega) - x_g\right)/T_2 & (18.9) \\ p_m^* &= x_g + \frac{1}{R} \frac{T_1}{T_2} (\omega_{\text{ref}} - \omega) + p_m^0 \\ p_m &= \begin{cases} p_m^* & \text{if } p^{\min} \leq p_m^* \leq p^{\max} \\ p^{\max} & \text{if } p_m^* > p^{\max} \\ p^{\min} & \text{if } p_m^* < p^{\min} \end{cases}\end{aligned}$$

Table 18.2 depicts the data format of the TG type II.

Table 18.2: Turbine Governor Type II Data Format (Tg.con)

Column	Variable	Description	Unit
1	-	Generator number	int
2	2	Turbine governor type	int
3	$\omega_{\text{ref}0}$	Reference speed	p.u.
4	R	Droop	p.u.
5	p^{\max}	Maximum turbine output	p.u.
6	p^{\min}	Minimum turbine output	p.u.
7	T_2	Governor time constant	s
8	T_1	Transient gain time constant	s
9	-	<i>Not used</i>	-
10	-	<i>Not used</i>	-
11	-	<i>Not used</i>	-
† 12	u	Connection status	{0, 1}

18.2 Automatic Voltage Regulator

Automatic Voltage Regulators (AVRs) define the primary voltage regulation of synchronous machines. Several AVR models have been proposed and realized in practice. PSAT allows to define three simple different types of AVRs. AVR Type I is a standard Italian regulator (ENEL), whereas AVR Type II is the standard IEEE model 1. AVR Type III is the simplest AVR model which can be used for rough stability evaluations. AVRs are stored in the global variable `Exc`, which is an instance of the class `AVclass`. Relevant properties are as follows:

1. `con`: data chart of the `Exc` components.
2. `n`: total number of automatic voltage regulators.
3. `syn`: generator numbers.
4. `vref`: indexes of algebraic variable v_{ref} .
5. `vref0`: reference voltage $v_{\text{ref}0}$ (*initial value*).
6. `vr1`: indexes of state variable v_{r1} .
7. `vr2`: indexes of state variable v_{r2} .
8. `vr3`: indexes of state variable v_{r3} .
9. `vm`: indexes of state variable v_m .
10. `vf`: indexes of state variable v_f .
11. `store`: data backup.
12. `u`: connection status.

The reference voltages v_{ref} are initialized after the power flow computations. Limits are checked at the initialization step. In case of violation, a warning message is displayed and AVR state variables are not correctly initialized.

Each AVR model has two algebraic equations, as follows:

$$0 = v_f - v_f^{\text{syn}} \quad (18.10)$$

$$0 = v_{\text{ref}}^0 - v_{\text{ref}} \quad (18.11)$$

where (18.10) represents the link in between the AVR and the synchronous machines, being v_f^{syn} the algebraic variable that defines the synchronous machine field voltage (see also (17.6)). Equation (18.11) defines the AVR reference voltage.

18.2.1 AVR Type I

The AVR Type I is depicted in Figs. 18.4 and 18.5 and is described by the following equations:

$$\dot{v}_m = (v - v_m)/T_r \quad (18.12)$$

$$\dot{v}_{r1} = (K_0(1 - \frac{T_2}{T_1})(v_{\text{ref}} - v_m) - v_{r1})/T_1 \quad (18.13)$$

$$\dot{v}_{r2} = ((1 - \frac{T_4}{T_3})(v_{r1} + K_0 \frac{T_2}{T_1}(v_{\text{ref}} - v_m)) - v_{r2})/T_3 \quad (18.14)$$

$$v_r^* = v_{r2} + \frac{T_4}{T_3}(v_{r1} + K_0 \frac{T_2}{T_1}(v_{\text{ref}} - v_m)) \quad (18.15)$$

$$v_r = \begin{cases} v_r^* & \text{if } v_r^{\min} \leq v_r^* \leq v_r^{\max}, \\ v_r^{\max} & \text{if } v_r^* > v_r^{\max}, \\ v_r^{\min} & \text{if } v_r^* < v_r^{\min}. \end{cases} \quad (18.16)$$

$$\dot{v}_f = -(v_f(1 + S_e(v_f)) - v_r)/T_e \quad (18.17)$$

where v is the generator terminal voltage and S_e is the ceiling function, as follows:¹

$$S_e(v_f) = A_e e^{B_e |v_f|} \quad (18.18)$$

For high values of the gain K_0 , the state variable v_{r2} takes also high values. To keep the values of v_{r1} and v_{r2} comparable, the following variable change is used:

$$\tilde{v}_{r2} = \frac{v_{r2}}{K_0} \quad (18.19)$$

¹The coefficients A_e and B_e can be determined by measuring two points of the ceiling function S_e . Typically, one knows the values S_e^{\max} and $S_e^{0.75\cdot\max}$ that correspond to the field voltages v_f^{\max} and $0.75 \cdot v_f^{\max}$, respectively. To compute A_e and B_e , one has to solve the following system:

$$\begin{aligned} 0 &= -(1 + S_e^{\max})v_f^{\max} + v_r^{\max} \\ S_e^{\max} &= A_e e^{B_e v_f^{\max}} \\ S_e^{0.75\cdot\max} &= A_e e^{B_e \cdot 0.75 \cdot v_f^{\max}} \end{aligned}$$

where S_e^{\max} , $S_e^{0.75\cdot\max}$, v_f^{\max} , and v_r^{\max} are given values.

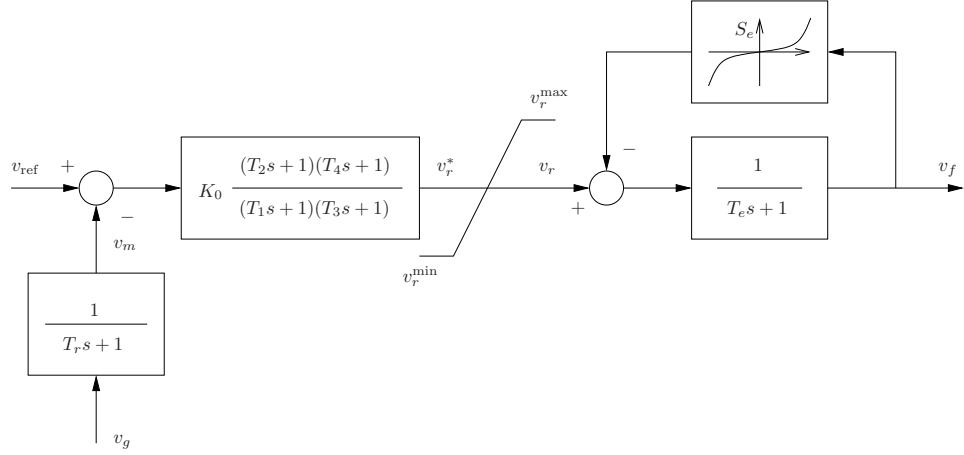


Figure 18.4: Exciter Type I.

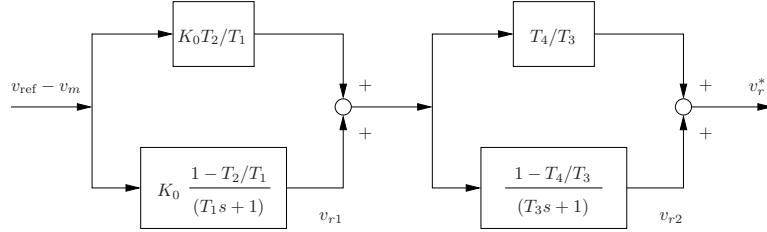


Figure 18.5: Detail of the double lead-lag block of Exciter Type I.

Hence, (18.14) and (18.15) are rewritten as follows:

$$\dot{\tilde{v}}_{r2} = ((1 - \frac{T_4}{T_3})(v_{r1} + K_0 \frac{T_2}{T_1}(v_{ref} - v_m)) - K_0 \tilde{v}_{r2}) / (K_0 T_3) \quad (18.20)$$

$$v_r^* = K_0 \tilde{v}_{r2} + \frac{T_4}{T_3}(v_{r1} + K_0 \frac{T_2}{T_1}(v_{ref} - v_m)) \quad (18.21)$$

Table 18.3 depicts the data format of AVR Type I.

18.2.2 AVR Type II

The AVR Type II is depicted in Fig. 18.6 and described by the following equations:

$$\dot{v}_m = (v - v_m) / T_r \quad (18.22)$$

$$\dot{v}_{r1} = (K_a(v_{ref} - v_m - v_{r2} - \frac{K_f}{T_f}v_f) - v_{r1}) / T_a$$

Table 18.3: Exciter Type I Data Format (Exc.con)

Column	Variable	Description	Unit
1	-	Generator number	int
2	1	Exciter type	int
3	v_r^{\max}	Maximum regulator voltage	p.u.
4	v_r^{\min}	Minimum regulator voltage	p.u.
5	K_0	Regulator gain	p.u./p.u.
6	T_1	1 st pole	s
7	T_2	1 st zero	s
8	T_3	2 nd pole	s
9	T_4	2 nd zero	s
10	T_e	Field circuit time constant	s
11	T_r	Measurement time constant	s
12	A_e	1 st ceiling coefficient	-
13	B_e	2 nd ceiling coefficient	-
† 14	u	Connection status	{0, 1}

$$v_r = \begin{cases} v_{r1} & \text{if } v_r^{\min} \leq v_{r1} \leq v_r^{\max}, \\ v_r^{\max} & \text{if } v_{r1} > v_r^{\max}, \\ v_r^{\min} & \text{if } v_{r1} < v_r^{\min}. \end{cases}$$

$$\dot{v}_{r2} = -\left(\frac{K_f}{T_f}v_f + v_{r2}\right)/T_f$$

$$\dot{v}_f = -(v_f(K_e + S_e(v_f)) - v_r)/T_e$$

where v is the generator terminal voltage and the ceiling function S_e is defined in (18.18). The amplifier block is subjected to an anti-windup limit. Table 18.4 depicts the data format of AVR Type II.

18.2.3 AVR Type III

The AVR Type III is depicted in Fig. 18.7 and described by the following equations:

$$\dot{v}_m = (v - v_m)/T_r \quad (18.23)$$

$$\dot{v}_r = (K_0(1 - \frac{T_1}{T_2})(v_{\text{ref}} - v_m) - v_r)/T_2$$

$$\dot{v}_f = ((v_r + K_0 \frac{T_1}{T_2}(v_{\text{ref}} - v_m) + v_f^0)(1 + s_0(\frac{v}{v^0} - 1)) - v_f)/T_e$$

Where v is the generator terminal voltage. The initial field voltage v_f^0 and bus voltage v^0 are set at the initialization step. The field voltage v_f is subjected to an anti-windup limiter. Table 18.5 depicts the data format of AVR Type III. If s_0 is

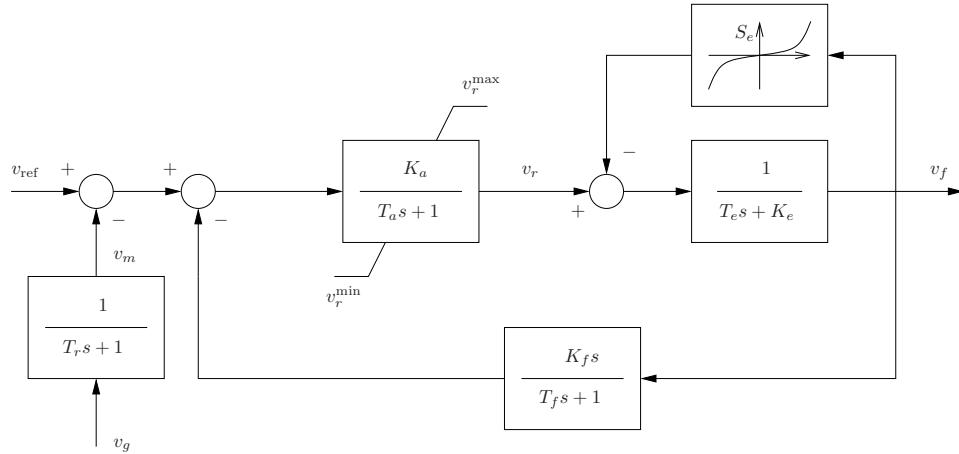


Figure 18.6: Exciter Type II.

Table 18.4: Exciter Type II Data Format (Exc.con)

Column	Variable	Description	Unit
1	-	Generator number	int
2	2	Exciter type	int
3	v_r^{\max}	Maximum regulator voltage	p.u.
4	v_r^{\min}	Minimum regulator voltage	p.u.
5	K_a	Amplifier gain	p.u./p.u.
6	T_a	Amplifier time constant	s
7	K_f	Stabilizer gain	p.u./p.u.
8	T_f	Stabilizer time constant	s
9	K_e	Field circuit integral deviation	p.u./p.u.
10	T_e	Field circuit time constant	s
11	T_r	Measurement time constant	s
12	A_e	1 st ceiling coefficient	-
13	B_e	2 nd ceiling coefficient	-
† 14	u	Connection status	{0, 1}

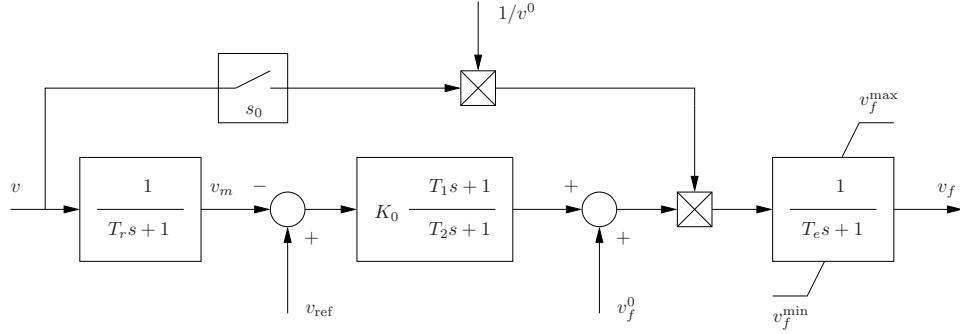


Figure 18.7: Exciter Type III.

set to 1, the signal v/v^0 is enabled. After the initialization of the AVR, the value of s_0 is set to v^0 .

18.3 Power System Stabilizer

Power System Stabilizers (PSSs) are typically used for damping power system oscillations and many different models have been proposed in the literature. In addition to the simple PSS embedded in the synchronous machine equations (models III, IV, V.1, V.2 and VI), five models of PSS are implemented in PSAT.

All models accept as input signals the rotor speed ω , the active power p_G and the bus voltage magnitude v of the generator to which the PSS is connected through the automatic voltage regulator. The PSS output signal is the state variable v_s , which modifies the reference voltage v_{ref} of the AVR. The output signal v_s is subjected to an anti-windup limiter and its dynamic is given by a small time constant $T_\epsilon = 0.001$ s.² Note that PSSs cannot be used with order II generators.

Each PSS model has two algebraic equations, as follows:

$$0 = g_s(\mathbf{x}, \mathbf{y}) - v_s \quad (18.24)$$

$$0 = v_{\text{ref}}^0 - v_{\text{ref}} + v_s \quad (18.25)$$

where (18.24) defines the PSS signal v_s , and (18.25) sums the signal v_s to the AVR reference voltage (see also (18.11)).

PSSs are stored in the global variable `PSS`, which is an instance of the class `PSclass`. Relevant properties are as follows:

1. `con`: PSS data.
2. `n`: total number of PSSs.
3. `bus`: bus numbers.

²Observe that T_ϵ is not defined by the user. However it can be changed directly in the function `fm_pss.m`

Table 18.5: Exciter Type III Data Format (Exc.con)

Column	Variable	Description	Unit
1	-	Generator number	int
2	3	Exciter type	int
3	v_f^{\max}	Maximum field voltage	p.u.
4	v_f^{\min}	Minimum field voltage	p.u.
5	K_0	Regulator gain	p.u./p.u.
6	T_2	Regulator pole	s
7	T_1	Regulator zero	s
† 8	v_{f0}	Field voltage offset	p.u.
9	s_0	Bus voltage signal	{0, 1}
10	T_e	Field circuit time constant	s
11	T_r	Measurement time constant	s
12	-	<i>Not used</i>	-
13	-	<i>Not used</i>	-
† 14	u	Connection status	{0, 1}

4. **syn**: synchronous machine numbers.
5. **exc**: automatic voltage regulator numbers.
6. **v1**: indexes of the state variable v_1 .
7. **v2**: indexes of the state variable v_2 .
8. **v3**: indexes of the state variable v_3 .
9. **va**: indexes of the state variable v_a .
10. **vss**: indexes of the algebraic variable v_s .
11. **vref**: indexes of the algebraic variable v_{ref} .
12. **s1**: current status of switches s_1 .
13. **store**: data backup.
14. **u**: connection status.

The complete PSS data format is depicted in Table 18.6.

18.3.1 Type I

PSS Type I is depicted in Fig. 18.8, and is described by the following differential equation:

$$\begin{aligned}\dot{v}_1 &= -(K_w\omega + K_p p + K_v v + v_1)/T_w \\ v_s &= K_w\omega + K_p p + K_v v + v_1\end{aligned}\tag{18.26}$$

Table 18.6: Power System Stabilizer Data Format (`Pss.con`)

Column	Variable	Description	Unit	
1	-	AVR number	int	all
2	-	PSS model	int	all
3	-	PSS input signal $1 \Rightarrow \omega, 2 \Rightarrow p, 3 \Rightarrow v$	int	II, III, IV, V
4	v_s^{\max}	Max stabilizer output signal	p.u.	all
5	v_s^{\min}	Min stabilizer output signal	p.u.	all
6	K_w	Stabilizer gain (used for ω in model I)	p.u./p.u.	all
7	T_w	Wash-out time constant	s	all
8	T_1	First stabilizer time constant	s	II, III, IV, V
9	T_2	Second stabilizer time constant	s	II, III, IV, V
10	T_3	Third stabilizer time constant	s	II, III, IV, V
11	T_4	Fourth stabilizer time constant	s	II, III, IV, V
12	K_a	Gain for additional signal	p.u./p.u.	IV, V
13	T_a	Time constant for additional signal	s	IV, V
14	K_p	Gain for active power	p.u./p.u.	I
15	K_v	Gain for bus voltage magnitude	p.u./p.u.	I
16	v_a^{\max}	Max additional signal (anti-windup)	p.u.	IV, V
17	\tilde{v}_a^{\max}	Max additional signal (windup)	p.u.	IV, V
18	\tilde{v}_s^{\max}	Max output signal (before adding v_a)	p.u.	IV, V
19	\tilde{v}_s^{\min}	Min output signal (before adding v_a)	p.u.	IV, V
20	e_{thr}	Field voltage threshold	p.u.	IV, V
21	ω_{thr}	Rotor speed threshold	p.u.	IV, V
22	s_2	Allow for switch S2	Boolean	IV, V
† 23	u	Connection status	{0, 1}	all

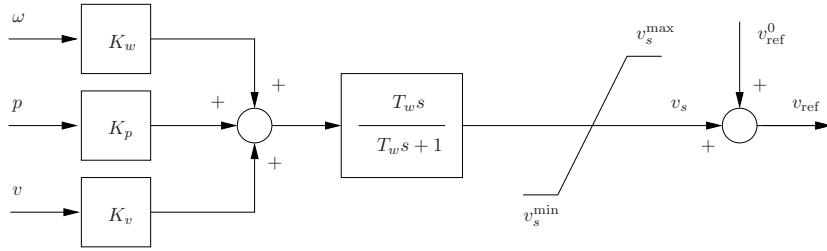


Figure 18.8: Power system stabilizer Type I.

where ω , p and v are the rotor speed, the active power and the voltage magnitude of the generator to which the PSS is connected through the AVR.

18.3.2 Type II

The PSS Type II is depicted in Fig. 18.9, and is described by the equations:

$$\begin{aligned}\dot{v}_1 &= -(K_w v_{SI} + v_1)/T_w \\ \dot{v}_2 &= ((1 - \frac{T_1}{T_2})(K_w v_{SI} + v_1) - v_2)/T_2 \\ \dot{v}_3 &= ((1 - \frac{T_3}{T_4})(v_2 + (\frac{T_1}{T_2}(K_w v_{SI} + v_1))) - v_3)/T_4 \\ v_s &= v_3 + \frac{T_3}{T_4}(v_2 + \frac{T_1}{T_2}(K_w v_{SI} + v_1))\end{aligned}\tag{18.27}$$

Observe that (18.27) are an arbitrary choice of DAE that describes the control blocks depicted in Fig. 18.9. Another possible formulation is as follows:

$$\begin{aligned}\dot{v}_1 &= (K_w v_{SI} - v_1)/T_w \\ \dot{v}_2 &= (K_w v_{SI} - v_1 - v_2)/T_2 \\ \dot{v}_3 &= ((1 - \frac{T_1}{T_2})v_2 + \frac{T_1}{T_2}(K_w v_{SI} - v_1) - v_3)/T_4 \\ v_s &= (1 - \frac{T_3}{T_4})v_3 + \frac{T_3}{T_4}((1 - \frac{T_1}{T_2})v_2 + \frac{T_1}{T_2}(K_w v_{SI} - v_1))\end{aligned}\tag{18.28}$$

The two sets of DAE (18.27) and (18.28) have same dynamic response and stability properties, but the values of state variables are different.

18.3.3 Type III

The PSS Type III is depicted in Fig. 18.10, and is described by the equations:

$$\begin{aligned}\dot{v}_1 &= -(K_w v_{SI} + v_1)/T_w \\ \dot{v}_2 &= v_3\end{aligned}\tag{18.29}$$

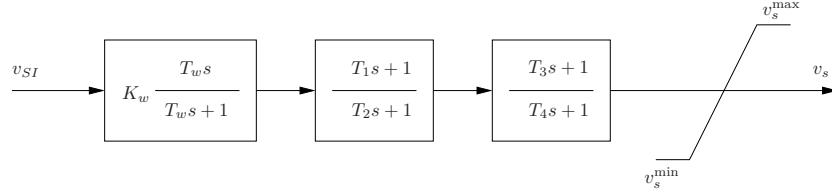


Figure 18.9: Power system stabilizer Type II.

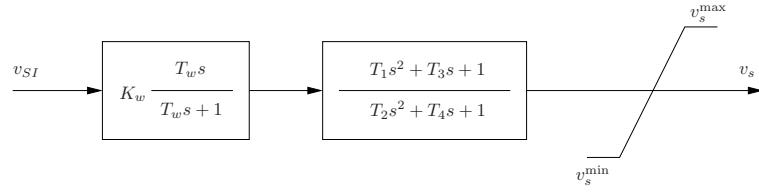


Figure 18.10: Power system stabilizer Type III.

$$\begin{aligned}\dot{v}_3 &= (K_w v_{SI} + v_1 - T_4 v_3 - v_2) / T_2 \\ v_s &= v_2 + \frac{T_1}{T_2} (K_w v_{SI} + v_1) + (T_3 - \frac{T_1}{T_2} T_4) v_3 + (1 - \frac{T_1}{T_2}) v_2\end{aligned}$$

18.3.4 Type IV and V

PSS Type IV and V are a slight variation of Type II and III respectively. The block diagrams are depicted in Figs. 18.11 and 18.12. The additional signal v_a is generally disabled, being the switch $S1$ open. $S1$ closes if the machine field voltage is lower than a threshold value $v_f < e_{thr}$ and remains closed even after $v_f \geq e_{thr}$. $S1$ opens if the rotor speed is lower than a threshold value $\omega < \omega_{thr}$. It is possible to enable the action of a second switch $S2$ after the lag block of the additional signal v_a . If $S2$ is enabled, it stays generally open. $S2$ closes when the rotor speed deviation $\Delta\omega < 0$ and remains closed even after $\Delta\omega \geq 0$.

18.4 Over Excitation Limiter

Over excitation limiters (OXLs) provide an additional signal v_{OXL} to the reference voltage v_{ref}^0 of automatic voltage regulators (AVRs). The OXL is modeled as a pure integrator, with anti-windup hard limits (see Fig. 18.13). This regulator is generally sleeping, i.e. $v_{OXL} = 0$, unless the field current is greater than its thermal limit ($i_f > i_f^{lim}$). It is implicitly assumed that at the initial condition given by the power flow solution, all $i_f \leq i_f^{lim}$, thus leading to $v_{OXL} = 0$ at $t = 0$. If the field

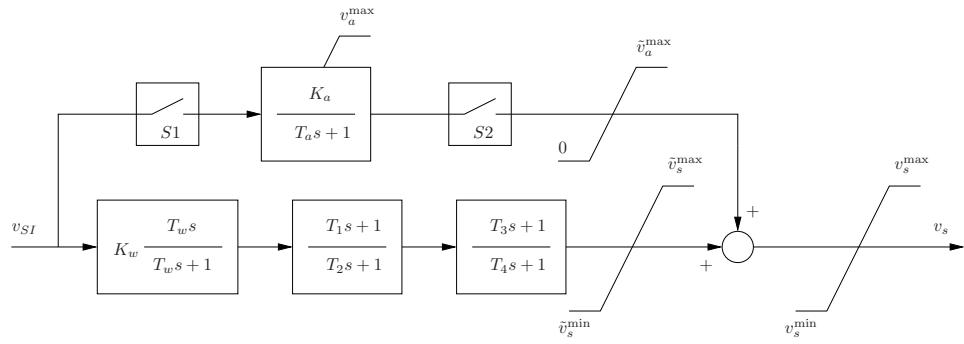


Figure 18.11: Power system stabilizer Type IV.

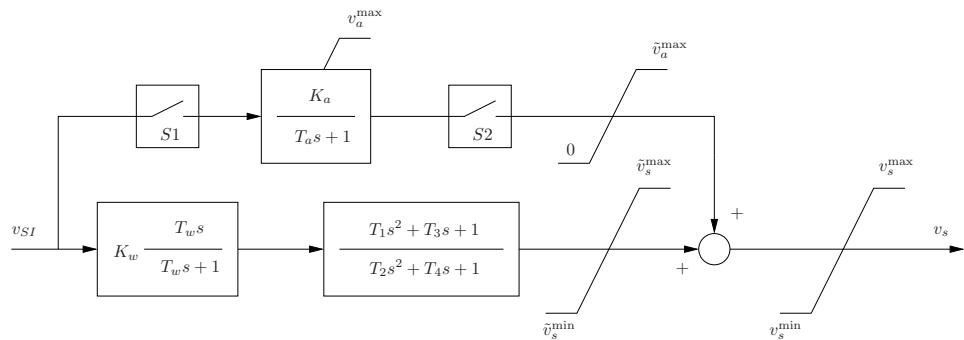


Figure 18.12: Power system stabilizer Type V.

current exceeds its limits, a warning message is shown and the initialization is not correctly completed.

The differential equation for the OXL is as follows:

$$\begin{aligned}\dot{v}_{\text{OXL}} &= (i_f - i_f^{\text{lim}})/T_0 && \text{if } i_f > i_f^{\text{lim}} \\ \dot{v}_{\text{OXL}} &= 0 && \text{if } i_f \leq i_f^{\text{lim}}\end{aligned}\quad (18.30)$$

Each OXL model has also two algebraic equations, as follows:

$$0 = \sqrt{(v + \gamma_q)^2 + p^2} + \left(\frac{x_d}{x_q} + 1\right) \frac{\gamma_q(v + \gamma_q) + \gamma_p}{\sqrt{(v_g + \gamma_q)^2 + p^2}} - i_f \quad (18.31)$$

$$0 = v_{\text{ref}}^0 - v_{\text{ref}} + v_{\text{OXL}} \quad (18.32)$$

where

$$\begin{aligned}\gamma_p &= x_q p / v \\ \gamma_q &= x_q q / v\end{aligned}$$

and v is the voltage at the generator bus, and p and q are the active and the reactive power of the generator, respectively. Equation (18.31) approximates the synchronous machine field current i_f , and (18.32) sums the signal v_{OXL} to the AVR reference voltage (see also (18.11)).

Observe that the definition of the current limiter needs the values of the reactances x_d and x_q of the generator at which the OXL is connected through the AVR. These values can be automatically grabbed from the synchronous machine data or set by the user along with the other data, as illustrated in Table 18.7.

OXLs are stored in the global variable `0x1`, which is an instance of the class `OXclass`. Relevant properties are as follows:

1. `con`: data chart of the `0x1` components.
2. `n`: total number of over excitation limiters.
3. `exc`: index of AVR to which the OXL is connected.
4. `syn`: index of synchronous machine to which the OXL is connected through the AVR.
5. `bus`: index of bus at which the generators `0x1.syn` are connected.
6. `v`: indexes of the state variable v_{OXL} .
7. `If`: indexes of the algebraic variables i_f .
8. `vref`: indexes of the algebraic variables v_{ref} .
9. `store`: data backup.
10. `u`: connection status.

The output signal v_{OXL} is added to the reference voltage v_{ref}^0 of the AVR to which the OXL is connected. If no value is set for T_0 , the default value ($T_0 = 10\text{s}$) will be used.

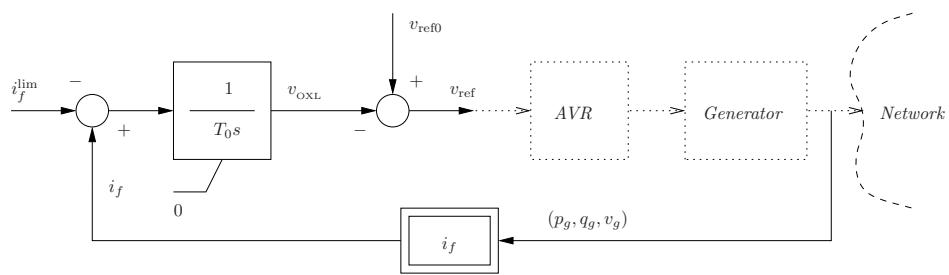


Figure 18.13: Over excitation limiter.

Table 18.7: Over Excitation Limiter Data Format (0x1.con)

Column	Variable	Description	Unit
1	-	AVR number	int
2	T_0	Integrator time constant	s
3	-	Use estimated generator reactances	{0, 1}
4	x_d	d -axis estimated generator reactance	p.u.
5	x_q	q -axis estimated generator reactance	p.u.
6	i_f^{lim}	Maximum field current	p.u.
7	$v_{\text{OXL}}^{\text{max}}$	Maximum output signal	p.u.
† 8	u	Connection status	{0, 1}

18.5 Secondary Voltage Control

A Secondary Voltage Control is included in PSAT by means of a Central Area Controller (CAC) which controls the voltage at a pilot bus, and Cluster Controllers (CC), which compare the CAC signal with the reactive power generated by synchronous machines and/or SVCs and modify the reference voltages of AVR and SVCs.³ Figure 18.14 depicts the secondary voltage control scheme.

CAC equations are as follows:

$$\begin{aligned}\dot{q}_1 &= K_I(v_P^{\text{ref}} - v_P) \\ q &= q_1 + K_P(v_P^{\text{ref}} - v_P)\end{aligned}\quad (18.33)$$

whereas the CC equations are:

$$\begin{aligned}\dot{v}_{s,g} &= \frac{1}{T_g}(x_t + x_{eq})(q_{r,g}q - q_g) \\ \dot{v}_{s,svc} &= \frac{1}{T_{\text{svc}}}x_{eq}(q_{r,\text{svc}}q - q_{\text{svc}})\end{aligned}\quad (18.34)$$

where $v_{s,g}$ and $v_{s,svc}$ are the output signals of CCs for AVR and SVC regulators respectively, x_t is the reactance of the transformer connected to the generator and x_{eq} is an equivalent reactance computed considering the pilot bus and the generator or the SVC bus. CAC and CC integrators are subjected to anti-windup limiters.

Each CAC has to be connected at least to one CC. There is no limitation in the number of CC connected to a CAC.

Central Area and Cluster Controllers are stored in the global variables `CAC` and `Cluster`, respectively, which are instances of the classes `CCclass` and `CLclass`, respectively. Relevant properties are as follows:

Central Area Controller Data

1. `con`: Central Area Controller data
2. `n`: total number of CAC
3. `bus`: indexes of pilot buses
4. `q1`: indexes of the state variable q_1
5. `store`: data backup.
6. `u`: connection status

³These models were realized in collaboration with Dr. Sameh Kamel Mena Kodsi, at University of Waterloo.

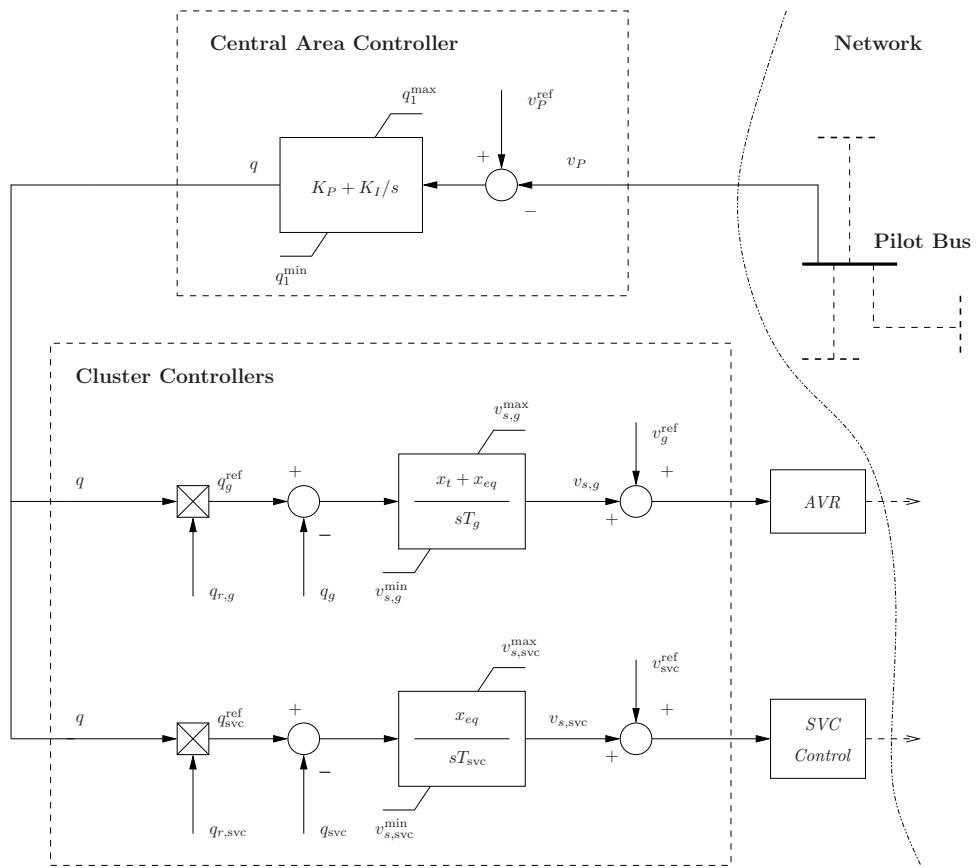


Figure 18.14: Secondary voltage control scheme.

Table 18.8: Central Area Controller Data Format (CAC.con)

Column	Variable	Description	Unit
1	-	Pilot bus number	int
2	S_n	Power rating	MVA
3	V_n	Voltage rating	kV
4	-	number of connected CC	int
5	v_P^{ref}	Reference pilot bus voltage	p.u.
6	K_I	Integral control gain	p.u.
7	K_P	Proportional control gain	p.u.
8	q_I^{\max}	Maximum output signal	p.u.
9	q_I^{\min}	Minimum output signal	p.u.
† 10	u	Connection status	{0, 1}

Cluster Controller Data

1. **con**: Cluster Controller data
2. **n**: total number of CC
3. **bus**: indexes of generator or SVC buses
4. **syn**: indexes of generators
5. **avr**: indexes of AVR
6. **svc**: indexes of SVCs
7. **vs**: indexes of the state variable v_s
8. **store**: data backup.
9. **u**: connection status

Tables 18.8 and 18.9 depicts the data format of Central Area and Cluster Controllers.

18.6 Power Oscillation Damper

This section describes a Power Oscillation Damper (POD). This model has been initially implemented by Hugo M. Ayres and Marcelo S. Castro,⁴ and later rewritten and improved by the main author of PSAT. An important contribution was given also by Dr. Alberto Del Rosso.⁵ The differential equations as well as the

⁴Hugo M. Ayres and Marcelo S. Castro are with Faculdade de Engenharia Elétrica e de Computação, UNICAMP, Brasil. E-mail: hmayres@dsce.fee.unicamp.br and mcastro@dsce.fee.unicamp.br

⁵Dr. Alberto Del Rosso is with the Electric Power Research Institute (EPRI), Knoxville, TN, and with National University of Technology, Buenos Aires. E-mail: adelrosso@epri.com

Table 18.9: Cluster Controller Data Format (`Cluster.con`)

Column	Variable	Description	Unit
1	-	Central Area Controller number	int
2	-	AVR or SVC number	int
3	-	Control type (1) AVR; (2) SVC	int
4	T_g (T_{svc})	Integral time constant	s
5	x_t	Generator transformer reactance	p.u.
6	x_{eq}	Equivalent reactance	p.u.
7	$q_{r,g}$ ($q_{r,svc}$)	Reactive power ratio	p.u.
8	v_s^{\max}	Maximum output signal	p.u.
9	v_s^{\min}	Minimum output signal	p.u.
† 10	u	Connection status	{0, 1}

control scheme of the POD are the same as the PSS Type II (see equations (18.27) and Fig. 18.9). The output signal of the POD can be used with SVC, TCSC, STATCOM, SSSC, UPFC, nad DFIG devices (see Chapter 20 for details).

The input signal v_{SI} can be as follows:

1. Bus voltage magnitude v .
2. Line active power flow p_{ij} .
3. Line active power flow p_{ji} .
4. Line current flow i_{ij} .
5. Line current flow i_{ji} .
6. Line reactive power flow q_{ij} .
7. Line reactive power flow q_{ji} .

where i and j indicates “from bus” and “to bus”, respectively. the controlled bus can be any “controllable” bus, i.e. any bus which is not controlled by other control loops such as LTC transformers. The controlled line can be any transmission line and any not regulated transformer. The output signal of the POD is v_s .

PODs are stored in the global variables `Pod`, which is an instance of the class `P0class`. Relevant properties are as follows:

1. `con`: data chart of the Pod components.
2. `n`: total number of PODs.
3. `svc`: index of SVCs to which the PODs are connected.
4. `tcsc`: index of TCSCs to which the PODs are connected.
5. `statcom`: index of STATCOMs to which the PODs are connected.

Table 18.10: Power Oscillation Damper Data Format (Pod.con)

Column	Variable	Description	Unit
1	-	Bus or line number	int
2	-	FACTS number	int
3	-	1 Bus voltage v 2 Line active power p_{ij} 3 Line active power p_{ji} Input signal 4 Line current i_{ij} 5 Line current i_{ji} 6 Line reactive power q_{ij} 7 Line reactive power q_{ji}	int
4	-	1 SVC 2 TCSC FACTS type 3 STATCOM 4 SSSC 5 UPFC 6 DFIG	int
5	v_s^{\max}	Max stabilizer output signal	p.u.
6	v_s^{\min}	Min stabilizer output signal	p.u.
7	K_w	Stabilizer gain	p.u./p.u.
8	T_w	Wash-out time constant	s
9	T_1	First stabilizer time constant	s
10	T_2	Second stabilizer time constant	s
11	T_3	Third stabilizer time constant	s
12	T_4	Fourth stabilizer time constant	s
† 13	u	Connection status	{0, 1}

6. **sssc**: index of SSSCs to which the PODs are connected.
7. **upfc**: index of UPFCs to which the PODs are connected.
8. **dfig**: index of DFIGs to which the PODs are connected.
9. **v1**: indexes of the state variable v_1 .
10. **v2**: indexes of the state variable v_2 .
11. **v3**: indexes of the state variable v_3 .
12. **vs**: indexes of the state variable v_s .
13. **store**: data backup.
14. **u**: connection status.

The data format of PODs components is depicted in Table 18.10.

Chapter 19

Regulating Transformers

This chapter describes dynamic models and data formats of the Under Load Tap Changer (ULTC) and the Phase Shifting Transformer (PST). The presented models are included in power flow analysis and do not need re-factorization.

19.1 Under Load Tap Changer

The equivalent π circuit of the Under Load Tap Changer (ULTC) transformer is depicted in Fig. 19.1. No magnetizing shunt is considered. The algebraic equations of the power injections are as follows:

$$\begin{aligned} p_k &= v_k^2 g_T / \tilde{m}^2 - v_k v_m (g_T \cos \theta_{km} + b_T \sin \theta_{km}) / \tilde{m} \\ q_k &= -v_k^2 b_T / \tilde{m}^2 - v_k v_m (g \sin \theta_{km} - b \cos \theta_{km}) / \tilde{m} \\ p_m &= v_m^2 g_T - v_k v_m (g_T \cos \theta_{km} - b_T \sin \theta_{km}) / \tilde{m} \\ q_m &= -v_m^2 b_T + v_k v_m (g_T \sin \theta_{km} + b_T \cos \theta_{km}) / \tilde{m} \end{aligned} \quad (19.1)$$

where \tilde{m} is the transformer tap ratio, $\theta_{km} = \theta_k - \theta_m$ and $g_T + jb_T = 1/(r_T + jx_T)$ is the series admittance of the transformer.

If the user sets a tap ratio step $\Delta m = 0$, then

$$\tilde{m} = m \quad (19.2)$$

where m is the state variable of one of the following continuous differential equations that are used for the ULTC control. For the voltage control, one has:

$$\dot{m} = -Hm + K(v_m - v_{\text{ref}}) \quad (19.3)$$

where the sign of the error $\epsilon_v = v_m - v_{\text{ref}}$ is due to the stability characteristic of the non-linear control loop. The ULTC can also control the voltage v_r at a remote bus r . For the reactive power control, the following differential equation holds:

$$\dot{m} = -Hm + K(q_{\text{ref}} + q_m) \quad (19.4)$$

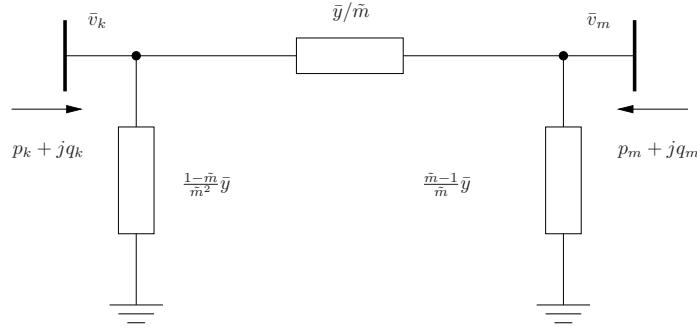
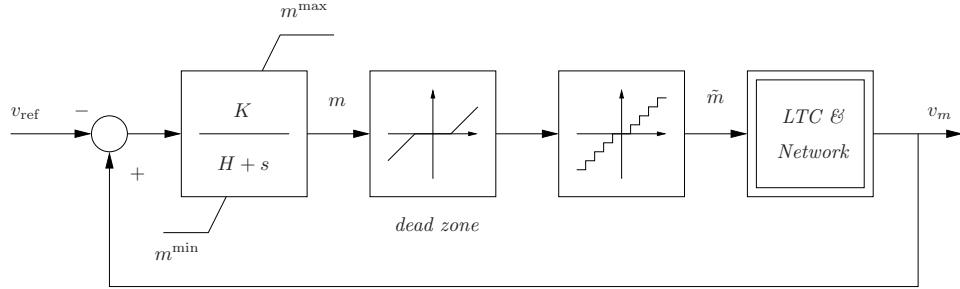
Figure 19.1: Under Load Tap Changer: equivalent π circuit.

Figure 19.2: Under Load Tap Changer: secondary voltage control scheme.

where q_m is inductive and injected at the bus m (hence the plus sign). The variable m undergoes an anti-windup limiter. Figure 19.2 depicts the ULTC secondary voltage control scheme. Similar schemes hold for the remote voltage and the reactive power controls.

If the tap ratio step $\Delta m > 0$, PSAT uses a discrete model for the ULTC and the tap ratio \tilde{m} is used.

It is not allowed to control the voltage on a PV generator or the reactive power of a PQ load. If this control is set, the power flow routine does not reach any convergence, or the message *badly scaled Jacobian matrix* is displayed. The data used for the transformer and the control are in p.u., and in nominal condition, the tap ratio is $m = 1$ p.u. Table 19.1 reports the ULTC data format. The bus number r can be $r = 0$ if local voltage or reactive power control are used. ULTCs are defined in the global variable `Ltc`, which is an instance of the class `@LTclass`. Relevant properties are as follows:

1. `con`: ULTCs data.
2. `n`: total number of ULTCs.
3. `bus1, v1`: indexes of voltages of bus k (primary winding).

Table 19.1: Load Tap Changer Data Format (`Ltc.con`)

Column	Variable	Description	Unit
1	k	Bus number (from)	int
2	m	Bus number (to)	int
3	S_n	Power rating	MVA
4	V_n	Voltage rating	kV
5	f_n	Frequency rating	Hz
6	k_T	Nominal tap ratio	kV/kV
7	H	Integral deviation	p.u.
8	K	Inverse time constant	1/s
9	m^{\max}	Max tap ratio	p.u./p.u.
10	m^{\min}	Min tap ratio	p.u./p.u.
11	Δm	Tap ratio step	p.u./p.u.
12	$v_{\text{ref}} (q_{\text{ref}})$	Reference voltage (power)	p.u.
13	x_T	Transformer reactance	p.u.
14	r_T	Transformer resistance	p.u.
15	r	Remote control bus number	int
16	-	1 Secondary voltage v_m control 2 Reactive power q_m control 3 Remote voltage v_r control	int
† 17	d	Dead zone percentage	%
† 18	u	Connection status	{0, 1}

4. `bus2`, `v2`: indexes of voltages of bus m (secondary winding).
5. `dat`: ULTC parameters.
6. `mc`: indexes of the state variable m .
7. `md`: indexes of the algebraic variable \tilde{m} .
8. `mold`: current value of the algebraic variable \tilde{m} .
9. `store`: data backup.
10. `u`: connection status.

19.2 Phase Shifting Transformer

The equivalent circuit of the Phase Shifting Transformer (PHS) is depicted in Fig. 19.3. No magnetizing shunt is considered. The algebraic equations of the

power injections are as follows:

$$\begin{aligned} p_k &= v_k^2 g_T / m^2 - v_k v_m (g_T \cos(\theta_{km} - \alpha) + b_T \sin(\theta_{km} - \alpha)) / m \\ q_k &= -v_k^2 b_T / m^2 - v_k v_m (g \sin(\theta_{km} - \alpha) - b \cos(\theta_{km} - \alpha)) / m \\ p_m &= v_m^2 g_T - v_k v_m (g_T \cos(\theta_{km} - \alpha) - b_T \sin(\theta_{km} - \alpha)) / m \\ q_m &= -v_m^2 b_T + v_k v_m (g_T \sin(\theta_{km} - \alpha) + b_T \cos(\theta_{km} - \alpha)) / m \end{aligned} \quad (19.5)$$

where $\theta_{km} = \theta_k - \theta_m$ and $g_T + jb_T = 1/(r_T + jx_T)$ is the series admittance of the transformer. Figure 19.4 depicts the PHS control block diagrams. The measure p_{mes} of the real power flow p_k is compared with the desired power flow p_{ref} and a PI controller is used for varying the phase angle α . Differential equations are as follows:

$$\begin{aligned} \dot{\alpha} &= K_p(p_k - p_{mes})/T_m + K_i(p_{mes} - p_{ref}) \\ \dot{p}_{mes} &= (p_k - p_{mes})/T_m \end{aligned} \quad (19.6)$$

The phase angle α is subjected to an anti-windup limiter. It is not allowed to connect two areas of a network only by means of PHSs, as this would lock the total real power transfer between the two areas. The data format is reported in Table 19.2.

Phase angle regulating transformers are defined in the global variable `Phs`, which is an instance of the class `@PHclass`. Relevant properties are as follows:

1. `con`: PHS data.
2. `n`: total number of PHSs.
3. `bus1`, `v1`: indexes of voltages of bus k (primary winding).
4. `bus2`, `v2`: indexes of voltages of bus m (secondary winding).
5. `alpha`: indexes of the state variable α .
6. `Pm`: indexes of the state variable p_{mes} .
7. `store`: data backup.
8. `u`: connection status.

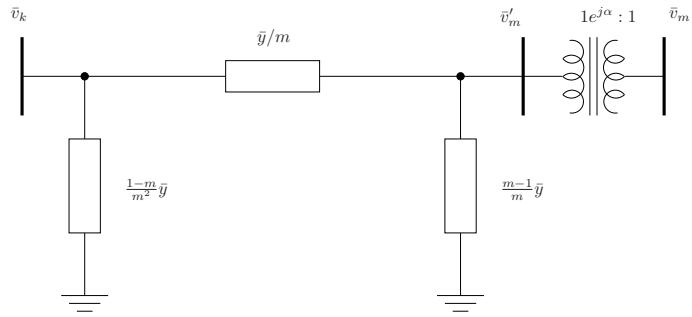


Figure 19.3: Phase shifting transformer circuit.

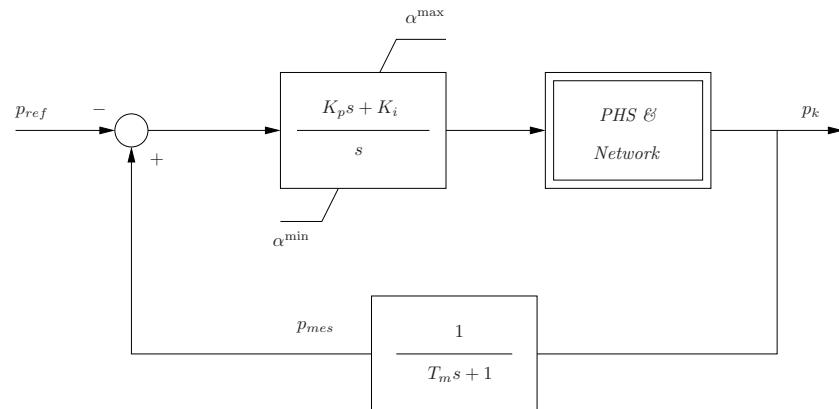


Figure 19.4: Phase shifting transformer control scheme.

Table 19.2: Phase Shifting Transformer Data Format (Phs.con)

Column	Variable	Description	Unit
1	k	Bus number (from)	int
2	m	Bus number (to)	int
3	S_n	Power rating	MVA
4	V_{n1}	Primary voltage rating	kV
5	V_{n2}	Secondary voltage rating	kV
6	f_n	Frequency rating	Hz
7	T_m	Measurement time constant	s
8	K_p	Proportional gain	-
9	K_i	Integral gain	-
10	p_{ref}	Reference power	p.u.
11	r_T	Transformer resistance	p.u.
12	x_T	Transformer reactance	p.u.
13	α^{\max}	Maximum phase angle	rad
14	α^{\min}	Minimum phase angle	rad
15	m	Transformer fixed tap ratio	p.u./p.u.
† 16	u	Connection status	{0, 1}

Chapter 20

FACTS

This chapter describes the models of Thyristor Controlled Reactor (TCR) and Voltage Sourced Inverter (VSI) based Flexible ac Transmission System (FACTS) Controllers and High Voltage dc (HVDC) transmission system. In particular, TCR are represented by Static Var Compensator (SVC) and Thyristor Controlled Series Compensator (TCSC) , whereas VSI are the Static Var Compensator (STATCOM), the Static Synchronous Source Series Compensator (SSSC) and the Unified Power Flow Controller (UPFC) .

Shunt components, i.e. SVCs, STATCOMs and UPFCs, require a PV generator to be properly initialized. In the case of UPFCs, the PV generator must be placed at the sending end bus.

SVC, TCSC, STATCOM, SSSC and UPFC models have an additional stabilizing signal v_s^{POD} , which is the output of the Power Oscillation Damper described in Section 18.6.

TCSC, STATCOM, SSSC and UPFC models have been initially implemented by Hugo M. Ayres and Marcelo S. Castro,¹ and later rewritten and improved by the main author of PSAT. Important contributions were also given by Dr. Alberto Del Rosso.²

20.1 SVC

Two SVC regulators are implemented in the program. The first one assumes a time constant regulator, as depicted in Fig. 20.1. In this model, a total reactance b_{SVC} is assumed and the following differential equation holds:

$$\dot{b}_{\text{SVC}} = (K_r(v_{\text{ref}} + v_s^{\text{POD}} - v) - b_{\text{SVC}})/T_r \quad (20.1)$$

¹Hugo M. Ayres and Marcelo S. Castro are with Faculdade de Engenharia Elétrica e de Computação, UNICAMP, Brasil.

E-mail: hmayres@dsce.fee.unicamp.br and mcastro@dsce.fee.unicamp.br

²Dr. Alberto Del Rosso is with the Electric Power Research Institute (EPRI), Knoxville, TN, and with National University of Technology, Buenos Aires. E-mail: adelrosso@epri.com

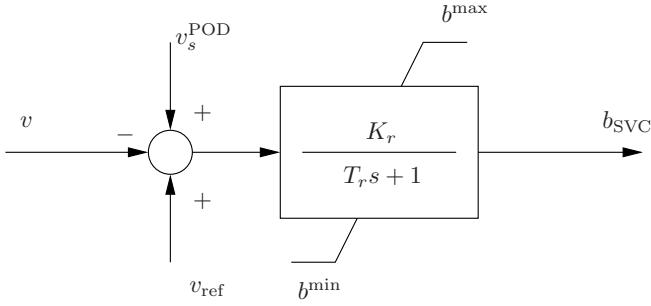


Figure 20.1: SVC Type 1 Regulator.

The model is completed by the algebraic equation expressing the reactive power injected at the SVC node:

$$q = b_{\text{SVC}}v^2 \quad (20.2)$$

The regulator has an anti-windup limiter, thus the reactance b_{SVC} is locked if one of its limits is reached and the first derivative is set to zero. Table 20.1 reports the data and control parameter format for the SVC type 1.

The second model takes into account the firing angle α , assuming a balanced, fundamental frequency operation. Thus, the model can be developed with respect to a sinusoidal voltage. The differential and algebraic equations are as follows:

$$\begin{aligned} \dot{v}_M &= (K_M v - v_M)/T_M & (20.3) \\ \dot{\alpha} &= (-K_D \alpha + K \frac{T_1}{T_2 T_M} (v_M - K_M v) + K(v_{\text{ref}} + v_s^{\text{POD}} - v_M))/T_2 \\ q &= \frac{2\alpha - \sin 2\alpha - \pi(2 - x_L/x_C)}{\pi x_L} v^2 = b_{\text{SVC}}(\alpha)v^2 \end{aligned}$$

The state variable α undergoes an anti-windup limiter.

The SVCs state variables are initialized after the power flow solution. To impose the desired voltages at the compensated buses, a PV generator with zero active power should be used. After the power flow solution the PV bus is removed and the SVC equations are used. During the state variable initialization a check for SVC limits is performed.

Table 20.1 and Fig. 20.2 report the complete data format and the control block diagram for the SVC model 2.

SVC devices are defined in the global variable `Svc`, which is an instance of the class `SVclass`. Relevant properties are as follows:

1. `con`: SVC data.
2. `n`: total number of SVC.
3. `bus`: SVC bus numbers.

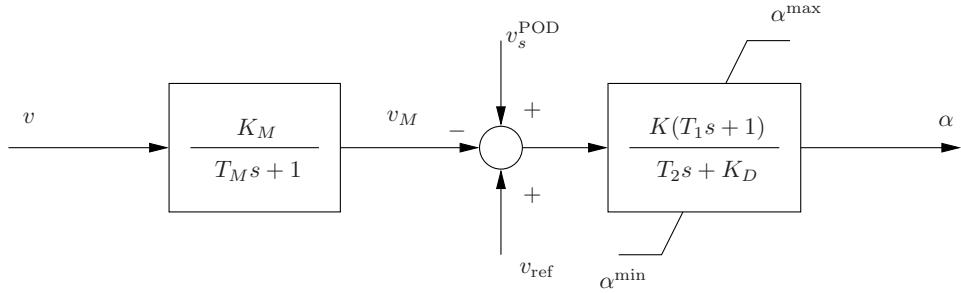


Figure 20.2: SVC Type 2 Regulator.

Table 20.1: SVC Type 1 Data Format (`Svc.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	V_n	Voltage rating	kV
4	f_n	Frequency rating	Hz
5	1	Model type	int
6	T_r	Regulator time constant	s
7	K_r	Regulator gain	p.u./p.u.
8	v_{ref}	Reference Voltage	p.u.
9	b^{\max}	Maximum susceptance	p.u.
10	b^{\min}	Minimum susceptance	p.u.
† 17	u	Connection status	{0, 1}

Table 20.2: SVC Type 2 Data Format (`Svc.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	V_n	Voltage rating	kV
4	f_n	Frequency rating	Hz
5	2	Model type	int
6	T_2	Regulator time constant	s
7	K	Regulator gain	p.u./p.u.
8	v_{ref}	Reference Voltage	p.u.
9	α^{\max}	Maximum firing angle	rad
10	α^{\min}	Minimum firing angle	rad
11	K_D	Integral deviation	p.u.
12	T_1	Transient regulator time constant	s
13	K_M	Measure gain	p.u./p.u.
14	T_M	Measure time delay	s
15	x_L	Reactance (inductive)	p.u.
16	x_C	Reactance (capacitive)	p.u.
† 17	u	Connection status	{0, 1}

4. `vbus`: indexes of bus voltages.
5. `bcv`: indexes of the state variable b_{SVC} .
6. `alpha`: indexes of the state variable α .
7. `vm`: indexes of the state variable v_M .
8. `vref`: indexes of the algebraic variable v_{ref} .
9. `Be`: equivalent admittances b_{SVC} .
10. `store`: data backup.
11. `u`: connection status.

20.2 TCSC

TCSC regulator is depicted in Fig. 20.3. The active and reactive power flows in the line compensated by the TCSC are:

$$\begin{aligned} p_{km} &= v_k v_m (y_{km} + b) \sin(\theta_k - \theta_m) \\ p_{mk} &= -p_{km} \end{aligned} \tag{20.4}$$

$$\begin{aligned} q_{km} &= v_k^2(y_{km} + b) - v_k v_m (y_{km} + b) \cos(\theta_k - \theta_m) \\ q_{mk} &= v_m^2(y_{km} + b) - v_k v_m (y_{km} + b) \cos(\theta_k - \theta_m) \end{aligned}$$

where k and m stand for the sending and receiving bus indexes, respectively, and $Y_{km} = 1/x_{km}$ is the admittance of the line compensated by the TCSC device.

The TCSC differential equations are as follows:

$$\begin{aligned} \dot{x}_1 &= (\{\tilde{x}_{\text{TCSC}}, \tilde{\alpha}\} + K_r v_s^{\text{POD}} - x_1)/T_r \\ \dot{x}_2 &= K_I(P_{km} - P_{\text{ref}}) \end{aligned} \quad (20.5)$$

where

$$\{\tilde{x}_{\text{TCSC}}, \tilde{\alpha}\} = K_P(p_{km} - p_{\text{ref}}) + x_2 \quad (20.6)$$

The state variable x_1 is the TCSC series reactance \tilde{x}_{TCSC} or the firing angle $\tilde{\alpha}$, depending on the TCSC model. The PI controller is enabled only for the constant power flow operation mode.

The output signal is the series susceptance b of the TCSC, as follows:³

$$b(x_{\text{TCSC}}) = -\frac{x_{\text{TCSC}}/x_{km}}{x_{km}(1 - x_{\text{TCSC}}/x_{km})} \quad (20.7)$$

or

$$\begin{aligned} b(\alpha) &= \pi(k_x^4 - 2k_x^2 + 1) \cos k_x(\pi - \alpha) / \\ &\quad \left[x_C \left(\pi k_x^4 \cos k_x(\pi - \alpha) \right. \right. \\ &\quad - \pi \cos k_x(\pi - \alpha) - 2k_x^4 \alpha \cos k_x(\pi - \alpha) \\ &\quad + 2\alpha k_x^2 \cos k_x(\pi - \alpha) - k_x^4 \sin 2\alpha \cos k_x(\pi - \alpha) \\ &\quad + k_x^2 \sin 2\alpha \cos k_x(\pi - \alpha) - 4k_x^3 \cos^2 \alpha \sin k_x(\pi - \alpha) \\ &\quad \left. \left. - 4k_x^2 \cos \alpha \sin \alpha \cos k_x(\pi - \alpha) \right) \right] \end{aligned} \quad (20.8)$$

³Equation (20.7) can be obtained as follows. The total active power flow in the line compensated by the TCSC is:

$$p_{km}^{\text{tot}} = \frac{v_k v_m \sin \theta_{km}}{(x_{km} - x_{\text{TCSC}})}$$

while the flows of the line without compensation and of the sole TCSC are, respectively:

$$\begin{aligned} p_{km}^{\text{line}} &= \frac{v_k v_m \sin \theta_{km}}{x_{km}} \\ p_{km}^{\text{TCSC}} &= b v_k v_m \sin \theta_{km} \end{aligned}$$

Thus, one has:

$$b = \frac{1}{x_{km}} - \frac{1}{x_{km} - x_{\text{TCSC}}}$$

from where (20.7) can be easily deduced. Observe that x_{TCSC}/x_{km} is a “dynamic” c_p . At the initial point (e.g. at the power flow solution), $x_{\text{TCSC}}/x_{km} = c_p$.

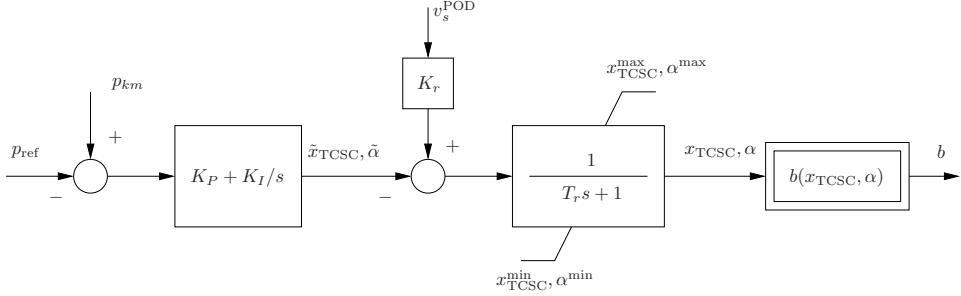


Figure 20.3: TCSC Regulator.

$$k_x = \sqrt{\frac{x_C}{x_L}}$$

During the power flow analysis the TCSC is modeled as a constant capacitive reactance that modifies the line reactance x_{km} , as follows:

$$x'_{km} = (1 - c_p)x_{km} \quad (20.9)$$

where c_p is the percentage of series compensation. The TCSC state variables are initialized after the power flow analysis as well as the reference power of the PI controller p_{ref} . At this step, a check of x_{TCSC} and/or α anti-windup limits is performed. In case of limit violation, a warning message is displayed. Table 20.3 reports the data format of TCSC devices.

TCSC devices are defined in the global variable `Tcsc`, which is an instance of the class `TCclass`. Relevant properties are as follows:

1. `con`: TCSC data.
2. `n`: total number of TCSCs.
3. `line`: line number i .
4. `bus1, v1`: indexes of voltages of bus k (from).
5. `bus2, v2`: indexes of voltages of bus m (to).
6. `sscl`: indexes of the SSCL connected to the TCSC.
7. `x1`: indexes of state variables x_1 .
8. `x2`: indexes of state variables x_2 .
9. `B`: series admittance B .
10. `Cp`: amount of series compensation c_p .
11. `x0`: initial series reactance $x_{\text{TCSC}0}$.

Table 20.3: TCSC Data Format (`Tcsc.con`)

Column	Variable	Description	Unit
1	i	Line number	int
2	-	Model type 1 Reactance x_{TCSC} 2 Firing angle α	int
3	-	Operation mode 1 Const. x_{TCSC} 2 Const. p_{km}	int
4	-	Scheduling strategy 1 Const. p_{km} 2 Const. θ_{km}	int
5	S_n	Power rating	MVA
6	V_n	Voltage rating	kV
7	f_n	Frequency rating	Hz
8	C_p	Percentage of series compensation	%
9	T_r	Regulator time constant	s
10	$x_{TCSC}^{\max} (\alpha^{\max})$	Maximum reactance (firing angle)	rad
11	$x_{TCSC}^{\min} (\alpha^{\min})$	Minimum reactance (firing angle)	rad
12	K_P	Proportional gain of PI controller	p.u./p.u.
13	K_I	Integral gain of PI controller	p.u./p.u.
14	x_L	Reactance (inductive)	p.u.
15	x_C	Reactance (capacitive)	p.u.
16	K_r	Gain of the stabilizing signal	p.u./p.u.
† 17	u	Connection status	{0, 1}

12. **Pref**: reference power flow P_{ref} .

13. **y**: line admittance $1/x_{km}$.

14. **store**: data backup.

15. **u**: connection status.

The TCSC data format is depicted in Table 20.3.

20.3 STATCOM

The implemented STATCOM model is a current injection model which is based on [37, 105, 53]. The STATCOM current is always kept in quadrature in relation to the bus voltage so that only reactive power is exchanged between the ac system and the STATCOM. The dynamic model is shown in Fig. 20.4 where it can be seen that the STATCOM assumes a time constant regulator like SVC.

The differential equation and the reactive power injected at the STATCOM node are given, respectively, by:

$$\dot{i}_{\text{SH}} = (K_r(v_{\text{ref}} + v_s^{\text{POD}} - v) - i_{\text{SH}})/Tr \quad (20.10)$$

$$q = i_{\text{SH}}v \quad (20.11)$$

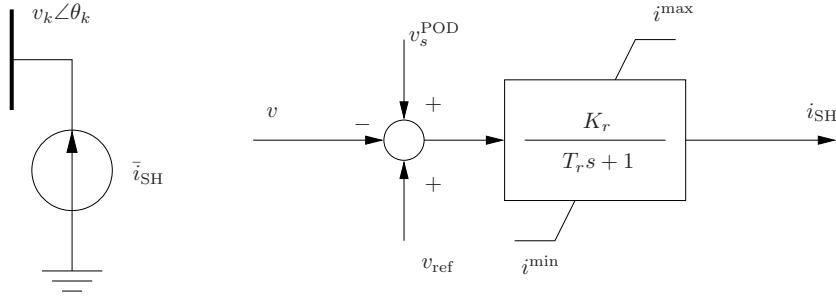


Figure 20.4: STATCOM circuit and control block diagram.

The regulator has a non-windup limiter, thus the current i_{SH} is locked if one of its limits is reached and the first derivative is set to zero.

STATCOM devices are defined in the global variable `Statcom`, which is an instance of the class `STclass`. Relevant properties are as follows:

1. `con`: STATCOM data.
2. `n`: total number of STATCOM.
3. `bus`, `vbus`: STATCOM bus voltage indexes.
4. `sscl`: indexes of the SSCL connected to the STATCOM.
5. `ist`: indexes of the state variable i_{SH} .
6. `Vref`: reference voltage of the STATCOM regulator.
7. `store`: data backup.
8. `u`: connection status.

The STATCOM data format is depicted in Table 20.4.

20.4 SSSC

The implemented SSSC model is based on [119, 74, 81]. The SSSC is represented by a series voltage source \bar{v}_S , as depicted in Fig. 20.5. The voltage \bar{v}_S is always kept in quadrature with line current. Thus the only controllable parameter is the magnitude v_S . The total active and reactive power flows in a transmission line with

Table 20.4: STATCOM Data Format (`Statcom.con`)

Column	Variable	Description	Unit
1	k	Bus number	int
2	S_n	Power rating	MVA
3	V_n	Voltage rating	kV
4	f_n	Frequency rating	Hz
5	K_r	Regulator gain	p.u./p.u.
6	T_r	Regulator time constant	s
7	i^{\max}	Maximum current	p.u.
8	i^{\min}	Minimum current	p.u.
† 9	u	Connection status	{0, 1}

a SSSC are as follows:

$$\begin{aligned} p_{km} &= (1 + \epsilon) \frac{v_k v_m}{x_{km}} \sin(\theta_k - \theta_m) \\ p_{mk} &= -P_{km} \\ q_{km} &= (1 + \epsilon) \frac{v_k}{x_{km}} (v_k - v_m \cos(\theta_k - \theta_m)) \\ q_{mk} &= (1 + \epsilon) \frac{v_m}{x_{km}} (v_m - v_k \cos(\theta_k - \theta_m)) \end{aligned} \quad (20.12)$$

where

$$\epsilon = \frac{v_S}{\sqrt{v_k^2 + v_m^2 - 2v_k v_m \cos \theta_{km}}} \quad (20.13)$$

where $\theta_{km} = \theta_k - \theta_m$. The SSSC dynamic model is illustrated in Fig. 20.6 and the differential equation that describes the dynamic behavior of the SSSC is:

$$\dot{v}_S = (v_S^0 + v_s^{\text{POD}} - v_S)/T_r \quad (20.14)$$

The input signal v_S^0 determines the SSSC operation mode which in turn determines the value of the SSSC voltage v_S in steady-state. Three different control modes are implemented for the SSSC: 1) constant voltage, 2) constant reactance, and 3) constant power flow. For each control mode, the input signal v_S^0 is given as follows:

- 1) **Constant voltage:** The magnitude of the voltage v_S at steady-state is kept constant independently of the line current, so the input $v_S^0 = \text{constant}$.
- 2) **Constant reactance:** The magnitude of the signal v_S varies proportionally to the line current keeping constant the total reactance of the transmission line

where the SSSC is installed. In this operation mode the signal v_S^0 is as follows:

$$\begin{aligned} v_S^0 &= c_p x_{km} I_{km} \\ &= c_p x_{km} \frac{\sqrt{v_k^2 + v_m^2 - 2v_k v_m \cos \theta_{km}}}{x'_{km}} \\ &= \frac{c_p}{1 - c_p} \sqrt{v_k^2 + v_m^2 - 2v_k v_m \cos \theta_{km}} \end{aligned} \quad (20.15)$$

where c_p is the degree of series compensation in p.u., I_{km} is the magnitude of the line current, and x_{km} and $x'_{km} = (1 - c_p)x_{km}$ are the full and the compensated reactance of the transmission line, respectively.

3) Constant power flow: Constant power control mode: In this mode, the signal v_S^0 is the output of a PI controller used to control the power flow in transmission systems, as shown in Fig. 20.6. Two strategies are implemented for the constant power flow control mode:

- 3.a) *Constant line power*: This control strategy is used to keep constant the power flow in the transmission line where the SSSC is installed.
- 3.b) *Constant angle*: This control strategy is used to control the power flow in a parallel transmission line.

SSSC devices are defined in the global variable `Sssc`, which is an instance of the class `SSclass`. Relevant properties are as follows:

1. `con`: SSSC data.
2. `n`: total number of SSSC.
3. `line`: line numbers i .
4. `bus1, v1`: indexes of voltages of bus k (from).
5. `bus2, v2`: indexes of voltages of bus m (to).
6. `sscl`: indexes of the SSCL connected to the SSSC.
7. `vcs`: indexes of the state variable v_S .
8. `vpi`: indexes of the state variable of the PI controller.
9. `xcs`: compensation reactance $(1 - c_p)x_{km}$.
10. `Cp`: amount of series compensation c_p .
11. `V0`: initial compensation voltage v_{S0} .
12. `Pref`: reference power flow p_{ref} .
13. `y`: line admittance $1/x_{km}$.

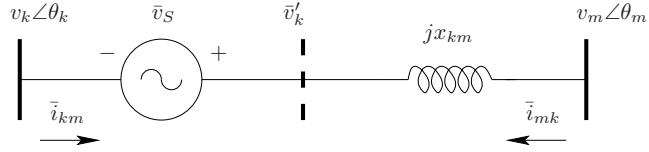


Figure 20.5: SSSC circuit.

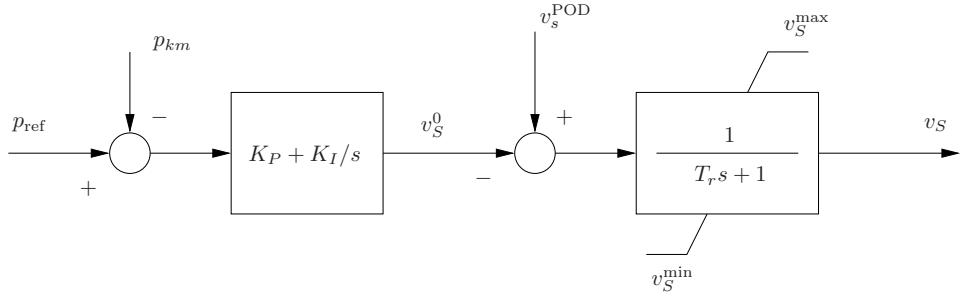


Figure 20.6: SSSC control block diagram.

14. `store`: data backup.

15. `u`: connection status.

Table 20.5 reports the complete SSSC data format.

20.5 UPFC

The implemented UPFC model is based on [98, 73, 80]. The circuital model of the UPFC is obtained from the STATCOM and SSSC. It is represented by one series voltage source \bar{v}_S and by one shunt current source \bar{i}_{SH} , as depicted in Fig. 20.7. The series voltage source and the shunt current source are defined as follows:

$$\begin{aligned}\bar{v}_S &= (v_p + v_q)e^{j\phi} = r\bar{v}_k e^{j\gamma} \\ \bar{i}_{SH} &= (i_p + i_q)e^{j\theta_k}\end{aligned}\quad (20.16)$$

The equivalent circuit vector diagram of series voltage source is shown in Fig. 20.8, and the power equations that describe the power injection model of the UPFC are:

$$\begin{aligned}p_{km} &= brv_k v_m \sin(\gamma + \theta_k - \theta_m) \\ q_{km} &= brv_k^2 \cos \gamma - i_q v_k \\ p_{mk} &= -brv_k v_m \sin(\gamma + \theta_k - \theta_m) \\ q_{mk} &= -brv_k v_m \cos(\gamma + \theta_k - \theta_m)\end{aligned}\quad (20.17)$$

Table 20.5: SSSC Data Format (`Sssc.con`)

Column	Variable	Description	Unit
1	i	Line number	int
2	-	1 Constant voltage Operation mode 2 Constant reactance 3 Constant power	int
3	S_n	Power rating	MVA
4	V_n	Voltage rating	kV
5	f_n	Frequency rating	Hz
6	C_p	Percentage of series compensation	%
7	T_r	Regulator time constant	s
8	v_S^{\max}	Maximum series voltage v_S	p.u.
9	v_S^{\min}	Minimum series voltage v_S	p.u.
10	-	Scheduling type 1 Constant p_{km} 2 Constant θ_{km}	int
11	K_P	Proportional gain of PI controller	p.u./p.u.
12	K_I	Integral gain of PI controller	p.u./p.u.
† 13	u	Connection status	{0, 1}

The UPFC dynamic model has a 3rd order, as depicted in Fig. 20.9. Observe that the POD controller can be used to modulate whatever of UPFC variables (v_p , v_q , i_q). The set of differential equations are as follows:

$$\begin{aligned}\dot{v}_p &= \frac{1}{T_r}(v_{p0} + u_1 v_s^{\text{POD}} - v_p) \\ \dot{v}_q &= \frac{1}{T_r}(v_{q0} + u_2 v_s^{\text{POD}} - v_q) \\ \dot{i}_q &= \frac{1}{T_r}[K_r(v_{\text{ref}} + u_3 v_s^{\text{POD}} - v_k) - i_q]\end{aligned}\quad (20.18)$$

where u_1 , u_2 and u_3 are 1 if the correspondent stabilizing POD signal is enabled, 0 otherwise.

UPFC State Variables

v_p : This variable represents the component of the series voltage \bar{v}_S that is in phase with the line current. In steady-state, the input v_p^0 is set to zero so that the exchange of active power between the UPFC and the ac system only takes place when this variable is modulated by the POD controller (i.e. during transients).

v_q : This variable represents the component of series voltage \bar{v}_S that is in quadrature with line current. The input v_q^0 determines the value of the variable v_q in steady-state. Two control modes are implemented for this variable:

1. *Constant voltage*: the magnitude of voltage v_q is constant independently of the line current;
2. *constant reactance*: the magnitude of the voltage v_q varies proportionally to the line current keeping constant the total impedance (the resistance is actually neglected) of the transmission line.

i_q : This variable represents the component of shunt current \bar{i}_{SH} which is in quadrature with the bus voltage \bar{v}_k . This current keeps the bus voltage around a specified level through the regulator gain K_r .

UPFC devices are defined in the global variable `Upfc`, which is an instance of the class `UPclass`. Relevant properties are as follows:

1. `con`: UPFC data.
2. `n`: total number of UPFC.
3. `line`: line numbers i .
4. `bus1`, `v1`: indexes of voltages of bus k (from).
5. `bus2`, `v2`: indexes of voltage of bus m (to).
6. `sscl`: indexes of the SSCL connected to the UPFC.
7. `xcs`: compensation reactance $(1 - c_p)x_{km}$.
8. `Cp`: amount of series compensation c_p .
9. `vp0`: initial compensation voltage v_p^0 .
10. `vq0`: initial compensation voltage v_q^0 .
11. `Vref`: reference voltage v_{ref} .
12. `y`: line admittance $1/x_{km}$.
13. `gamma`: relative UPFC angle γ .
14. `vp`: indexes of the state variable v_p .
15. `vq`: indexes of the state variable v_q .
16. `iq`: indexes of the state variable i_q .
17. `store`: data backup.
18. `u`: connection status.

Table 20.6 illustrates the complete UPFC data format.

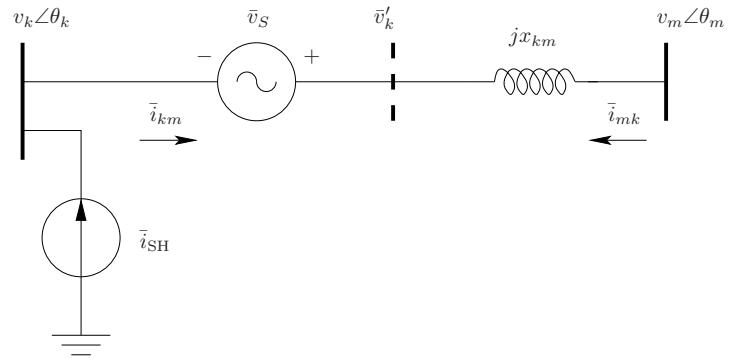


Figure 20.7: UPFC circuit.

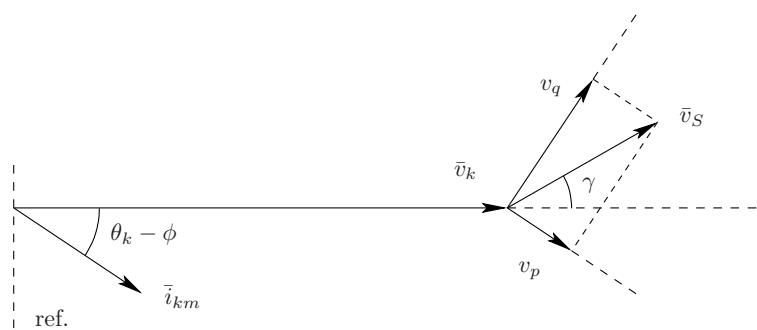


Figure 20.8: UPFC phasor diagram.

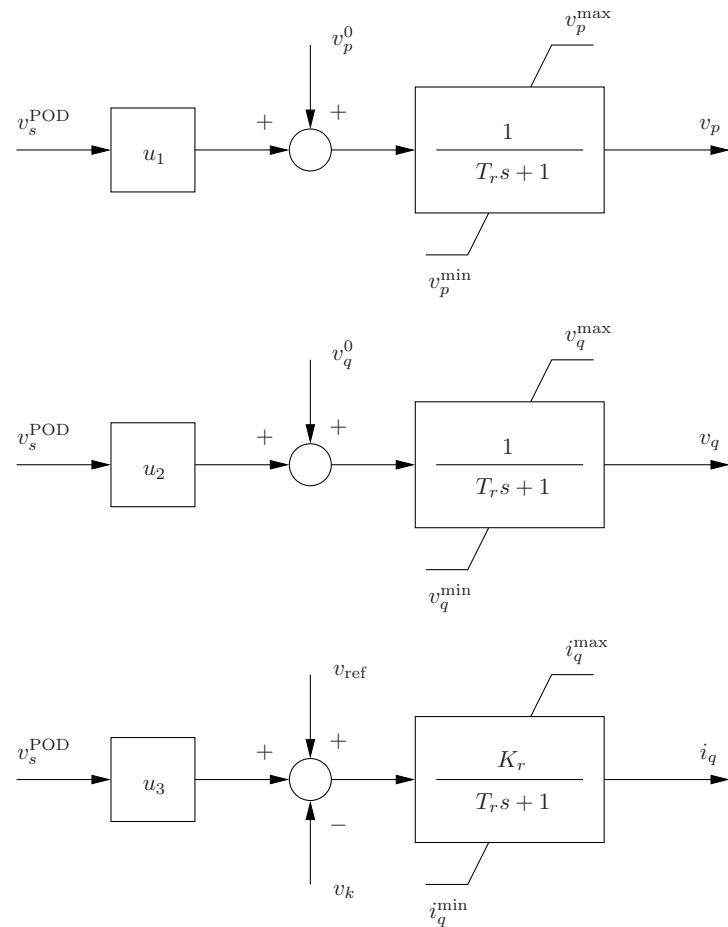


Figure 20.9: UPFC control block diagrams.

Table 20.6: UPFC Data Format (`Upfc.con`)

Column	Variable	Description	Unit
1	i	Line number	int
2	-	Operation mode 1 Constant voltage 2 Constant reactance	int
3	S_n	Power rating	MVA
4	V_n	Voltage rating	kV
5	f_n	Frequency rating	Hz
6	C_p	Percentage of series compensation	%
7	K_r	Regulator gain	p.u./p.u.
8	T_r	Regulator time constant	s
9	v_p^{\max}	Maximum v_p	p.u.
10	v_p^{\min}	Minimum v_p	p.u.
11	v_q^{\max}	Maximum v_q	p.u.
12	v_q^{\min}	Minimum v_q	p.u.
13	i_q^{\max}	Maximum i_q	p.u.
14	i_q^{\min}	Minimum i_q	p.u.
15	-	Stabilizing v_p signal	{0, 1}
16	-	Stabilizing v_q signal	{0, 1}
17	-	Stabilizing i_q signal	{0, 1}
† 18	u	Connection status	{0, 1}

20.6 HVDC

A simple HVDC system is implemented in PSAT, representing two ac/dc converters connected by a single dc line (see Fig. 20.10). The line is modeled as a dynamic RL circuit, whereas the firing angle α and the extinction angle γ are controlled by PI regulators, as depicted in Fig. 20.11. The controllers regulate the current or the power flow in the dc line. By default, the dc current i^{dc} is assumed to flows from the rectifier to the inverter. The normal operation mode is $i^{dc} \geq 0$ and $\alpha^{\min} \leq \alpha \leq \alpha^{\max}$. In normal operation mode, the inverter controller is inactive and $\gamma = \gamma^{\min}$.

Under normal operating conditions, the differential and algebraic equations for the fundamental frequency of the HVDC are as follows:

$$\begin{aligned}
\dot{i}^{dc} &= (v_R^{dc} - v_I^{dc} - r^{dc}i^{dc})/l^{dc} & (20.19) \\
\dot{x}_R &= K_i(y_R - (u_i + u_p)i^{dc} - u_v v_R^{dc}) \\
p_R &= v_R^{dc} i^{dc} \\
q_R &= k \frac{3\sqrt{2}}{\pi} v_R i^{dc} \sin \phi_R \\
p_I &= -v_I^{dc} i^{dc} \\
q_I &= k \frac{3\sqrt{2}}{\pi} v_I i^{dc} \sin \phi_I \\
\cos \alpha &= x_R + K_p(y_R - (u_i + u_p)i^{dc} - u_v v_R^{dc}) \\
v_R^{dc} &= k \frac{3\sqrt{2}}{\pi} m_R v_R \cos \alpha - \frac{3}{\pi} x_R^t i^{dc} \\
v_R^{dc} &= k \frac{3\sqrt{2}}{\pi} m_R v_R \cos \phi_R \\
y_R &= u_p P_{ord}/v_R^{dc} + u_i i_{ord} + u_v v_{ord} \\
\cos(\pi - \gamma) &= \cos(\pi - \gamma_{\min}) \\
v_I^{dc} &= k \frac{3\sqrt{2}}{\pi} m_I v_I \cos(\pi - \gamma) - \frac{3}{\pi} x_I^t i^{dc} \\
v_I^{dc} &= k \frac{3\sqrt{2}}{\pi} m_I v_I \cos \phi_I
\end{aligned}$$

where the index R is used for the rectifier quantities and I for the inverter ones; m_R and m_I are tap ratio of the transformers that connects the converter and the inverter to the ac grid; v_R and v_I are the ac primary voltages at the transformer terminals of the rectifier and inverter sides; $\cos \phi_R$ and $\cos \phi_I$ are the power factors at the rectifier and the inverter ac sides; v_R^{dc} and v_I^{dc} are the dc voltages on the dc terminals; i^{dc} is the dc current in the dc transmission line; y_R and y_I are the controller input signals for the rectifier and the inverter controllers; u_p , u_i and u_v are 1 or 0, depending of the selected control type; and $k = 0.995$ for a twelve-pulse ac/dc converter. All other parameters are defined in Table 20.7.

If the inverter takes over the current of voltage control, the equations for \dot{x}_I , $\cos \alpha$, y_I and $\cos(\pi - \gamma)$ are replaced by:

$$\begin{aligned}\dot{x}_I &= K_i((u_i + u_p)i^{dc} + u_v v_R^{dc} - y_I) \\ \cos(\pi - \gamma) &= x_I + K_p(i^{dc} - i_I^0) \\ \cos \alpha &= \cos \alpha^{\min} \\ y_I &= u_p p_{\text{ord}}/v_I^{dc} + u_i i_{\text{ord}} + u_v v_{\text{ord}}\end{aligned}\quad (20.20)$$

UPFC devices are defined in the global variable `Hvdc`, which is an instance of the class `HVclass`. Relevant properties are as follows:

1. `con`: HVDC data.
2. `n`: total number of HVDC.
3. `bus1`, `v1`: converter bus voltage indexes.
4. `bus2`, `v2`: inverter bus voltage indexes.
5. `dat`: HVDC parameters.
6. `Idc`: indexes of the state variable i^{dc} .
7. `xr`: indexes of the state variable x_R .
8. `xi`: indexes of the state variable x_I .
9. `Vrdc`: indexes of the algebraic variable v_R^{dc} .
10. `Vidc`: indexes of the algebraic variable v_I^{dc} .
11. `cosa`: indexes of the algebraic variable $\cos \alpha$.
12. `cosg`: indexes of the algebraic variable $\cos(\pi - \gamma)$.
13. `Sr`: indexes of the algebraic variable s_R .
14. `Si`: indexes of the algebraic variable s_i .
15. `Ir0`: indexes of the algebraic variable i_R^0 .
16. `Ii0`: indexes of the algebraic variable i_I^0 .
17. `store`: data backup.
18. `u`: connection status.

Table 20.7 reports the complete data format for the HVDC.

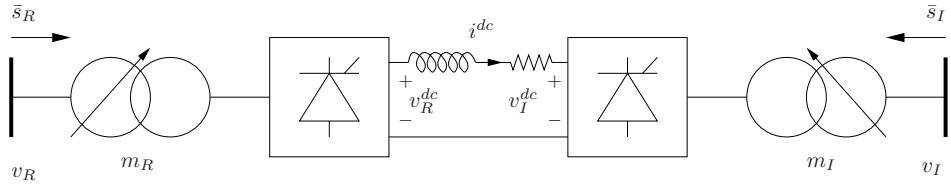
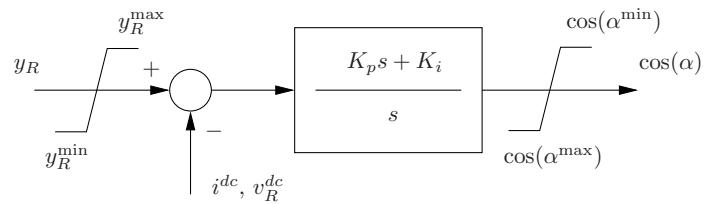


Figure 20.10: HVDC scheme.

Rectifier



Inverter

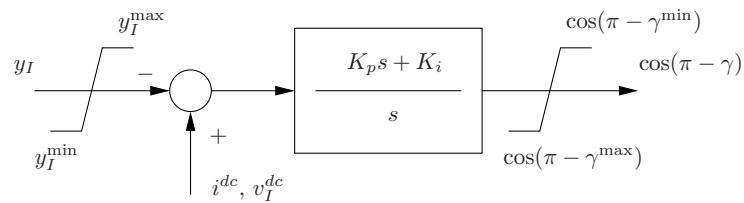


Figure 20.11: HVDC current control.

Note on the per unit system for dc quantities

In [75], the following bases are used for computing the dc per unit quantities:

$$\begin{aligned} V_{\text{base}}^{\text{dc}} &= \frac{3\sqrt{2}}{\pi} V_{\text{base}}^{\text{ac}} \\ I_{\text{base}}^{\text{dc}} &= I_{\text{rate}}^{\text{dc}} = I_n^{\text{dc}} \\ S_{\text{base}}^{\text{dc}} &= V_{\text{base}}^{\text{dc}} I_{\text{base}}^{\text{dc}} \\ Z_{\text{base}}^{\text{dc}} &= V_{\text{base}}^{\text{dc}} / I_{\text{base}}^{\text{dc}} \end{aligned} \quad (20.21)$$

In Table 20.7, dc per unit data are referred to the dc voltage and the dc current rates. The dc power base is considered equal to the ac one, i.e. $S_{\text{base}}^{\text{dc}} = S_n$. In order to avoid inconsistencies, the user should use $S_n \approx V_n^{\text{dc}} I_n^{\text{dc}}$. Before solving the power flow analysis, PSAT converts all ac per unit data to system bases.

Table 20.7: HVDC Data Format (`Hvdc.con`)

Column	Variable	Description	Unit
1	R	Bus number (rectifier)	int
2	I	Bus number (inverter)	int
3	S_n	Power rate	MVA
4	V_{Rn}	ac voltage rate at rectifier side	kV
5	V_{In}	ac voltage rate at inverter side	kV
6	f_n	Frequency rate	Hz
7	V_n^{dc}	dc voltage rate	kV
8	I_n^{dc}	dc current rate	kA
9	x_R^t	Transformer reactance (rectifier)	p.u.
10	x_I^t	Transformer reactance (inverter)	p.u.
11	m_R	Tap ratio (rectifier)	p.u.
12	m_I	Tap ratio (inverter)	p.u.
13	K_i	Integral gain	1/s
14	K_p	Proportional gain	p.u./p.u.
15	r^{dc}	Resistance of the dc connection	p.u.
16	l^{dc}	Inductance of the dc connection	p.u.
17	α^{\max}	Max. firing angle α	deg
18	α^{\min}	Min. firing angle α	deg
19	γ^{\max}	Max. extinction angle γ	deg
20	γ^{\min}	Min. extinction angle γ	deg
21	y_R^{\max}	Max. reference current or voltage (rectifier)	p.u.
22	y_R^{\min}	Min. reference current or voltage (rectifier)	p.u.
23	y_I^{\max}	Max. reference current or voltage (inverter)	p.u.
24	y_I^{\min}	Min. reference current or voltage (inverter)	p.u.
25	-	Control (1: current, 2: power, 3: voltage)	int.
26	i_{ord}	dc current order	p.u.
27	p_{ord}	dc active power order	p.u.
28	v_{ord}	dc voltage order	p.u.
† 29	u	Connection status	{0, 1}

Chapter 21

Wind Turbines

This chapter describes wind turbines and wind speed models. Three models of wind turbines are included: constant speed wind turbine with squirrel cage induction generator, variable speed wind turbine with doubly fed (wound rotor) induction generator and variable speed wind turbine with direct drive synchronous generator. Wind speed models are a Weibull's distribution and a wind model composed of average speed, ramp, gust and turbulence. Wind speed measurement data can be used as well.

Wind turbines are initialized after power flow computations and a PV generator is needed to impose the desired voltage and active power at the wind turbine bus. Once the power flow solution has been determined, v^0 , θ^0 , p^0 and q^0 at the generation bus are used for initializing the state and input variables, the latter being the wind speed v_w^0 , which is used as the average wind speed v_w^a for the wind speed models.

Controls and converter models are included in the wind turbine equations. Wind turbine models presented here were mostly based on models discussed in [118].

21.1 Wind Models

Wind speed models included in PSAT are the Weibull's distribution , the Mexican hat wavelet model [59], and a composite model which includes average speed, ramp, gust and turbulence. Real measurement data can be used as well. Regardless the wind speed model, the first value of the wind speed sequence will be the initial average speed ($v_w(t_0) = v_w^a$) as computed at the initialization step of the wind turbines (see Section 21.2).

Table 21.1 depicts the data format for wind speed models. Air density ρ at 15°C and standard atmospheric pressure is 1.225 kg/m³, and depends on the altitude (e.g. at 2000 m ρ is 20% lower than at the sea level).

The initial wind speed, which is also used as the average speed, is determined during the initialization of wind turbines. It is important to note that, if two or more wind turbines are associated with the same wind (e.g. same wind speed

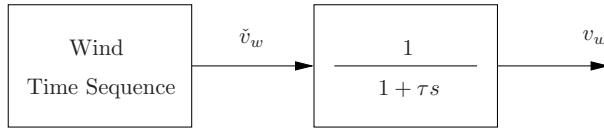


Figure 21.1: Low-pass filter to smooth wind speed variations.

model index), only the last wind turbine will be correctly initialized. To avoid inconsistencies, PSAT displays an error message if two or more wind turbines are associated with the same wind speed model.

Wind speed time sequences are calculated after solving the power flow and initializing wind turbine variables. To visualize these sequences, type `fm_wind(-1)` at the MATLAB prompt or use the menu *View/Plot wind speeds* in the main PSAT window. During time domain simulations, the actual wind speed values which are used for calculating the mechanical power of wind turbines are the output of a low-pass filter with time constant τ (see Fig. 21.1), in order to simulate the smoothing of high-frequency wind speed variations over the rotor surface:

$$\dot{v}_m = (\check{v}_w(t) - v_w)/\tau \quad (21.1)$$

As all other state variables, the filtered wind speeds can be plotted in the plotting GUI only after running the time domain simulation.

Wind data is defined in the global variable `Wind`, which is an instance of the class `WNclass`. Relevant properties are as follows:

1. `con`: Wind data.
2. `n`: total number of wind components.
3. `speed.time`: time vector.
4. `speed.vw`: wind speed vector.
5. `vwa`: average (initial) wind speed.
6. `vw`: indexes of state variable v_w .
7. `store`: data backup.

21.1.1 Weibull's Distribution

A common way to describe the wind speed is by means of the Weibull's distribution, which is as follows:

$$f(v_w, c, k) = \frac{k}{c^k} v_w^{k-1} e^{-(\frac{v_w}{c})^k} \quad (21.2)$$

Table 21.1: Wind Speed Data Format (`Wind.con`)

Column	Variable	Description	Unit
1	-	1 Measurement data Wind model 2 Weibull's distribution 3 Composite model 4 Mexican hat wavelet model	int
2	v_{wn}	Nominal wind speed	m/s
3	ρ	Air density	kg/m ³
4	τ	Filter time constant	s
5	Δt	Sample time for wind measurements	s
6	c	Scale factor for Weibull's distribution	-
7	k	Shape factor for Weibull's distribution	-
8	t_s^r	Starting ramp time	s
9	t_e^r	Ending ramp time	s
10	v_w^r	Ramp speed magnitude	m/s
11	t_s^g	Starting gust time	s
12	t_e^g	Ending gust time	s
13	v_w^g	Gust speed magnitude	m/s
14	h	Height of the wind speed signal	m
15	z_0	Roughness length	m
16	Δf	Frequency step	Hz
17	n	Number of harmonics	int
18	t_0	Centering time of the Mexican hat wavelet	s
19	σ	Shape factor of the Mexican hat wavelet	-

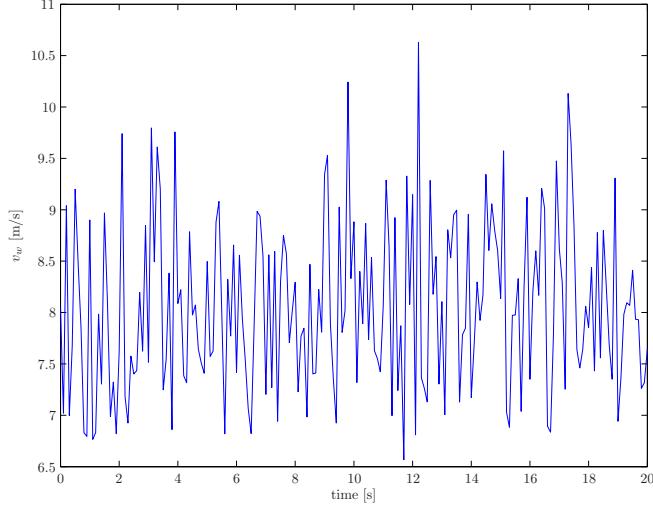


Figure 21.2: Weibull's distribution model of the wind speed.

where v_w is the wind speed and c and k are constants as defined in the wind model data matrix. Time variations $\xi_w(t)$ of the wind speed are then obtained by means of a Weibull's distribution, as follows:

$$\xi_w(t) = \left(-\frac{\ln \iota(t)}{c}\right)^{\frac{1}{k}} \quad (21.3)$$

where $\iota(t)$ is a generator of random numbers ($\iota \in [0, 1]$). Usually the shape factor $k = 2$, which leads to the Rayleigh's distribution, while $k > 3$ approximates the normal distribution and $k = 1$ gives the exponential distribution. The scale factor c should be chosen in the range $c \in (1, 10)$. Finally, the wind speed is computed setting the initial average speed v_w^a determined at the initialization step as mean speed:

$$\check{v}_w(t) = (1 + \xi_w(t) - \hat{\xi}_w)v_w^a \quad (21.4)$$

where $\hat{\xi}_w$ is the mean value of $\xi_w(t)$. An example of wind speed time evolution generated using a Weibull's distribution is depicted in Fig. 21.2.

21.1.2 Composite Wind Model

A composite wind model similar to what proposed in [135, 5] is included in PSAT. This model considers the wind as composed of four parts, as follows:

1. average and initial wind speed v_w^a ;
2. ramp component of the wind speed v_w^r ;

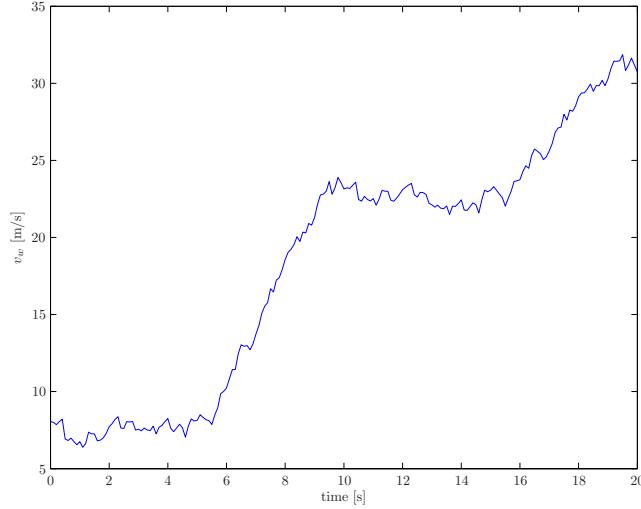


Figure 21.3: Composite model of the wind speed.

3. gust component of the wind speed v_w^g ;
4. wind speed turbulence v_w^t ;

thus the resulting wind speed \check{v}_w is:

$$\check{v}_w(t) = v_w^a + v_w^r(t) + v_w^g(t) + v_w^t(t) \quad (21.5)$$

where all components are time-dependent except for the initial average speed v_w^a . An example of wind speed time evolution generated using a composite model is depicted in Fig. 21.3.

Wind Ramp Component

The wind ramp component is defined by an amplitude A_w^r and starting and ending times, t_s^r and t_e^r respectively:

$$\begin{aligned} t < t_s^r : \quad v_w^r(t) &= 0 \\ t_s^r \leq t \leq t_e^r : \quad v_w^r(t) &= A_w^r \left(\frac{t - t_s^r}{t_e^r - t_s^r} \right) \\ t > t_e^r : \quad v_w^r(t) &= A_w^r \end{aligned} \quad (21.6)$$

Table 21.2: Roughness length z_0 for various ground surfaces [102, 117]

Ground surface	Roughness length z_0 [m]
Open sea, sand	$10^{-4} \div 10^{-3}$
Snow surface	$10^{-3} \div 5 \cdot 10^{-3}$
Mown grass, steppe	$10^{-3} \div 10^{-2}$
Long grass, rocky ground	$0.04 \div 0.1$
Forests, cities, hilly areas	$1 \div 5$

Wind Gust Component

The wind gust component is defined by an amplitude A_w^g and starting and ending times, t_s^g and t_e^g respectively:

$$\begin{aligned} t < t_s^g : \quad v_w^g(t) &= 0 \\ t_s^g \leq t \leq t_e^g : \quad v_w^g(t) &= \frac{A_w^g}{2} \left(1 - \cos \left(2\pi \frac{t - t_s^g}{t_e^g - t_s^g} \right) \right) \\ t > t_e^g : \quad v_w^g(t) &= A_w^g \end{aligned} \tag{21.7}$$

Wind Turbulence Component

The wind turbulence component is described by a power spectral density as follows:

$$S_w^t = \frac{\frac{1}{(\ln(h/z_0))^2} \ell v_w^a}{\left(1 + 1.5 \frac{\ell f}{v_w^a} \right)^{\frac{5}{3}}} \tag{21.8}$$

where f is the electrical frequency, h the wind turbine tower height, z_0 is the roughness length and ℓ is the turbulence length scale:

$$\begin{aligned} h < 30 : \quad \ell &= 20h \\ h \geq 30 : \quad \ell &= 600 \end{aligned} \tag{21.9}$$

Table 21.2 depicts roughness values z_0 for various ground surfaces.

The spectral density is then converted in a time domain cosine series as illustrated in [118]:

$$v_w^t(t) = \sum_{i=1}^n \sqrt{S_w^t(f_i) \Delta f} \cos(2\pi f_i t + \phi_i + \Delta\phi) \tag{21.10}$$

where f_i and ϕ_i are the frequency and the initial phase of the i^{th} frequency component, being ϕ_i random phases ($\phi_i \in [0, 2\pi)$). The frequency step Δf should be $\Delta f \in (0.1, 0.3)$ Hz. Finally $\Delta\phi$ is a small random phase angle introduced to avoid periodicity of the turbulence signal.

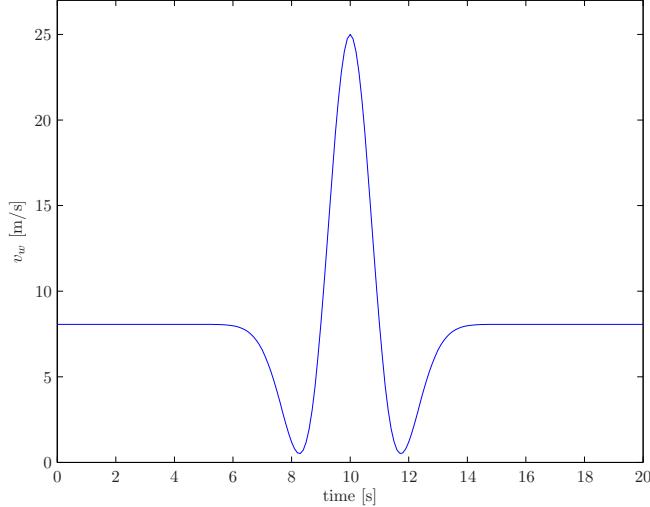


Figure 21.4: Mexican hat model of the wind speed.

21.1.3 Mexican Hat Wavelet Model

The Mexican hat wavelet is a deterministic wind gust that has been standardized by IEC [59]. The Mexican hat wavelet is the normalized second derivative of the Gaussian function, i.e., up to scale normalization, the second Hermite function. In particular, the expression used in PSAT is:

$$\ddot{v}_w(t) = v_w^a + (v_w^g - v_w^a) \left(1 - \frac{(t - t_0)^2}{\sigma^2} \right) e^{-\frac{(t-t_0)^2}{2\sigma^2}} \quad (21.11)$$

where t_0 is the centering time of the gust, σ is the gust shape factor, v_w^g is the peak wind speed value and v_w^a the average speed value. The latter is obtained from the initialization of wind turbines. Figure 21.4 shows an example of wind gust modeled through a Mexican hat wavelet with $t_0 = 10$ s, $\sigma = 1$, and $v_w^g = 25$ m/s.

21.1.4 Measurement Data

Measurement data can be used for wind speed time sequence simply by defining in the PSAT data file the field `Wind.speed(i).vw` as a two column array, where the first column is the time and the second one the wind speed in m/s, and i is the wind speed number. If no wind speed data are found in the file, the Weibull's distribution model will be used. Observe that measurement data cannot be set with a SIMULINK model. Thus one first should convert the SIMULINK model into a PSAT data file and then add the wind speed data editing the file itself.

Table 21.3: Recent wind turbines [46]

Type	Power [MW]	Diam. [m]	Height [m]	Control	Speed [rpm]
Bonus	2	86	80	GD/TS/PS	17
NEC NM 1500/72	1.5	72	98	GD/TS/PS	17.3
Nordex N-80	2.5	80	80	GD/VS/PC	19
Vestas V-80	2	80	78	GD/VS/PC	19
Enercon e-66	1.5	66	85	GD/VS/PC	22
GD	gearbox drive	DD	direct drive	VS	variable speed
TS	two speed	PC	pitch control	PS	shift pitch by stall

21.2 Wind Turbines

This section describes the three wind turbine types as implemented in PSAT: the constant speed wind turbine with squirrel cage induction generator, the variable speed wind turbine with doubly fed (wound rotor) induction generator and the direct drive synchronous generator. These configurations were chosen as they are widely used nowadays and their models are mostly based on the models discussed in [118]. Figure 21.5 depicts three wind turbines types, while Table 21.3 illustrates a few recent wind turbines data as documented in [46].

The wind turbine and generators models that are implemented in PSAT are adequate for a single machine as well as for a wind park composed of several generators (aggregate wind turbine model). For a correct definition of the wind turbine data, use the following rules:

1. The nominal power S_n is the total power in MVA of the single or aggregate wind turbine. If the wind turbine is an aggregate equivalent model of a wind park, then the number of generators $n_g > 1$. The size of each generator is about [0.5, 5] MVA (being 1 MVA the most common size), thus $n_g \in [0.2S_n, 2S_n]$. Using an inconsistent value of n_g could lead to an inadequate initialization of the wind speed.
2. Machine impedances and inertias are equivalent or mean p.u. values of the machines that compose the aggregate wind park. PSAT takes care of making the base conversions in case the wind park is composed of several machines.

21.2.1 Constant Speed Wind Turbine

The simplified electrical circuit used for the squirrel cage induction generator is the same as the one for the single-cage induction motor, depicted in Fig. 17.7, the only difference with respect to the induction motor being that the currents are positive if injected in the network. The equations are formulated in terms of the real (r) and imaginary (m) axis, with respect to the network reference angle. In a synchronously rotating reference frame, the link between the network and the

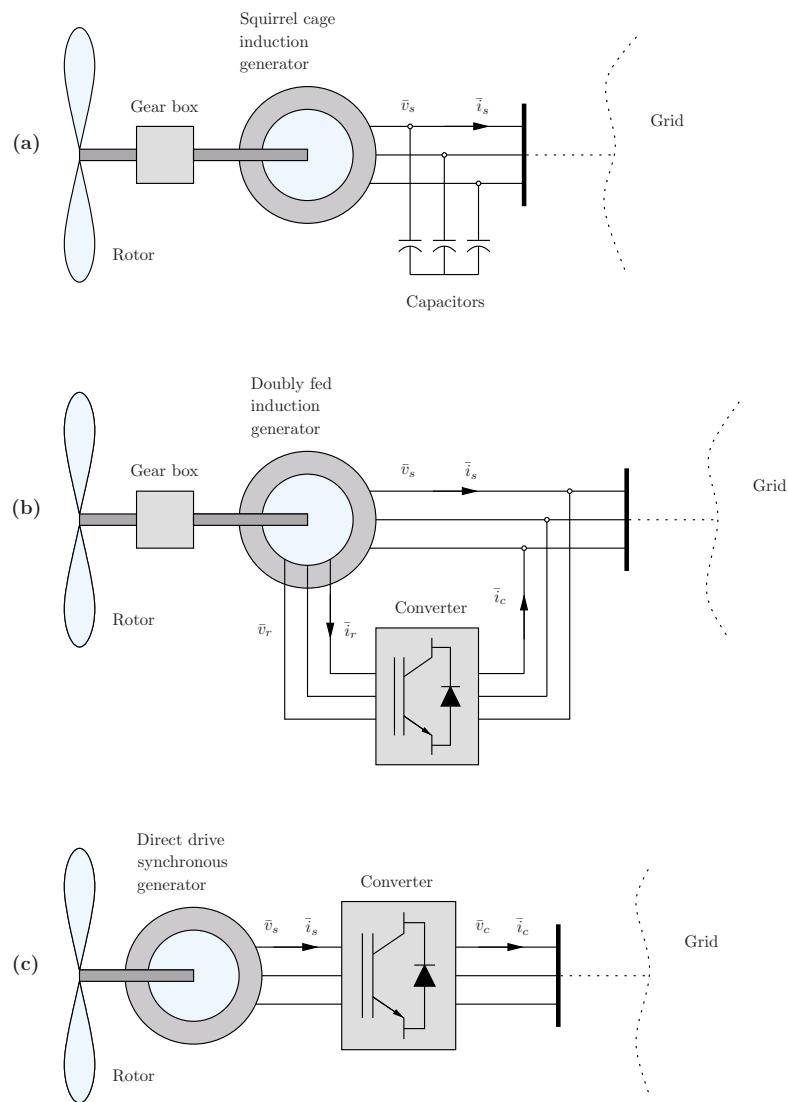


Figure 21.5: Wind turbine types. **(a)** Constant speed wind turbine with squirrel cage induction generator; **(b)** Variable speed wind turbine with doubly fed induction generator; **(c)** Variable speed wind turbine with direct drive synchronous generator.

stator machine voltages is as follows:

$$\begin{aligned} v_r &= -v \sin \theta \\ v_m &= v \cos \theta \end{aligned} \quad (21.12)$$

and the power absorptions are:

$$\begin{aligned} p &= v_r i_r + v_m i_m \\ q &= v_m i_r - v_r i_m + b_c(v_r^2 + v_m^2) \end{aligned} \quad (21.13)$$

where b_c is the fixed capacitor conductance which is determined at the initialization step. The differential equations in terms of the voltage behind the stator resistance r_s are:

$$\begin{aligned} e'_r - v_r &= r_s i_r - x' i_m \\ e'_m - v_m &= r_s i_m + x' i_r \end{aligned} \quad (21.14)$$

whereas the link between voltages, currents and state variables is as follows:

$$\begin{aligned} \dot{e}'_r &= \Omega_b(1 - \omega_m)e'_m - (e'_r - (x_0 - x')i_m)/T'_0 \\ \dot{e}'_m &= -\Omega_b(1 - \omega_m)e'_r - (e'_m + (x_0 - x')i_r)/T'_0 \end{aligned} \quad (21.15)$$

where ω_m is the rotor angular speed, and x_0 , x' and T'_0 can be obtained from the generator parameters:

$$\begin{aligned} x_0 &= x_s + x_\mu \\ x' &= x_s + \frac{x_r x_\mu}{x_r + x_\mu} \\ T'_0 &= \frac{x_r + x_\mu}{\Omega_b r_R} \end{aligned} \quad (21.16)$$

The mechanical differential equations which take into account the turbine and rotor inertiae H_t and H_m , respectively, and shaft stiffness K_s are as follows:

$$\begin{aligned} \dot{\omega}_t &= (\tau_t - K_s \gamma)/(2H_t) \\ \dot{\omega}_m &= (K_s \gamma - \tau_e)/(2H_m) \\ \dot{\gamma} &= \Omega_b(\omega_t - \omega_m) \end{aligned} \quad (21.17)$$

where ω_t is the wind turbine angular speed, and the electrical torque τ_e is defined as:

$$\tau_e = e'_r i_r + e'_m i_m \quad (21.18)$$

The mechanical torque τ_t is:

$$\tau_t = \frac{p_w}{\omega_t} \quad (21.19)$$

where p_w is the mechanical power extracted from the wind. The latter is a function of both the wind and the rotor speeds and can be approximated as follows:

$$p_w = \frac{n_g \rho}{2S_n} c_p(\lambda) A_r v_w^3 \quad (21.20)$$

where n_g is the number of machines that compose the wind park, ρ is the air density, c_p the performance coefficient or power coefficient, λ the tip speed ratio and A_r the area swept by the rotor. The speed tip ratio λ is the ratio between the blade tip speed v_{bt} and the wind upstream the rotor v_w :

$$\lambda = \frac{v_{bt}}{v_w} = \eta_{GB} \frac{2R\omega_t}{n_p v_w} \quad (21.21)$$

where η_{GB} is the gear box ratio, n_p the number of poles of the induction generator and R the rotor radius. Finally, the $c_p(\lambda)$ curve is approximated as follows:

$$c_p = 0.44 \left(\frac{125}{\lambda_i} - 6.94 \right) e^{-\frac{16.5}{\lambda_i}} \quad (21.22)$$

with

$$\lambda_i = \frac{1}{\frac{1}{\lambda} + 0.002} \quad (21.23)$$

To simulate the *tower shadow* effect, a periodic torque pulsation is added to τ_t , whose frequency depends on the rotor speed ω_t , the gear box ratio η_{GB} , and the number of blades n_b , as follows:

$$\tilde{\tau}_t = \tau_t \left(1 + 0.025 \sin \left(\eta_{GB} \frac{\Omega_b \omega_t}{n_b} t \right) \right) \quad (21.24)$$

where the torque pulsation amplitude is fixed to 0.025 according to what was presented in [2].

The constant speed wind turbine with squirrel cage induction generator is defined in the global variable **Cswt**, which is an instance of the class **CSclass**. Relevant properties are as follows:

1. **con**: constant speed wind turbine data.
2. **n**: total number of constant speed wind turbines.
3. **bus**, **vbus**: wind turbines bus voltage indexes.
4. **wind**: numbers of wind speed models to which wind turbines are connected.
5. **dat**: wind turbines parameters.
6. **omega_t**: indexes of the state variable ω_t .
7. **omega_m**: indexes of the state variable ω_m .
8. **gamma**: indexes of the state variable γ .
9. **e1r**: indexes of the state variable e'_r .
10. **e1m**: indexes of the state variable e'_m .
11. **store**: data backup.
12. **u**: connection status.

Table 21.4 depicts the data format of the constant speed wind turbine with squirrel cage induction generator.

Table 21.4: Constant Speed Wind Turbine Data Format (`Cswt.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	-	Wind speed number	int
3	S_n	Power rating	MVA
4	V_n	Voltage rating	kV
5	f_n	Frequency rating	Hz
6	r_s	Stator resistance	p.u.
7	x_s	Stator reactance	p.u.
8	r_r	Rotor resistance	p.u.
9	x_r	Rotor reactance	p.u.
10	x_μ	Magnetizing reactance	p.u.
11	H_t	Wind turbine inertia	kWs/kVA
12	H_m	Rotor inertia	kWs/kVA
13	K_s	Shaft stiffness	p.u.
14	R	Rotor radius	m
15	n_p	Number of poles	int
16	n_b	Number of blades	int
17	η_{GB}	Gear box ratio	-
18	n_g	# of machines	int
† 19	u	Connection status	{0, 1}

21.2.2 Doubly Fed Induction Generator

Steady-state electrical equations of the doubly fed induction generator are assumed, as the stator and rotor flux dynamics are fast in comparison with grid dynamics and the converter controls basically decouple the generator from the grid. As a result of these assumptions, one has:

$$\begin{aligned} v_{ds} &= -r_s i_{ds} + ((x_s + x_\mu) i_{qs} + x_\mu i_{qr}) \\ v_{qs} &= -r_s i_{qs} - ((x_s + x_\mu) i_{ds} + x_\mu i_{dr}) \\ v_{dr} &= -r_r i_{dr} + (1 - \omega_m)((x_s + x_\mu) i_{qr} + x_\mu i_{qs}) \\ v_{qr} &= -r_r i_{qr} - (1 - \omega_m)((x_s + x_\mu) i_{dr} + x_\mu i_{ds}) \end{aligned} \quad (21.25)$$

where the stator voltages are functions of the grid voltage magnitude and phase:

$$\begin{aligned} v_{ds} &= -v \sin \theta \\ v_{qs} &= v \cos \theta \end{aligned} \quad (21.26)$$

The generator active and reactive powers depend on the stator and converter currents, as follows:

$$\begin{aligned} p &= v_{ds} i_{ds} + v_{qs} i_{qs} + v_{dc} i_{dc} + v_{qc} i_{qc} \\ q &= v_{qs} i_{ds} - v_{ds} i_{qs} + v_{qc} i_{dc} - v_{dc} i_{qc} \end{aligned} \quad (21.27)$$

Due to the converter operation mode, the power injected in the grid can be written as a function of stator and rotor currents.

The converter powers on the grid side are:

$$\begin{aligned} p_c &= v_{dc} i_{dc} + v_{qc} i_{qc} \\ q_c &= v_{qc} i_{dc} - v_{dc} i_{qc} \end{aligned} \quad (21.28)$$

whereas, on the rotor side:

$$\begin{aligned} p_r &= v_{dr} i_{dr} + v_{qr} i_{qr} \\ q_r &= v_{qr} i_{dr} - v_{dr} i_{qr} \end{aligned} \quad (21.29)$$

Assuming a loss-less converter model, the active power of the converter coincides with the rotor active power, thus $p_c = p_r$. The reactive power injected into the grid can be approximated neglecting stator resistance and assuming that the d -axis coincides with the maximum of the stator flux. Therefore, the powers injected in the grid result:

$$\begin{aligned} p &= v_{ds} i_{ds} + v_{qs} i_{qs} + v_{dr} i_{dr} + v_{qr} i_{qr} \\ q &= -\frac{x_\mu v i_{dr}}{x_s + x_\mu} - \frac{v^2}{x_\mu} \end{aligned} \quad (21.30)$$

The generator motion equation is modeled as a single shaft, as it is assumed that the converter controls are able to filter shaft dynamics. For the same reason,

no tower shadow effect is considered in this model. Thus one has:

$$\begin{aligned}\dot{\omega}_m &= (\tau_m - \tau_e)/2H_m \\ \tau_e &= \psi_{ds}i_{qs} - \psi_{qs}i_{ds}\end{aligned}\quad (21.31)$$

where the link between stator fluxes and generator currents is as follows:

$$\begin{aligned}\psi_{ds} &= -((x_s + x_\mu)i_{ds} + x_\mu i_{dr}) \\ \psi_{qs} &= -((x_s + x_\mu)i_{qs} + x_\mu i_{qr})\end{aligned}\quad (21.32)$$

Thus the electrical torque τ_e results:

$$\tau_e = x_\mu(i_{qr}i_{ds} - i_{dr}i_{qs}) \quad (21.33)$$

To simplify computations, the electrical torque τ_e is approximated as follows:

$$\tau_e \approx -\frac{x_\mu vi_{qr}}{\omega_b(x_s + x_\mu)} \quad (21.34)$$

where ω_b is the system frequency rate in rad/s. The mechanical torque is:

$$\tau_m = \frac{p_w}{\omega_m} \quad (21.35)$$

being p_w the mechanical power extracted from the wind. The latter is a function of the wind speed v_w , the rotor speed ω_m and the pitch angle θ_p . The mechanical power p_w can be approximated as follows:

$$p_w = \frac{n_g \rho}{2S_n} c_p(\lambda, \theta_p) A_r v_w^3 \quad (21.36)$$

where n_g is the number of machines that compose the wind park and other parameters and variables are the same as in (21.20) and the speed tip ratio λ is defined as in (21.21). The $c_p(\lambda, \theta_p)$ curve is approximated as follows:

$$c_p = 0.22 \left(\frac{116}{\lambda_i} - 0.4\theta_p - 5 \right) e^{-\frac{12.5}{\lambda_i}} \quad (21.37)$$

with

$$\frac{1}{\lambda_i} = \frac{1}{\lambda + 0.08\theta_p} - \frac{0.035}{\theta_p^3 + 1} \quad (21.38)$$

Converter dynamics are highly simplified, as they are fast with respect to the electromechanical transients. Thus, the converter is modeled as an ideal current source, where i_{qr} and i_{dr} are state variables and are used for the rotor speed control and the voltage control respectively, which are depicted in Figures 21.6 and 21.7. Differential and algebraic equations for the converter currents are as follows:

$$\dot{i}_{qr} = \left(-\frac{x_s + x_\mu}{x_\mu v} p_w^*(\omega_m)/\omega_m - i_{qr} \right) \frac{1}{T_e} \quad (21.39)$$

$$\begin{aligned}\dot{i}_{dr} &= K_V(v - v_{\text{ref}}) - v/x_\mu - i_{dr} \\ 0 &= v_{\text{ref}}^0 - v_{\text{ref}} + v_s^{\text{POD}}\end{aligned}\quad (21.40)$$

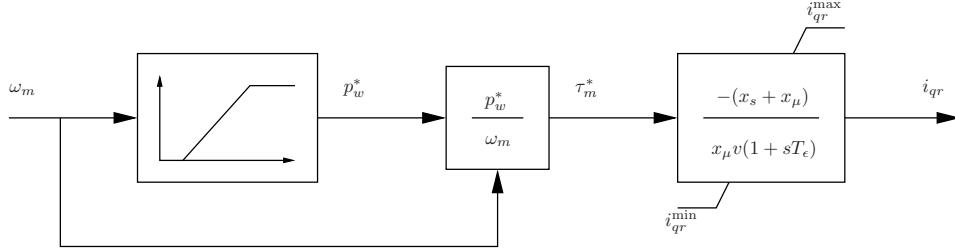


Figure 21.6: Rotor speed control scheme.

where v_{ref}^0 is the initial reference voltage; v_s^{POD} is an additional signal of the POD (see Section 18.6) ; and $p_w^*(\omega_m)$ is the power-speed characteristic which roughly optimizes the wind energy capture and is calculated using the current rotor speed value, as follows:

$$p_w^*(\omega_m) = \begin{cases} 0 & \text{if } \omega_m < 0.5 \\ 2\omega_m - 1 & \text{if } 0.5 \leq \omega_m \leq 1 \\ 1 & \text{if } \omega_m > 1 \end{cases} \quad (21.41)$$

Thus, the rotor speed control only has effect for sub-synchronous speeds. Both the speed and voltage controls undergo anti-windup limiters in order to avoid converter over-currents. Rotor current limits are computed based on active and reactive limits, and assuming bus voltage $v \approx 1$ as follows:

$$\begin{aligned} i_{qr}^{\text{max}} &\approx -\frac{x_s + x_\mu}{x_\mu} p^{\text{min}} & (21.42) \\ i_{qr}^{\text{min}} &\approx -\frac{x_s + x_\mu}{x_\mu} p^{\text{max}} \\ i_{dr}^{\text{max}} &\approx -\frac{x_s + x_\mu}{x_\mu} q^{\text{min}} - \frac{x_s + x_\mu}{x_\mu^2} \\ i_{dr}^{\text{min}} &\approx -\frac{x_s + x_\mu}{x_\mu} q^{\text{max}} - \frac{x_s + x_\mu}{x_\mu^2} \end{aligned}$$

Finally the pitch angle control is illustrated in Fig. 21.8 and described by the differential equation:

$$\dot{\theta}_p = (K_p \phi(\omega_m - \omega_{\text{ref}}) - \theta_p)/T_p \quad (21.43)$$

where ϕ is a function which allows varying the pitch angle set point only when the difference $(\omega_m - \omega_{\text{ref}})$ exceeds a predefined value $\pm\Delta\omega$. The pitch control works only for super-synchronous speeds. An anti-windup limiter locks the pitch angle to $\theta_p = 0$ for sub-synchronous speeds.

The wind turbine with doubly fed induction generator is defined in the global variable `Dfig`, which is an instance of the class `DFclass`. relevant properties are as follows:

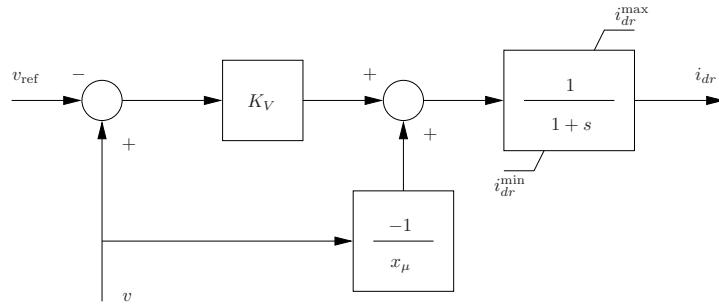


Figure 21.7: Voltage control scheme of the doubly-fed induction generator.

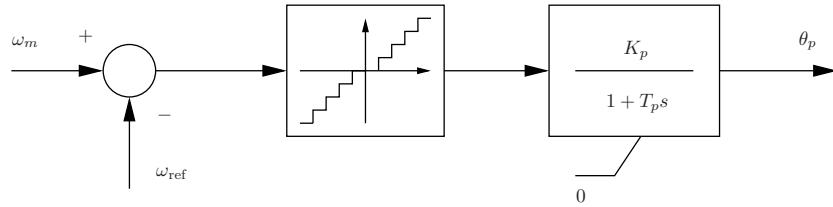


Figure 21.8: Pitch angle control scheme.

1. **con**: doubly fed induction generator data.
2. **n**: total number of doubly fed induction generators.
3. **bus**, **vbus**: wind turbine bus voltage indexes.
4. **wind**: numbers of wind speed models to which generators are connected.
5. **dat**: generator parameters.
6. **omega_m**: indexes of the state variable ω_m .
7. **theta_p**: indexes of the state variable θ_p .
8. **idr**: indexes of the state variable i_{dr} .
9. **iqr**: indexes of the state variable i_{qr} .
10. **pwa**: indexes of the algebraic variable p_w^* .
11. **vref**: indexes of the algebraic variable v_{ref} .
12. **store**: data backup.
13. **u**: connection status.

Table 21.5 depicts the data format of the wind turbine with doubly fed induction generator.

Table 21.5: Doubly Fed Induction Generator Data Format (Dfig.con)

Column	Variable	Description	Unit
1	-	Bus number	int
2	-	Wind speed number	int
3	S_n	Power rating	MVA
4	V_n	Voltage rating	kV
5	f_n	Frequency rating	Hz
6	r_s	Stator resistance	p.u.
7	x_s	Stator reactance	p.u.
8	r_r	Rotor resistance	p.u.
9	x_r	Rotor reactance	p.u.
10	x_μ	Magnetizing reactance	p.u.
11	H_m	Rotor inertia	kWs/kVA
12	K_p	Pitch control gain	-
13	T_p	Pitch control time constant	s
14	K_V	Voltage control gain	-
15	T_ϵ	Power control time constant	s
16	R	Rotor radius	m
17	n_p	Number of poles	int
18	n_b	Number of blades	int
19	η_{GB}	Gear box ratio	-
20	p^{\max}	Maximum active power	p.u.
21	p^{\min}	Minimum active power	p.u.
22	q^{\max}	Maximum reactive power	p.u.
23	q^{\min}	Minimum reactive power	p.u.
24	n_g	# of machines	int
† 25	u	Connection status	{0, 1}

21.2.3 Direct Drive Synchronous Generator

Steady-state electrical equations of the direct drive synchronous generator (DDSG) are assumed, as the stator and rotor flux dynamics are fast in comparison with grid dynamics and the converter controls basically decouple the generator from the grid. As a result of these assumptions, one has:

$$\begin{aligned} v_{ds} &= -r_s i_{ds} + \omega_m x_q i_{qs} \\ v_{qs} &= -r_s i_{qs} - \omega_m (x_d i_{ds} - \psi_p) \end{aligned} \quad (21.44)$$

where a permanent field flux ψ_p is used here to represent the rotor circuit. The active and reactive power of the generator are as follows:

$$\begin{aligned} p_s &= v_{ds} i_{ds} + v_{qs} i_{qs} \\ q_s &= v_{qs} i_{ds} - v_{ds} i_{qs} \end{aligned} \quad (21.45)$$

while the active and reactive powers injected into the grid depend only on the grid side currents of the converter:

$$\begin{aligned} p_c &= v_{dc} i_{dc} + v_{qc} i_{qc} \\ q_c &= v_{qc} i_{dc} - v_{dc} i_{qc} \end{aligned} \quad (21.46)$$

where the converter voltages are functions of the grid voltage magnitude and phase, as follows:

$$\begin{aligned} v_{dc} &= -v \sin \theta \\ v_{qc} &= v \cos \theta \end{aligned} \quad (21.47)$$

Assuming a loss-less converter and a power factor equal to 1 (permanent magnet rotor), the output powers of the generator becomes:

$$\begin{aligned} p_c &= p_s \\ q_s &= 0 \end{aligned} \quad (21.48)$$

The reactive power injected in the grid is controlled by means of the converter direct current i_{dc} . From (21.46), (21.47) and (21.48), one obtains:

$$\begin{aligned} i_{qc} &= \frac{p_s + vi_{dc} \sin \theta}{v \cos \theta} \\ q_c &= v(i_{dc} \cos \theta + i_{qc} \sin \theta) \end{aligned} \quad (21.49)$$

Equations (21.48) and (21.49) are the algebraic equations of the DDSG model, while v , θ and i_{ds} and i_{qc} are the algebraic variables.

The generator motion equation is modeled as a single shaft, as it is assumed that the converter controls are able to filter shaft dynamics. For the same reason, no tower shadow effect is considered in this model. Thus one has:

$$\begin{aligned} \dot{\omega}_m &= (\tau_m - \tau_e)/2H_m \\ \tau_e &= \psi_{ds} i_{qs} - \psi_{qs} i_{ds} \end{aligned} \quad (21.50)$$

where the link between stator fluxes and generator currents is as follows:

$$\begin{aligned}\psi_{ds} &= -x_d i_{ds} + \psi_p \\ \psi_{qs} &= -x_q i_{qs}\end{aligned}\quad (21.51)$$

The mechanical torque and power are modeled as in the doubly fed induction motor, thus equations from (21.35) to (21.38) apply.

Converter dynamics are highly simplified, as they are fast with respect to the electromechanical transients. Thus, the converter is modeled as an ideal current source, where i_{qs} and i_{dc} are state variables and are used for the rotor speed control and the reactive power control and the voltage control, respectively. The voltage control is depicted in Fig. 21.9. Differential equations of the converter currents are as follows:

$$\begin{aligned}\dot{i}_{qs} &= (i_{qsref} - i_{qs})/T_{ep} \\ \dot{i}_{dc} &= (K_V(v_{ref} - v) - i_{dc})/T_V\end{aligned}\quad (21.52)$$

where

$$i_{qsref} = \frac{p_w^*(\omega_m)}{\omega_m(\psi_p - x_d i_{ds})} \quad (21.53)$$

where $p_w^*(\omega_m)$ is the power-speed characteristic which roughly optimizes the wind energy capture and which is calculated using the current rotor speed value (see Fig. ??). It is assumed that $p_w^* = 0$ if the $\omega_m < 0.5$ p.u. and that $p_w^* = 1$ p.u. if $\omega_m > 1$ p.u. Thus, the rotor speed control only has effect for sub-synchronous speeds. Both the speed and voltage controls undergo anti-windup limiters in order to avoid converter over-currents. For the same reason, the reactive power q_c and the converter quadrature current undergo windup limiters. Current and reactive power limits are approximated as follows:

$$\begin{aligned}i_{qs}^{\max}, i_{qc}^{\max} &= p^{\max} \\ i_{qs}^{\min}, i_{qc}^{\min} &= p^{\min} \\ i_{dc}^{\max}, q_c^{\max} &= q^{\max} \\ i_{dc}^{\min}, q_c^{\min} &= q^{\min}\end{aligned}\quad (21.54)$$

where p^{\max} , p^{\min} , q^{\max} , and q^{\min} are set by the user in the DDSG data. Finally the pitch angle control is illustrated in Fig. 21.8 and described by the differential equation (21.43).

The wind turbine with direct drive synchronous generator is defined in the global variable `Ddsg`, which is an instance of the class `DDclass`. relevant properties are as follows:

1. `con`: direct drive synchronous generator data.
2. `n`: total number of direct drive synchronous generators.
3. `bus`, `vbus`: wind turbine bus voltage indexes.

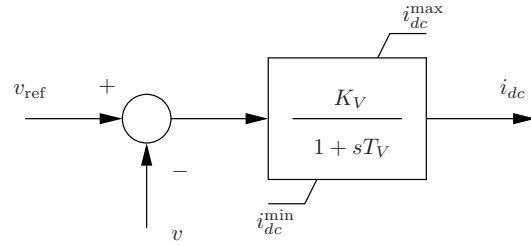


Figure 21.9: Voltage control scheme of the direct drive synchronous generator.

4. `wind`: numbers of wind speed models to which generators are connected.
5. `dat`: generator parameters.
6. `omega_m`: indexes of the state variable ω_m .
7. `theta_p`: indexes of the state variable θ_p .
8. `iqs`: indexes of the state variable i_{qs} .
9. `pwa`: indexes of the algebraic variable p_w^* .
10. `ids`: indexes of the algebraic variable i_{ds} .
11. `idc`: indexes of the algebraic variable i_{dc} .
12. `store`: data backup.
13. `u`: connection status.

Table 21.5 depicts the data format of the wind turbine with direct drive synchronous generator.

Table 21.6: Direct Drive Synchronous Generator Data Format (Ddsg.con)

Column	Variable	Description	Unit
1	-	Bus number	int
2	-	Wind speed number	int
3	S_n	Power rating	MVA
4	V_n	Voltage rating	kV
5	f_n	Frequency rating	Hz
6	r_s	Stator resistance	p.u.
7	x_d	d -axis reactance	p.u.
8	x_q	q -axis reactance	p.u.
9	ψ_p	Permanent field flux	p.u.
10	H_m	Rotor inertia	kWs/kVA
11	K_p	Pitch control gain	-
12	T_p	Pitch control time constant	s
13	K_V	Voltage control gain	-
14	T_V	Voltage control time constant	s
15	$T_{\epsilon p}$	Active power control time constant	s
16	$T_{\epsilon q}$	Reactive power control time constant (<i>not used</i>)	s
17	R	Rotor radius	m
18	n_p	Number of poles	int
19	n_b	Number of blades	int
20	η_{GB}	Gear box ratio	-
21	p^{\max}	Maximum active power	p.u.
22	p^{\min}	Minimum active power	p.u.
23	q^{\max}	Maximum reactive power	p.u.
24	q^{\min}	Minimum reactive power	p.u.
25	n_g	# of machines	int
† 26	u	Connection status	{0, 1}

Chapter 22

Other Models

This chapter describes additional components useful to represent particular dynamic phenomena. These are synchronous machine dynamic shaft, sub-synchronous resonance generator model, and solid oxide fuel cell.

22.1 Dynamic Shaft

A dynamic mass-spring model is used for defining the shaft of the synchronous machine. Figure 22.1 depicts the shaft scheme (springs are in solid black). The rotor mass is dashed since it is not actually part of the model. The dynamic shaft has to be connected to a synchronous machine. Turbine governors can be connected to dynamic shafts.

Table 22.1 depicts the dynamic shaft data format. The state variables are initialized after solving the power flow, and a PV or a slack generator are required at the machine bus. A nominal frequency ($\omega = 1$) is assumed when the shaft speeds are initialized. The power and frequency ratings of the shaft are inherited from the synchronous machine associated with the shaft.

The complete set of differential equations which describe the dynamic shaft is as follows:

$$\begin{aligned}\dot{\delta}_{\text{HP}} &= \Omega_b(\omega_{\text{HP}} - 1) \\ \dot{\omega}_{\text{HP}} &= (\tau_m - D_{\text{HP}}(\omega_{\text{HP}} - 1) - D_{12}(\omega_{\text{HP}} - \omega_{\text{IP}}) \\ &\quad + K_{\text{HP}}(\delta_{\text{IP}} - \delta_{\text{HP}}))/M_{\text{HP}} \\ \dot{\delta}_{\text{IP}} &= \Omega_b(\omega_{\text{IP}} - 1) \\ \dot{\omega}_{\text{IP}} &= (-D_{\text{IP}}(\omega_{\text{IP}} - 1) - D_{12}(\omega_{\text{IP}} - \omega_{\text{HP}}) - D_{23}(\omega_{\text{IP}} - \omega_{\text{LP}}) \\ &\quad + K_{\text{HP}}(\delta_{\text{HP}} - \delta_{\text{IP}}) + K_{\text{IP}}(\delta_{\text{LP}} - \delta_{\text{IP}}))/M_{\text{IP}} \\ \dot{\delta}_{\text{LP}} &= \Omega_b(\omega_{\text{LP}} - 1) \\ \dot{\omega}_{\text{LP}} &= (-D_{\text{LP}}(\omega_{\text{LP}} - 1) - D_{23}(\omega_{\text{LP}} - \omega_{\text{IP}}) - D_{34}(\omega_{\text{LP}} - \omega) \\ &\quad + K_{\text{IP}}(\delta_{\text{IP}} - \delta_{\text{LP}}) + K_{\text{LP}}(\delta - \delta_{\text{LP}}))/M_{\text{LP}}\end{aligned}\tag{22.1}$$

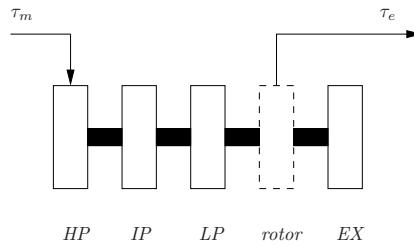


Figure 22.1: Synchronous machine mass-spring shaft model.

$$\begin{aligned}
 \dot{\delta} &= \Omega_b(\omega - 1) \\
 \dot{\omega} &= (-\tau_e - D(\omega - 1) - D_{34}(\omega - \omega_{LP}) - D_{45}(\omega - \omega_{EX}) \\
 &\quad + K_{LP}(\delta_{LP} - \delta) + K_{EX}(\delta_{EX} - \delta))/M \\
 \dot{\delta}_{EX} &= \Omega_b(\omega_{EX} - 1) \\
 \dot{\omega}_{EX} &= (-D_{EX}(\omega_{EX} - 1) - D_{45}(\omega_{EX} - \omega) \\
 &\quad + K_{EX}(\delta - \delta_{EX}))/M_{EX}
 \end{aligned}$$

Dynamic shafts are defined in the global variable `Mass`, which is an instance of the class `DSclass`. Relevant properties are as follows:

1. `con`: data of the `Mass` components.
2. `syn`: indexes of generators to which the shafts are connected.
3. `n`: total number of dynamic shafts.
4. `delta_HP`: indexes of the state variable δ_{HP} .
5. `omega_HP`: indexes of the state variable ω_{HP} .
6. `delta_IP`: indexes of the state variable δ_{IP} .
7. `omega_IP`: indexes of the state variable ω_{IP} .
8. `delta_LP`: indexes of the state variable δ_{LP} .
9. `omega_LP`: indexes of the state variable ω_{LP} .
10. `delta_EX`: indexes of the state variable δ_{EX} .
11. `omega_EX`: indexes of the state variable ω_{EX} .
12. `store`: data backup.
13. `u`: connection status.

Table 22.1: Dynamic Shaft Data Format (Mass.con)

Column	Variable	Description	Unit
1	-	Synchronous machine number	int
2	M_{HP}	High pressure turbine inertia	kWs/kVA
3	M_{IP}	Intermediate pressure turbine inertia	kWs/kVA
4	M_{LP}	Low pressure turbine inertia	kWs/kVA
5	M_{EX}	Exciter inertia	kWs/kVA
6	D_{HP}	High pressure turbine damping	p.u.
7	D_{IP}	Intermediate pressure turbine damping	p.u.
8	D_{LP}	Low pressure turbine damping	p.u.
9	D_{EX}	Exciter damping	p.u.
10	D_{12}	High-interm. pressure turbine damping	p.u.
11	D_{23}	Intermed.-low pressure turbine damping	p.u.
12	D_{34}	Low pressure turbine-rotor damping	p.u.
13	D_{45}	Rotor-exciter damping	p.u.
14	K_{HP}	High pressure turbine angle coeff.	p.u.
15	K_{IP}	Intermed. pressure turbine angle coeff.	p.u.
16	K_{LP}	Low pressure turbine angle coeff.	p.u.
17	K_{EX}	Exciter angle coefficient	p.u.
† 18	u	Connection status	{0, 1}

22.2 Sub-synchronous Resonance Model

Figure 22.2 depicts a generator with shaft dynamics and compensated line, which represents a simple model for studying the sub-synchronous resonance (SSR) problem. The shaft dynamics are similar to what described in Section 22.1 and are modeled as high, intermediate and low pressure turbine masses, exciter mass and machine rotor.

This is one of the simplest models [140] which presents the sub-synchronous resonance (SSR) phenomenon, a well known problem of undamped oscillations that may occur when the transmission line to which the machine is connected is compensated by a series capacitor [48, 107, 106].

The dynamics of the RLC circuit cannot be neglected since the line presents two modes whose frequency can be roughly estimated as $\Omega_b(1 \pm \sqrt{x_C/x_L})$. For typical values of the inductive and capacitive reactances, the lower of these two frequencies can be close to one of the mechanical oscillations of the generator shaft. Thus, beyond a certain value of the compensation level, the machine may experiment a negative damping of one of the mechanical modes that results in dangerous stresses on the shaft. This phenomenon can be also described in terms of the bifurcation theory [91, 92].

The model used for representing the machine and the line is the same used in [145]. It presents five electrical state variables (i_d , i_q , i_f , v_{dc} , v_{qc}) which can be

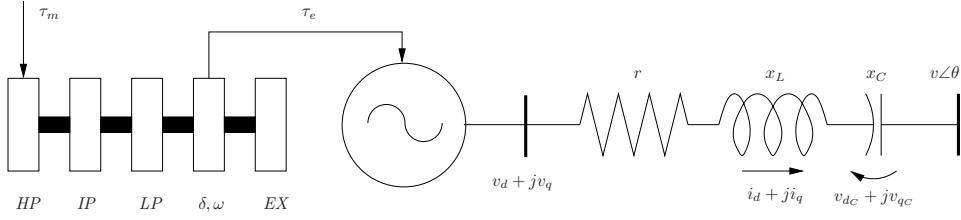


Figure 22.2: Generator with dynamic shaft and compensated line.

determined by the machine differential equations:

$$\begin{aligned}\dot{\psi}_f &= (v_{fd} - i_f)/T'_{d0} \\ \dot{\psi}_d &= \Omega_b(r_a i_d + \omega \psi_q + v_d) \\ \dot{\psi}_q &= \Omega_b(r_a i_q - \omega \psi_d + v_q)\end{aligned}\quad (22.2)$$

the line differential equations:

$$\begin{aligned}\dot{i}_d &= \Omega_b(i_q + (v_d - ri_d - v_{d_c} - V \sin(\delta - \theta))/x_L) \\ \dot{i}_q &= \Omega_b(-i_d + (v_q - ri_q - v_{q_c} - V \cos(\delta - \theta))/x_L) \\ \dot{v}_{d_c} &= \Omega_b(x_C i_d + v_{q_c}) \\ \dot{v}_{q_c} &= \Omega_b(x_C i_q - v_{d_c})\end{aligned}\quad (22.3)$$

along with the algebraic constraints that link the time derivatives of the generator fluxes and of the line currents:

$$\begin{aligned}\dot{\psi}_f &= \dot{i}_f - (x_d - x'_d) \dot{i}_d \\ \dot{\psi}_d &= \dot{i}_f - x_d \dot{i}_d \\ \dot{\psi}_q &= -x_q \dot{i}_q\end{aligned}\quad (22.4)$$

Finally, a five mass system is used for describing the shaft dynamics:

$$\begin{aligned}\dot{\delta}_{HP} &= \Omega_b(\omega_{HP} - 1) \\ \dot{\omega}_{HP} &= (\tau_m - D_{HP}(\omega_{HP} - 1) + K_{HP}(\delta_{IP} - \delta_{HP}))/M_{HP} \\ \dot{\delta}_{IP} &= \Omega_b(\omega_{IP} - 1) \\ \dot{\omega}_{IP} &= (-D_{IP}(\omega_{IP} - 1) + K_{HP}(\delta_{HP} - \delta_{IP}) \\ &\quad + K_{IP}(\delta_{LP} - \delta_{IP}))/M_{IP} \\ \dot{\delta}_{LP} &= \Omega_b(\omega_{LP} - 1) \\ \dot{\omega}_{LP} &= (-D_{LP}(\omega_{LP} - 1) + K_{IP}(\delta_{IP} - \delta_{LP}) \\ &\quad + K_{LP}(\delta - \delta_{LP}))/M_{LP} \\ \dot{\delta} &= \Omega_b(\omega - 1) \\ \dot{\omega} &= (-\tau_e - D(\omega - 1) + K_{LP}(\delta_{LP} - \delta))\end{aligned}\quad (22.5)$$

$$\begin{aligned}\dot{\delta}_{\text{EX}} &= \Omega_b(\omega_{\text{EX}} - 1) \\ \dot{\omega}_{\text{EX}} &= (-D_{\text{EX}}(\omega_{\text{EX}} - 1) + K_{\text{EX}}(\delta - \delta_{\text{EX}})) / M_{\text{EX}} \\ &\quad + K_{\text{EX}}(\delta_{\text{EX}} - \delta)) / M\end{aligned}$$

where the electrical torque is $\tau_{ae} = \psi_d i_q - \psi_q i_d$. The algebraic equations for the power injections

$$\begin{aligned}p &= -vi_d \sin(\delta - \theta) - vi_q \cos(\delta - \theta) \\ q &= -vi_d \cos(\delta - \theta) + vi_q \sin(\delta - \theta)\end{aligned}\tag{22.6}$$

complete the model. In the implemented code, the field reactance x_f , the field resistance r_f and the d -axis reactance x_{ad} are used instead of x'_d and T'_{d0} , with the following relationships:

$$\begin{aligned}T'_{d0} &= \frac{x_f}{\Omega_b r_f} \\ x'_d &= x_d - x_{ad}\end{aligned}\tag{22.7}$$

The sub-synchronous resonance generator models are defined in the global variable `SSR`, which is an instance of the class `SRclass`. Relevant properties are as follows:

1. `con`: SSR data.
2. `bus`, `vbus`: SSR bus voltage indexes.
3. `n`: total number of SSRs.
4. `Id`: indexes of the state variable i_d .
5. `Iq`: indexes of the state variable i_q .
6. `If`: indexes of the state variable i_f .
7. `Edc`: indexes of the state variable v_{dc} .
8. `Eqc`: indexes of the state variable v_{qc} .
9. `Tm`: mechanical torque τ_m .
10. `Efd`: field voltage v_{fd} .
11. `delta_HP`: indexes of the state variable δ_{HP} .
12. `omega_HP`: indexes of the state variable ω_{HP} .
13. `delta_IP`: indexes of the state variable δ_{IP} .
14. `omega_IP`: indexes of the state variable ω_{IP} .

15. `delta_LP`: indexes of the state variable δ_{LP} .
16. `omega_LP`: indexes of the state variable ω_{LP} .
17. `delta`: indexes of the state variable δ .
18. `omega`: indexes of the state variable ω .
19. `delta_EX`: indexes of the state variable δ_{EX} .
20. `omega_EX`: indexes of the state variable ω_{EX} .
21. `store`: data backup.
22. `u`: connection status.

The SSR data format is depicted in Table 22.2. The SSR state variables are initialized after solving the power flow and either a PV or a slack generator is needed at the SSR bus.

22.3 Solid Oxide Fuel Cell

A Solid Oxide Fuel Cell (SOFC) model is included in PSAT based on what was proposed in [100], [146], [54], and [67].¹ Figure 22.3 depicts the fuel cell scheme, which is based on the following equations:

$$\begin{aligned}\dot{p}_{H_2} &= ((q_{H_2} - 2K_r I_{dc})/K_{H_2} - p_{H_2})/T_{H_2} \\ \dot{p}_{H_2O} &= (2K_r I_{dc}/K_{H_2O} - p_{H_2O})/T_{H_2O} \\ \dot{p}_{O_2} &= ((q_{H_2}/r_{HO} - K_r I_{dc})/k_{O_2} - p_{O_2})/T_{O_2} \\ \dot{q}_{H_2} &= (2K_r I_{dc}/U_{opt} - q_{H_2})/T_f \\ \dot{V}_{dc} &= (-V_{dc} - R_{dc} I_{dc} + N_0(E_0 + \frac{R\Theta}{2F} \ln(p_{H_2} \sqrt{p_{O_2}}/p_{H_2O}))) / T_\epsilon\end{aligned}\tag{22.8}$$

where R is the gas constant ($R = 8.314 \text{ [J/(mol K)]}$), F is the Faraday constant ($F = 96487 \text{ [C/mol]}$), T the absolute gas temperature, and T_ϵ is a “small” time constant which does not affects the fuel cell dynamics. The fuel cell current I_{dc} can be subjected to a constant power control:

$$\dot{I}_{dc} = (S_n p_{ref}/(V_{dc}) - I_{dc})/T_e\tag{22.9}$$

or a to constant current control:

$$\dot{I}_{dc} = (S_n p_{ref}/(V_{dc}^0) - I_{dc})/T_e\tag{22.10}$$

where V_{dc}^0 is the initial fuel cell DC voltage. If the input signal exceeds the dynamic limits proportional to the fuel flow, one has:

$$\dot{I}_{dc} = (\frac{U^{\lim} q_{H_2}}{2K_r} - I_{dc})/T_e\tag{22.11}$$

¹This model was realized in 2002 in collaboration with Valery Knyazkin, Royal Institute of Technology, Sweden.

Table 22.2: SSR Data Format (SSR.con)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	V_n	Voltage rating	kV
4	f_n	Frequency rating	Hz
5	x_d	d -axis synchronous reactance	p.u.
6	x_q	q -axis synchronous reactance	p.u.
7	r_a	Armature resistance	p.u.
8	x_{ad}	d -axis reactance	p.u.
9	r	Line resistance	p.u.
10	x_L	Line inductive reactance	p.u.
11	x_C	Line capacitive reactance	p.u.
12	r_f	Field resistance	p.u.
13	x_f	Field reactance	p.u.
14	M_{HP}	High pressure turbine inertia	kWs/kVA
15	M_{IP}	Intermediate pressure turbine inertia	kWs/kVA
16	M_{LP}	Low pressure turbine inertia	kWs/kVA
17	M	Rotor inertia	kWs/kVA
18	M_{EX}	Exciter inertia	kWs/kVA
19	D_{HP}	High pressure turbine damping	p.u.
20	D_{IP}	Intermediate pressure turbine damping	p.u.
21	D_{LP}	Low pressure turbine damping	p.u.
22	D	Rotor damping	p.u.
23	D_{EX}	Exciter damping	p.u.
24	K_{HP}	High pressure turbine angle coeff.	p.u.
25	K_{IP}	Intermed. pressure turbine angle coeff.	p.u.
26	K_{LP}	Low pressure turbine angle coeff.	p.u.
27	K_{EX}	Exciter angle coefficient	p.u.
† 28	u	Connection status	{0, 1}

where U^{lim} is the maximum or the minimum fuel utilization (U^{max} , U^{min}). The connection with the network is assumed to be realized by means of an ideal inverter and a transformer with reactance x_T , as depicted in Fig. 22.4. The ac voltage is regulated by means of the inverter modulating amplitude m , as follows:

$$\dot{m} = (K_m(v_{\text{ref}} - v) - m)/T_m \quad (22.12)$$

The amplitude control has anti-windup limiters and is depicted in Fig. 22.5.

The DC power of the fuel cell ($P_{dc} = V_{dc}I_{dc}$) is considered to be the real power injected in the network ($p = P_{dc}/P_b$). Thus the link with the ac network is as follows:

$$\begin{aligned} p &= \frac{v_t v}{x_T} \sin(\theta_t - \theta) = V_{dc}I_{dc}/P_b \\ q &= \frac{v_t v}{x_T} \cos(\theta_t - \theta) - \frac{v^2}{x_T} \end{aligned} \quad (22.13)$$

where $v_t = kmV_{dc}/V_b$, $k = \sqrt{3/8}$. Thus, one has:

$$\theta_t = \theta + \arcsin\left(\frac{x_T I_{dc} V_b / P_b}{kmv}\right) \quad (22.14)$$

and, finally:

$$q = -\frac{v^2}{x_T} + \frac{vv_t}{x_T} \sqrt{\left(1 - \left(\frac{x_T I_{dc} V_b / S_b}{kmv}\right)^2\right)} \quad (22.15)$$

The reference voltage v_{ref} and the initial value of the inverter amplitude m_0 are computed based on the power flow solution, as follows:

$$\begin{aligned} m_0 &= \frac{x_T}{v^0 k V_{dc} / V_b} \sqrt{p^0{}^2 + \left(q^0 + \frac{v^0{}^2}{x_T}\right)^2} \\ v_{\text{ref}} &= v^0 + m_0 / K_m \end{aligned} \quad (22.16)$$

where v^0 , p^0 and q^0 are the PV generator voltage, active power and reactive power respectively. Observe that to be properly initialized, the fuel cell needs a PV generator connected at the same bus (slack generators are not allowed).

SOFCS are defined in the global variable `Sofc`, which is an instance of the class `FCclass`. Relevant properties are as follows:

1. `con`: Solid Oxide Fuel Cell data.
2. `bus`, `vbus`: SOFC bus voltage indexes.
3. `n`: total number of SOFCs.
4. `Ik`: indexes of the state variable I_{dc} .
5. `Vk`: indexes of the state variable V_{dc} .

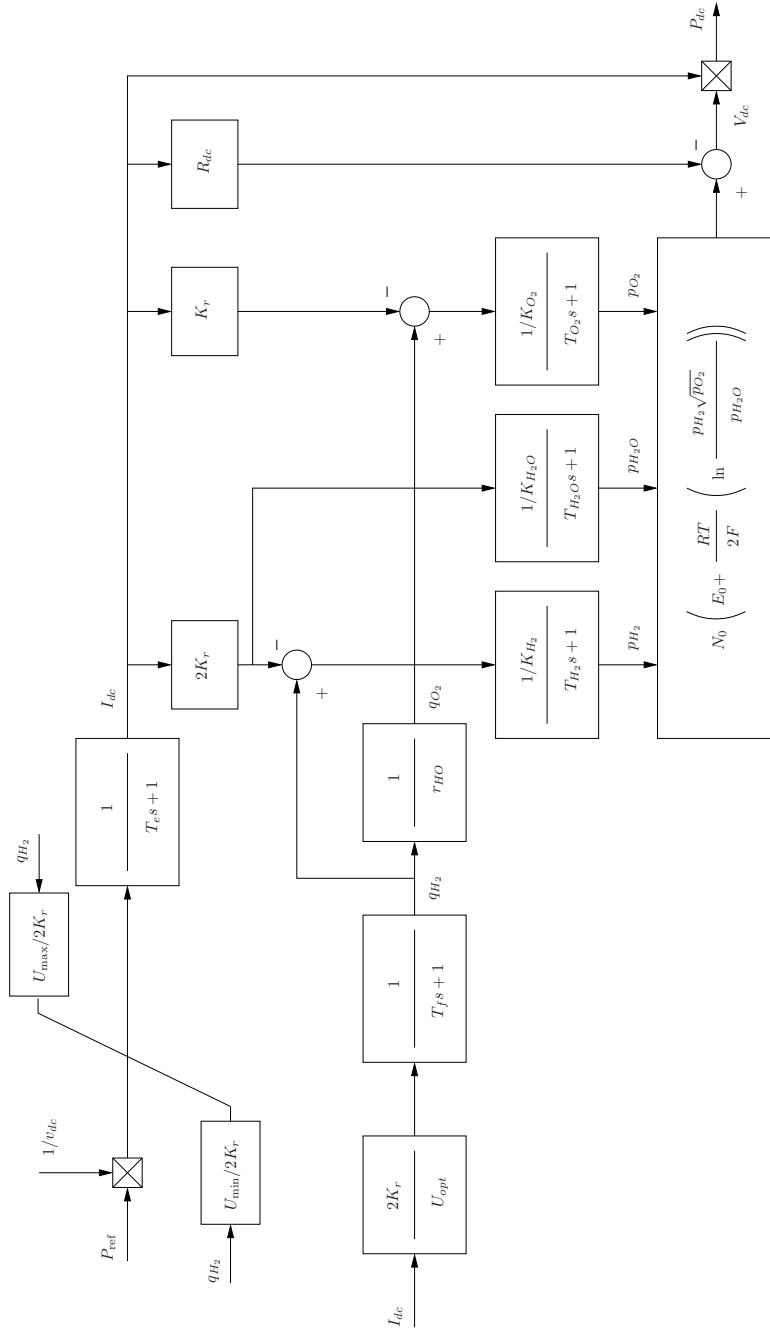


Figure 22.3: Solid Oxide Fuel Cell scheme.

Table 22.3: Solid Oxide Fuel Cell Data Format (`Sofc.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MW
3	V_n	Voltage rating	kV
4	T_e	Electrical response time	s
5	T_{H_2}	Response time for hydrogen flow	s
6	K_{H_2}	Valve molar constant for hydrogen	-
7	K_r	Constant	-
8	T_{H_2O}	Response time for water flow	s
9	K_{H_2O}	Valve molar constant for water	-
10	T_{O_2}	Response time for oxygen flow	s
11	K_{O_2}	Valve molar constant for oxygen	-
12	r_{HO}	Ratio of hydrogen to oxygen	-
13	T_f	Fuel processor response time	s
14	U_{opt}	Optimal fuel utilization	-
15	U^{\max}	Maximum fuel utilization	-
16	U^{\min}	Minimum fuel utilization	-
17	R_{dc}	Ohmic losses	Ω
18	N_0	Number of cells in series in the stack	p.u.
19	E_0	Ideal standard potential	V
20	Θ	Gas Absolute temperature	K
† 21	p_{ref}	Reference power	p.u.
† 22	v_{ref}	Reference ac voltage	p.u.
† 23	P_b	dc base power	MW
† 24	V_b	dc base voltage	kV
25	-	Control mode (1) current, (0) power	int
26	x_T	Transformer reactance	p.u.
27	K_m	Gain of the voltage control loop	p.u.
28	T_m	Time constant of the voltage control loop	s
29	m^{\max}	Maximum modulating amplitude	p.u./p.u.
30	m^{\min}	Minimum modulating amplitude	p.u./p.u.
† 31	u	Connection status	{0, 1}

Note: fields marked with a † are not set by the user.

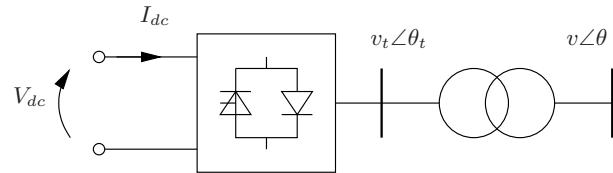


Figure 22.4: Solid Oxide Fuel Cell connection with the ac grid.

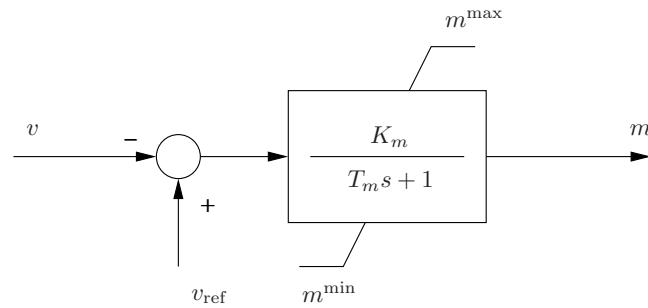
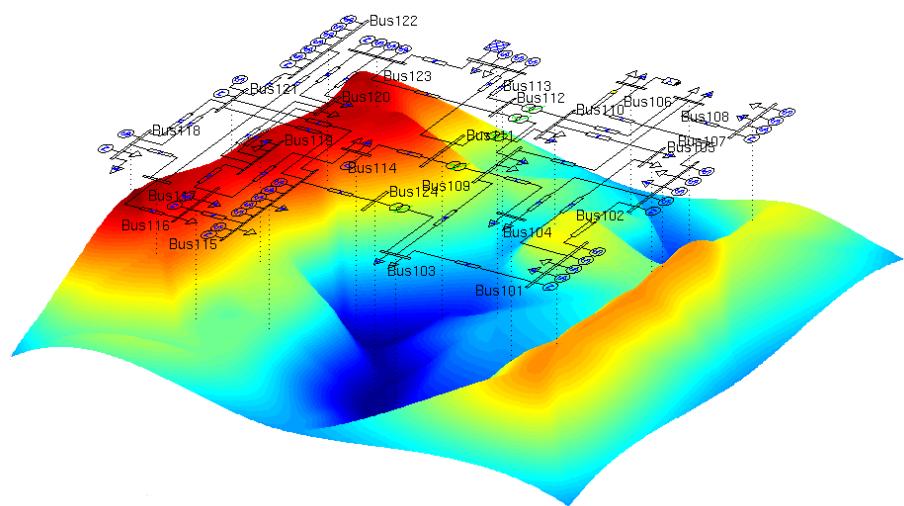


Figure 22.5: Control of the ac voltage for the Solid Oxide Fuel Cell.

6. pH2: indexes of the state variable p_{H_2} .
7. pH2O: indexes of the state variable p_{H_2O} .
8. pO2: indexes of the state variable p_{O_2} .
9. qH2: indexes of the state variable q_{H_2} .
10. m: indexes of the state variable m .
11. store: data backup.
12. u: connection status.

Part IV

CAD



Chapter 23

Network Design

This chapter describes the graphic library for network design which is built in SIMULINK and contains all components defined in the toolbox. The interaction between PSAT and the SIMULINK models is also briefly discussed. Finally, Section 23.4 depicts the SIMULINK models of three test systems used in this documentation, i.e. 14, 9 and 6-bus test systems.

23.1 Simulink Library

Figure 23.1 depicts the main frame of the PSAT SIMULINK library, which is defined in the file `fm_lib.md1`, whereas following Figures 23.2, 23.3, 23.4, 23.5, 23.6, 23.7, 23.8, 23.9, 23.10, 23.11, 23.12, and 23.13 illustrate the complete set of SIMULINK blocks for network design, which are grouped as follows: connections, power flow data, OPF & CPF data, faults & breakers, measurements, loads, machines, controls, regulating transformers, FACTS, wind turbines and other models respectively.

The SIMULINK library uses of Physical Model Components (PMCs), thus allowing bidirectional connections. Physical connectors are represented by means of circles while directional control connections are indicated by an arrow.

Running time domain simulations from the SIMULINK model menus **has no effect**, since no SIMULINK dynamic model is associated with PSAT blocks. Only blocks contained in the PSAT library can be used for building the network.

23.2 Extracting Data from Simulink Models

The SIMULINK models are used only as a graphical user interfaces. Other SIMULINK features, such as the time domain simulation, are not used by PSAT. After completing the network model, one has to extract the data from the model and create a PSAT data file. This operation is performed by the function `fm_sim` that is automatically called when a SIMULINK file is loaded as data file. Files created from

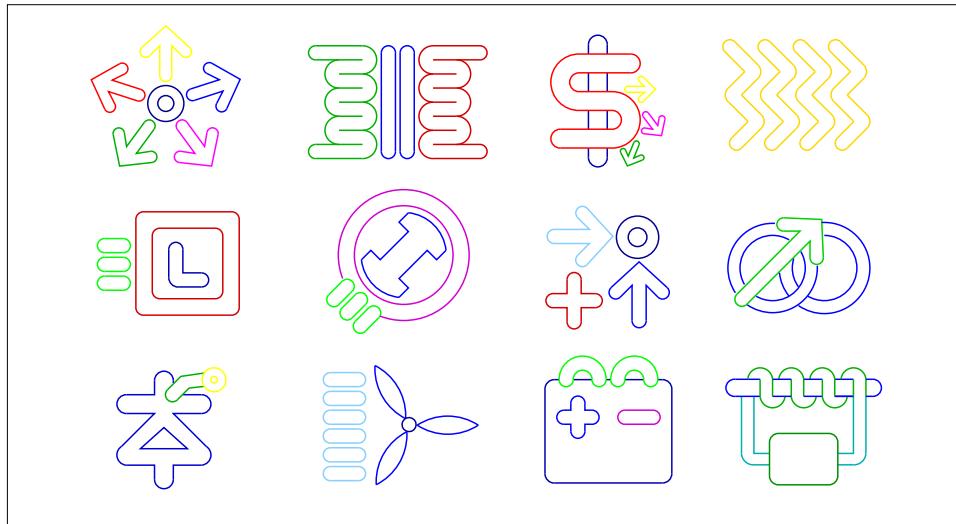


Figure 23.1: Simulink library: Main Window.

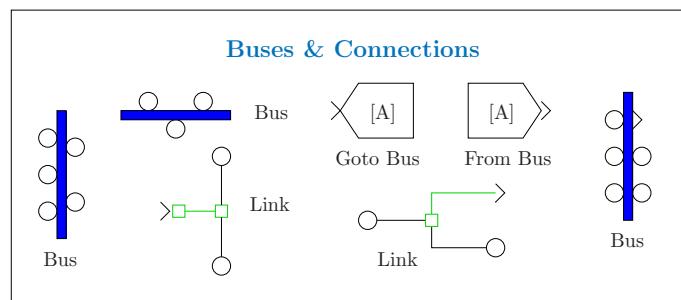


Figure 23.2: Simulink library: Connections.

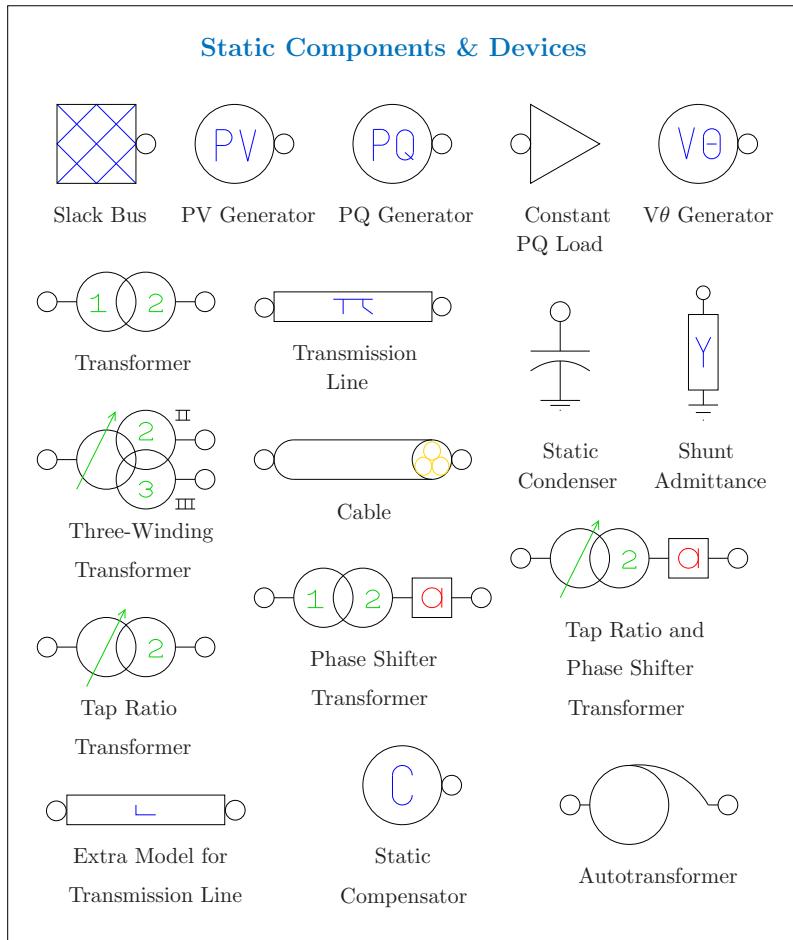


Figure 23.3: Simulink library: Power Flow data.

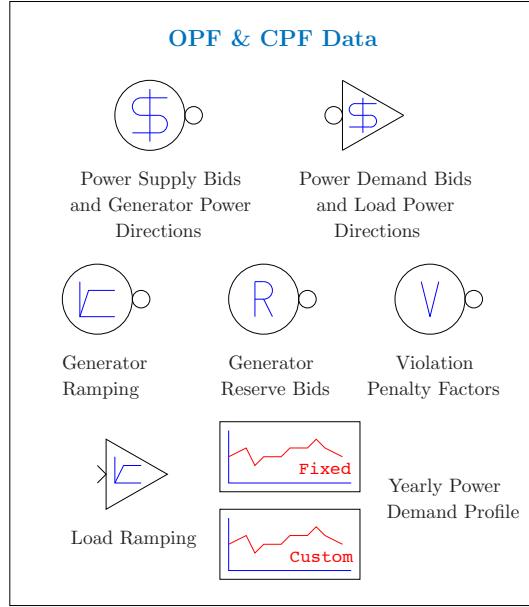


Figure 23.4: Simulink library: OPF & CPF data.

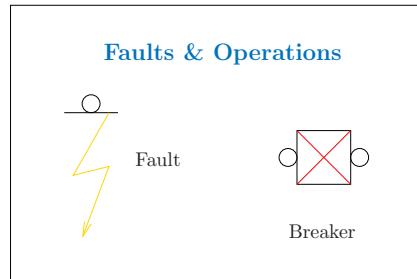


Figure 23.5: Simulink library: Faults & Breakers.

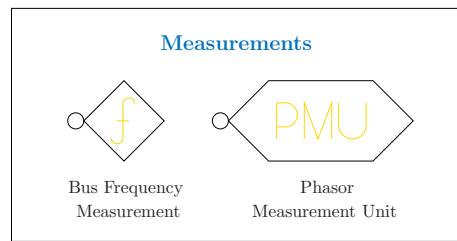


Figure 23.6: Simulink library: Measurements.

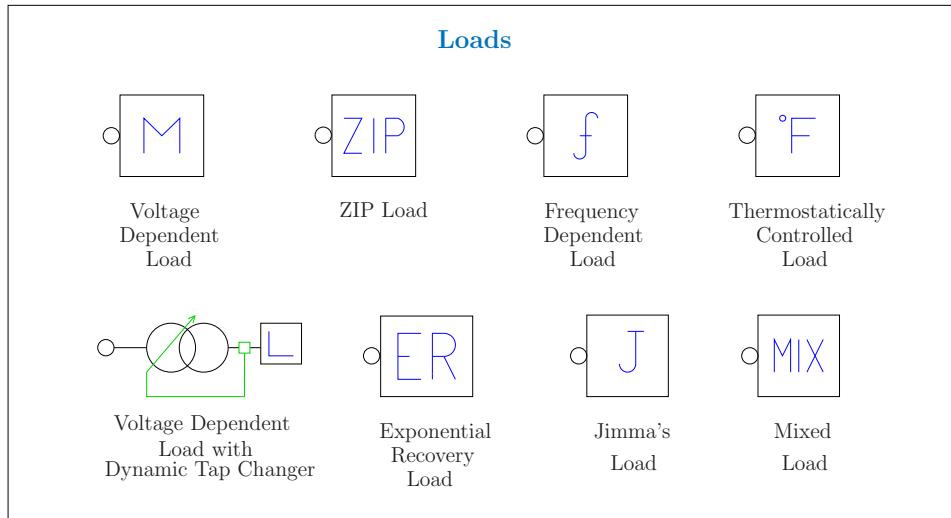


Figure 23.7: Simulink library: Loads.

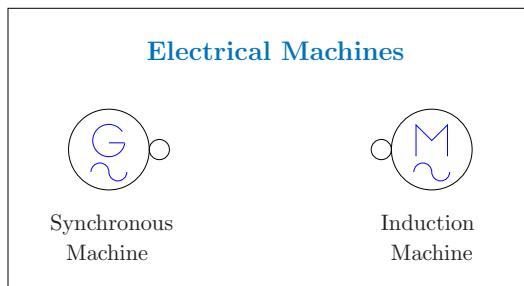


Figure 23.8: Simulink library: Machines.

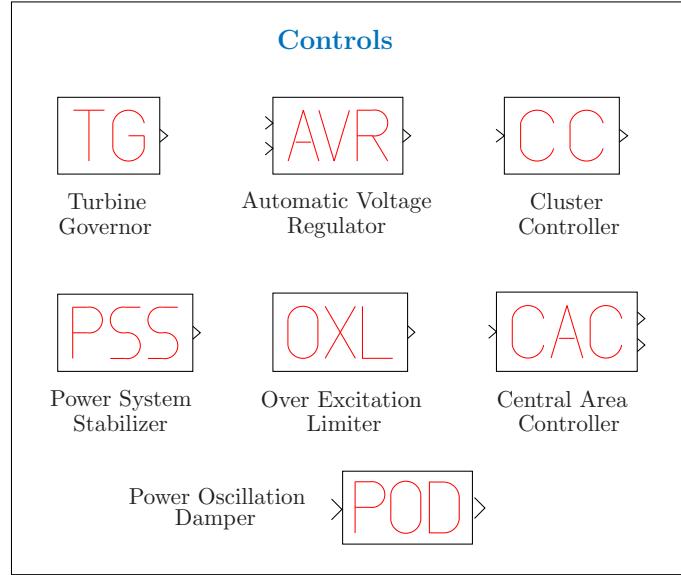


Figure 23.9: Simulink library: Regulators.

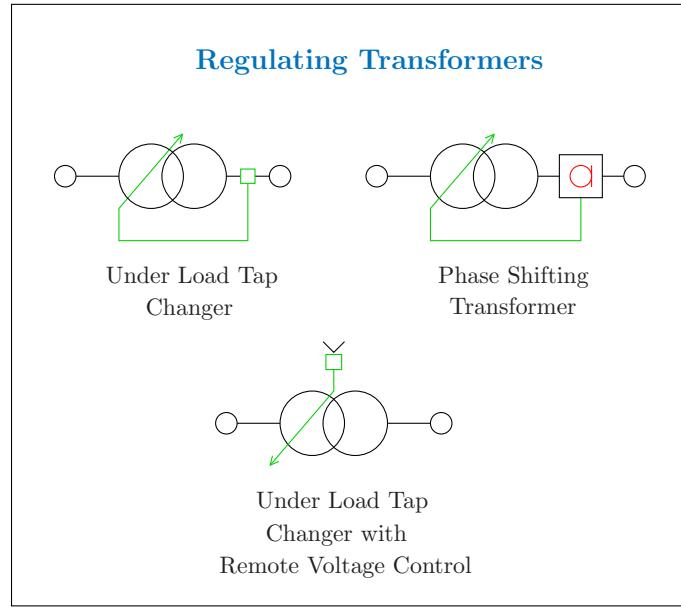


Figure 23.10: Simulink library: Regulating Transformers.

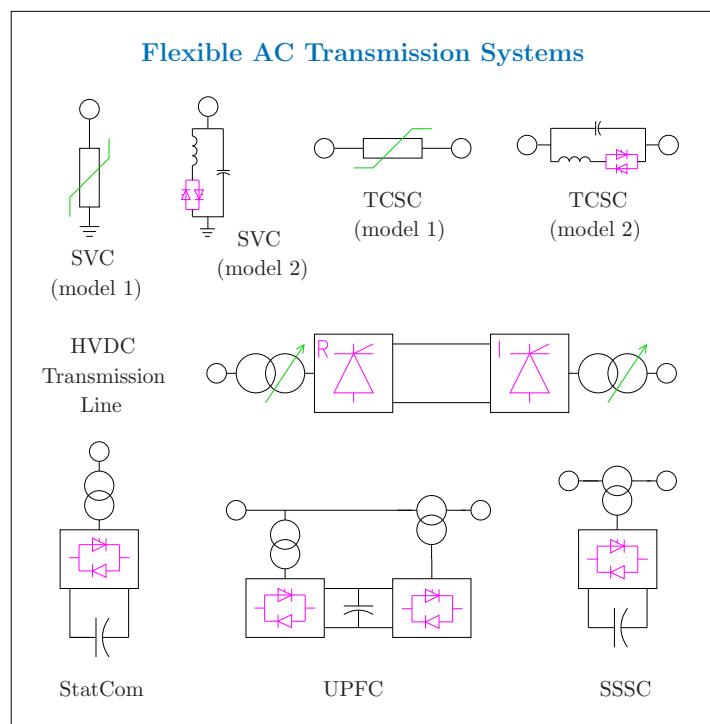


Figure 23.11: Simulink library: FACTS controllers.

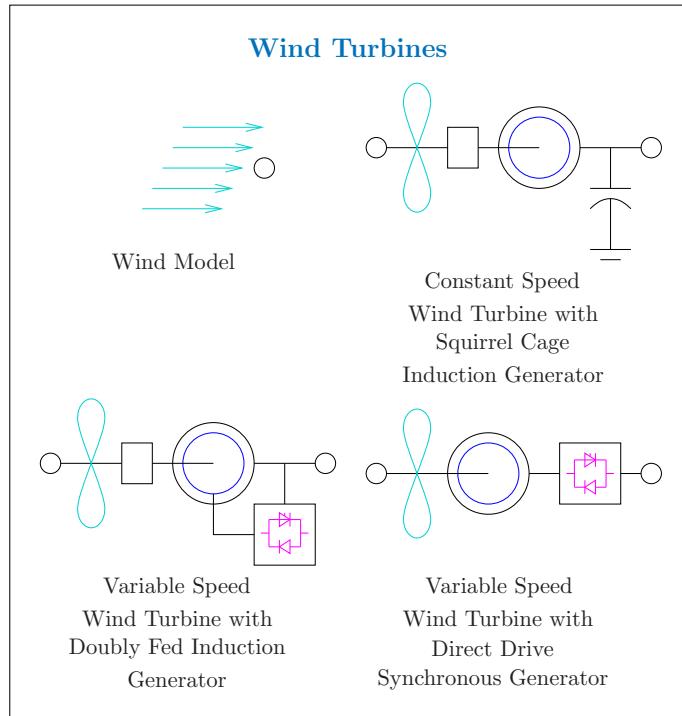


Figure 23.12: Simulink library: Wind Turbines.

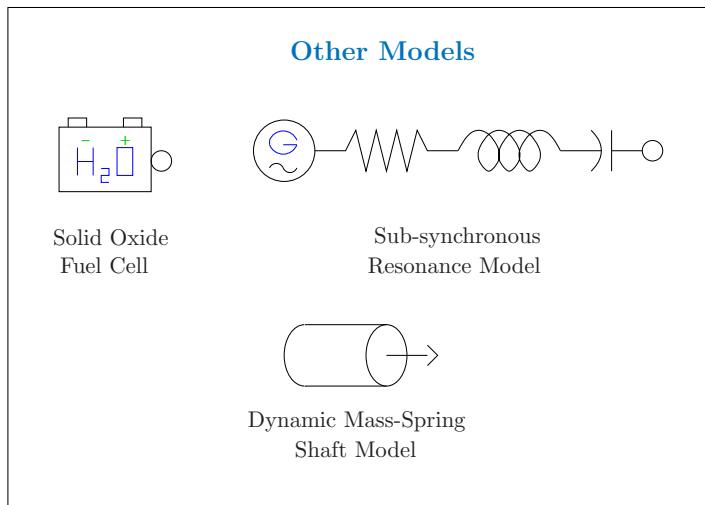


Figure 23.13: Simulink library: Other models.

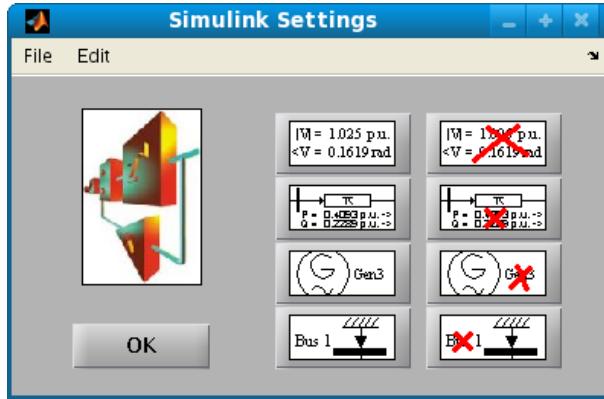


Figure 23.14: GUI for Simulink model settings.

SIMULINK models are marked with the flag (`mdl`).¹

When the loaded data file is generated from a SIMULINK model, a check of the model status is performed each time the power flow routine is launched. If the model has been changed, the data are extracted again from the model.

23.3 Displaying Results in Simulink Models

After solving the power flow, it is possible to display bus voltage and power flow values within the SIMULINK model of the currently loaded system. The GUI associated with this utility is depicted in Fig. 23.14 and is available in the menu *Edit/Simulink Model Settings* of the main window. Finally, SIMULINK models can be exported to Encapsulated Post Script files by clicking on the SIMULINK logo or using the menu *File/Export Network to EPS*. This utility allows removing the annoying black connection dots and arrows from the resulting `.eps` file (see examples depicted in the next Section 23.4).

23.4 Examples

Figures 23.15, 23.16 and 23.17 depict the SIMULINK models of the 9-bus, 14-bus and 6-bus test systems.² Figures 23.16 and 23.17 have been obtained using the export utility provided by PSAT that allows removing the black connection dots and arrows. Figure 23.16 depicts also the bus voltage report generated using the GUI for SIMULINK settings.

¹It is possible to convert a SIMULINK model without actually loading the data file, using the *Edit/Simulink Model Conversion* menu in the main window.

²The models are available in the sub-folder `tests` of the main PSAT folder.

WSCC 3-machine, 9-bus system (Copyright 1977)

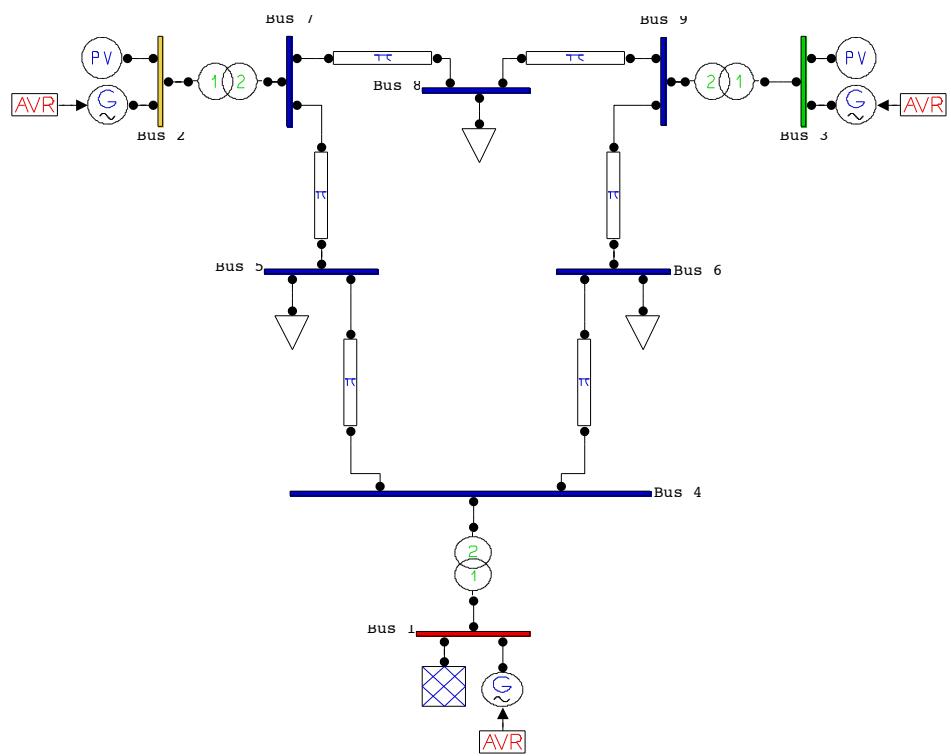


Figure 23.15: Simulink model of the WSCC 3-generator 9-bus test system.

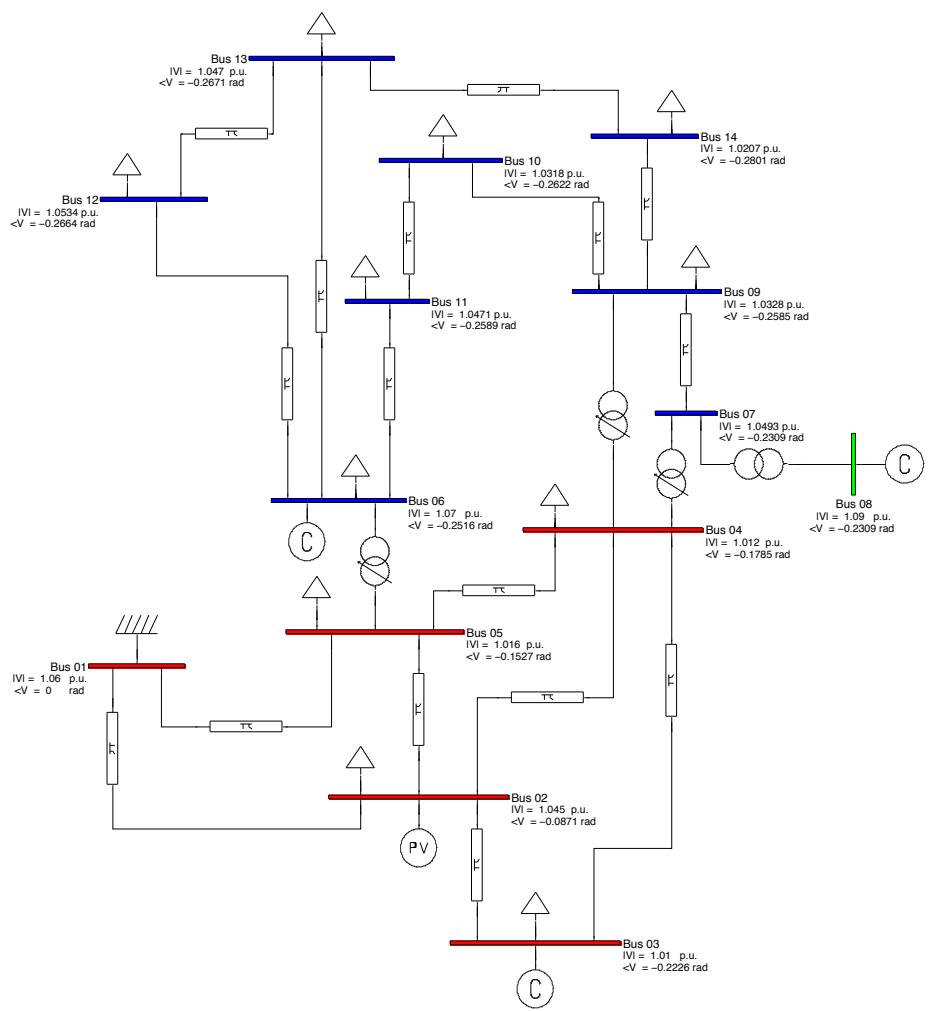


Figure 23.16: Simulink model of the IEEE 14-bus test system.

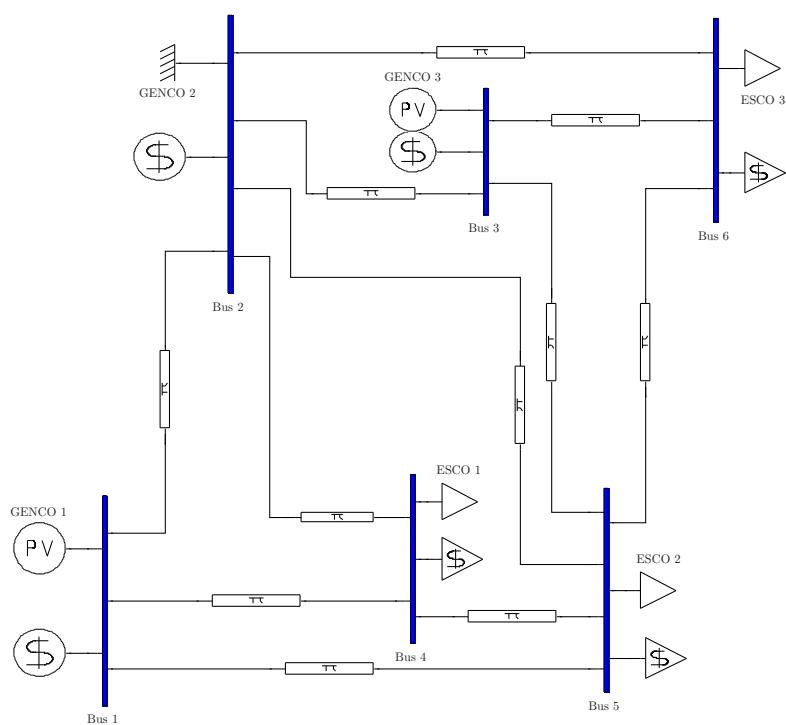


Figure 23.17: Simulink model of the 6-bus test system.

Chapter 24

Block Usage

This chapter describes how to use and connect blocks of the PSAT-SIMULINK library.

24.1 Block Connections

A well-formed PSAT SIMULINK model is a set of interconnected blocks with the following properties:

1. all connections are “allowed”;
2. all connections are “feasible”.

The first property depends on PSAT internal structures and routines, while the latter depends on mathematical or physical issues. In some cases, not allowed connections will result in error messages when compiling the data file from the SIMULINK model, while infeasible connections will typically cause singularities or inconsistent results when running PSAT routines.

A connection can be allowed but not be feasible (e.g. a slack bus and a PV generator with different desired voltages connected at the same bus). In other cases, one connection could be feasible in theory but is not allowed by PSAT (e.g. two or more PQ loads connected to the same bus).

PSAT takes generally care of not allowed connections.¹ The user should check for possible infeasible conditions. Following Sections 24.2 and 24.3 explains how to set up SIMULINK models with all allowed connections, i.e. models which will result in working PSAT data files. When possible, hints to avoid infeasible conditions are provided as well.

In this chapter, blocks are subdivided in two main groups: *standard* and *non-standard*. Standard blocks must be connected only to buses, while nonstandard blocks directly or indirectly depends on blocks other than buses. Observe that well formed models must contain **only** blocks that are provided with the PSAT SIMULINK library.

¹There is still some work to do on this issue.

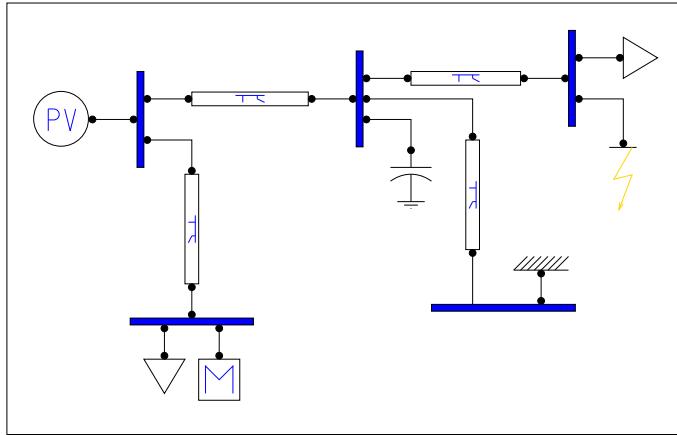


Figure 24.1: Examples of standard blocks of the PSAT SIMULINK Library.

24.2 Standard Blocks

Standard blocks only need to be connected to one bus for each input/output port. Blocks that do not follow this rule are described in the following Section 24.3. Some examples of standard blocks are depicted in Fig. 24.1.

In most cases, any number of the same standard block can be connected to the same bus, with the **only** exceptions of slack generators,² PV generators, and constant PQ loads (see Figs. 24.2 and 24.3). PSAT assumes that these blocks are unique for each bus. Connecting more than one slack bus, more than one PV generator, or more than one PQ load to the same bus would lead to unpredictable results. However, PSAT will display an error message and will not try to solve the power flow. Future versions of PSAT could include warning messages in case of other not allowed or infeasible combinations of multiple blocks being connected to the same bus.

Observe that connecting several components to the same bus, although permitted, can be sometimes inconsistent from the mathematical point of view. For example connecting one PV and one slack generator at the same bus or two under load tap changers in parallel may lead to unpredictable results or to singularities (see Fig. 24.4). This kind of inconsistency cannot be easily checked automatically. A particular care should be devoted to avoid infeasible constraints.

²The number of slack generators may be greater than one. This may occur if one defines two or more independent networks within the same SIMULINK model. However this usage is not recommended, since not all routines have been checked with a multiple network test case.

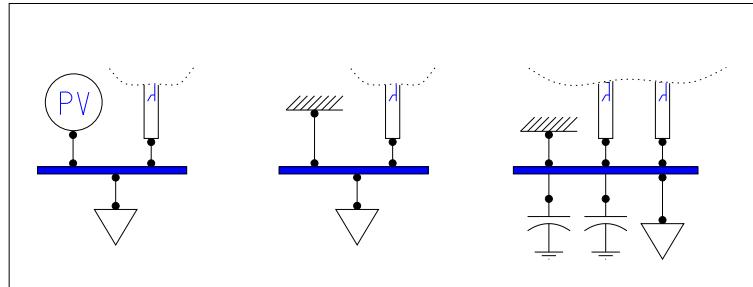


Figure 24.2: Examples of allowed connections of slack generators, PV generators and PQ loads.

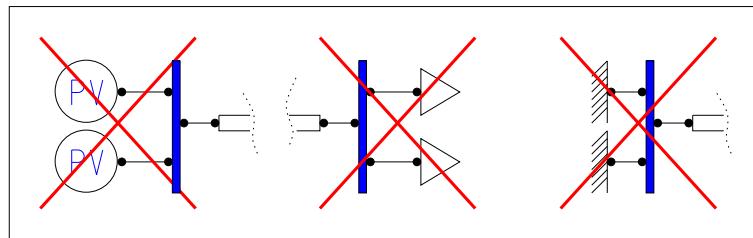


Figure 24.3: Not allowed connection of slack generators, PV generators and PQ loads.

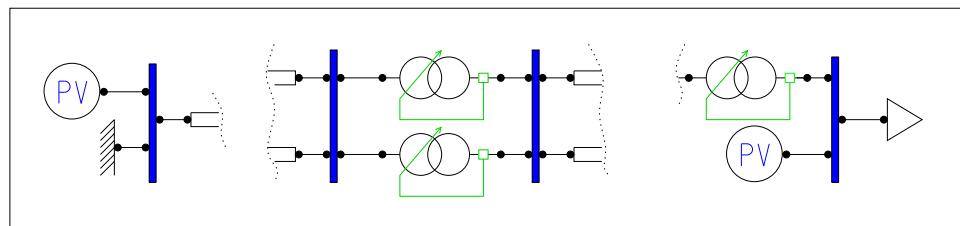


Figure 24.4: Infeasible or “likely” infeasible block connections.

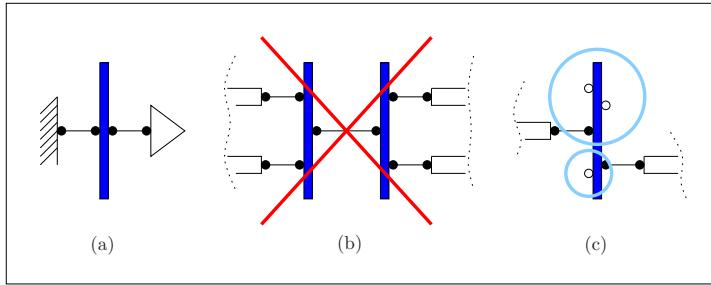


Figure 24.5: *Bus* block usage. (a) Minimal working network; (b) Not allowed bus connections; (c) Unused bus ports are allowed but not recommended.

24.3 Nonstandard Blocks

PSAT blocks are nonstandard if they cannot be directly connected to buses (this is the case of all synchronous machine regulators), need the presence of other blocks to work properly, or have input/output signals. Another way to define nonstandard blocks could be “dependent” blocks, as their usage depends on variables and parameters of other components and/or devices inserted in the network. Following subsections describes the usage of all nonstandard blocks.

24.3.1 Buses

Bus blocks are the basic elements of each model. A PSAT network has to contain at least one bus. Observe that *Bus* blocks cannot be connected directly one to another. The number of input and output ports is variable. It is not mandatory to use all ports, but the habit of leaving unused bus ports is not recommended. To avoid SIMULINK overflows as a consequence of typing errors, the maximum number of input and output ports is limited to 10 for each. This value can be changed by modifying the function `fm_inout.m`. Figure 24.5 illustrates the bus block usage.

24.3.2 Links

The *Link* block is a special kind of connection which is used **only** within a Secondary Voltage Regulation control system. See Section 24.3.10 for details.

24.3.3 Breakers

Breaker blocks works only when connected to one line and one bus. The relative position with respect to the line does not matter. Breakers works only with transmission lines and transformers and **cannot** be used as a means to disconnect devices. Use the device status u instead. Figure 24.6 illustrates the usage of breaker blocks.

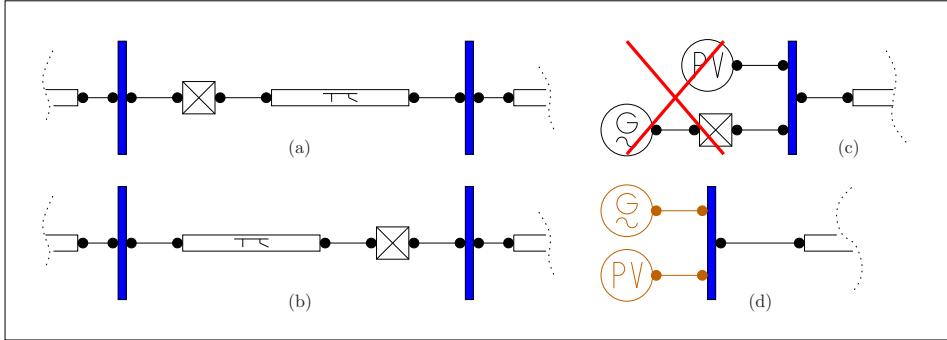


Figure 24.6: *Breaker* block usage. **(a)** Correct usage of a breaker block; **(b)** Same as case (a); **(c)** Not allowed usage of a breaker to disconnect a synchronous machine; **(d)** Usage of device stati. When off-line, blocks are displayed in orange.

24.3.4 Power Supplies and Demands

Supply blocks must be connected to one bus and need either one PV or one slack generator connected to the same bus. *Demand* blocks must be connected to one bus and need one PQ load connected to the same bus. Figure 24.7 illustrates Supply and Demand block usage.

24.3.5 Generator Ramp

Generator Ramp blocks must be connected to Supply blocks. Supply block mask allows having zero or one input port. The input port is needed only when connecting the ramp block; it is not recommended to leave unused input ports in Supply blocks. Figure 24.8 illustrates the ramp block usage. Ramp data only have effects when used with the PSAT-GAMS interface (multi-period and unit commitment methods).

24.3.6 Generator Reserves

Generator Reserve blocks must be connected to a bus and need either one PV or one slack generator and Supply block connected to the same bus. Figure 24.9 illustrates the Reserve block usage. Observe that Generator Reserve data only have effects when used with the PSAT OPF routine.

24.3.7 Non-conventional Loads

Non-conventional load blocks are those described in Chapter 16, i.e. Voltage Dependent Load, ZIP Load, Frequency Dependent Load, Exponential Recovery Load, Thermostatically Controlled Load, Jimma's Load, and Mixed Load. The first two ones can be used as standard blocks when the “Initialize after power flow” parameter is set to 0. However, in general, all non-conventional loads need a PQ load at

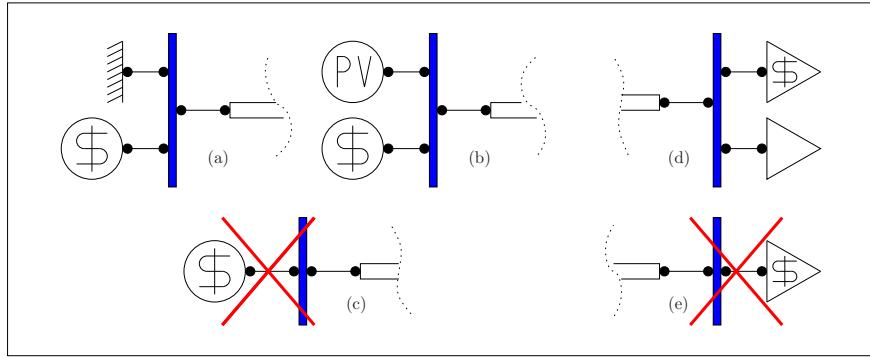


Figure 24.7: *Supply* and *Demand* block usage. (a) and (b) Correct usage of Supply blocks; (c) Incorrect usage of Supply blocks; (d) Correct usage of Demand blocks; (e) Incorrect usage of Demand blocks.

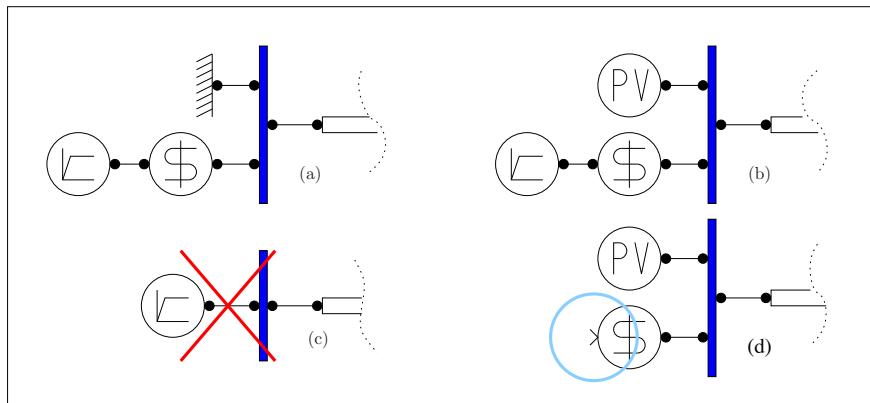


Figure 24.8: *Generator Ramp* block usage. (a) and (b) Correct usage of ramp blocks; (c) Incorrect usage of ramp blocks; (d) Not recommended usage of Supply blocks.

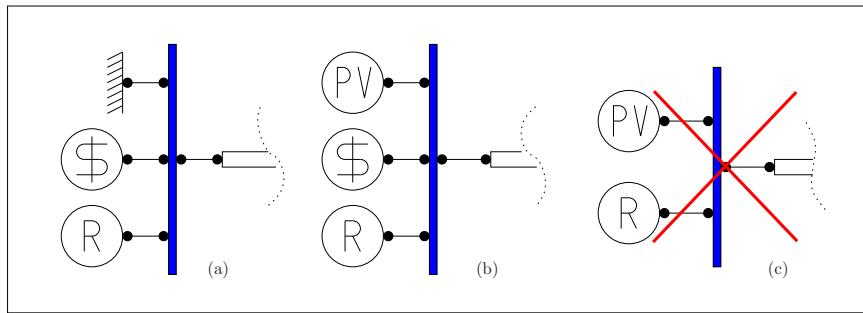


Figure 24.9: *Generator Reserve* block usage. (a) and (b) Correct usage of Reserve blocks; (c) Incorrect usage of Reserve blocks.

the same bus. Multiple non-conventional loads can be connected at the same bus. See Section 16.9 for a few remarks on the usage of non-conventional loads. Figure 24.10 illustrates non-conventional loads usage within SIMULINK models.

24.3.8 Synchronous Machines

Synchronous machine blocks must be connected to a bus and need either one PV or one slack generator connected to the same bus. If no PV or slack generator is present, synchronous machine state variables will not be properly initialized. Synchronous machine block mask allows having input ports. The input ports are needed only when using regulators; to leave unused input ports in Synchronous Machine blocks is deprecated. When connecting multiple synchronous machine to the same bus, the sum of parameters “Percentage of active and reactive power at bus” must be 1. PSAT does not check the consistency of active and reactive fraction used by Synchronous machines. The power used for computing the Synchronous machine power injections are those of PV or slack bus generators. Figure 24.11 illustrates the Synchronous Machine block usage.

24.3.9 Primary Regulators

Primary Regulator blocks such as Automatic Voltage Regulators and Turbine Governors must be connected to a synchronous machine; while Power System Stabilizers, and Over Excitation Systems must be connected to an Automatic Voltage Regulator. AVR block mask allows having input ports. The input ports are needed only when using PSSs, OXLs, or Secondary Voltage Regulator blocks. To leave unused input ports in AVR blocks is deprecated. Only one kind of regulator is allowed for each machine. Figure 24.12 illustrates the usage of the primary regulator blocks.

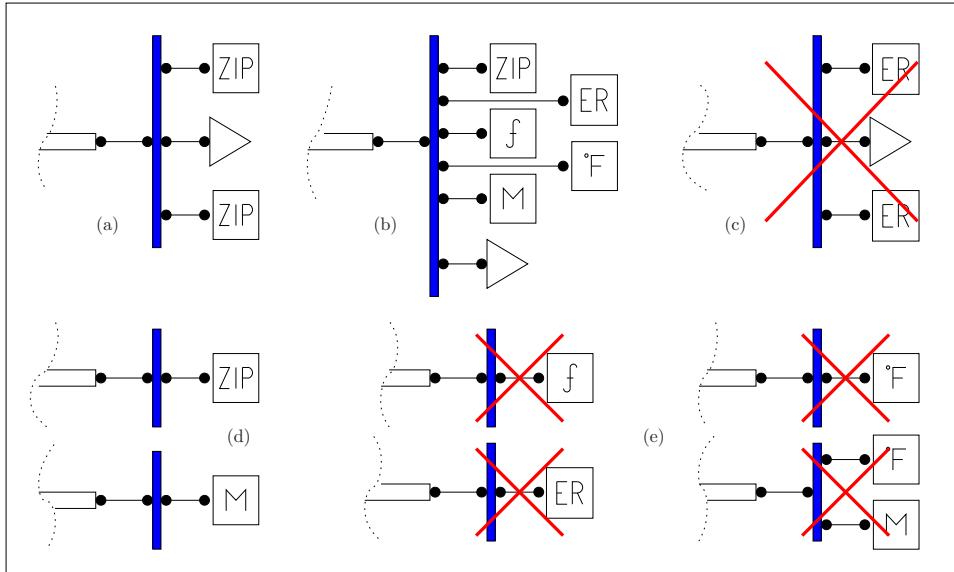


Figure 24.10: *Non-conventional Load* block usage. (a) and (b) Correct usage of non-conventional load blocks; (c) Incorrect usage of exponential recovery load. (d) Correct usage of voltage dependent and ZIP load blocks when the “Initialize after power flow” parameter is set to 0. (e) Incorrect usage of non-conventional load blocks.

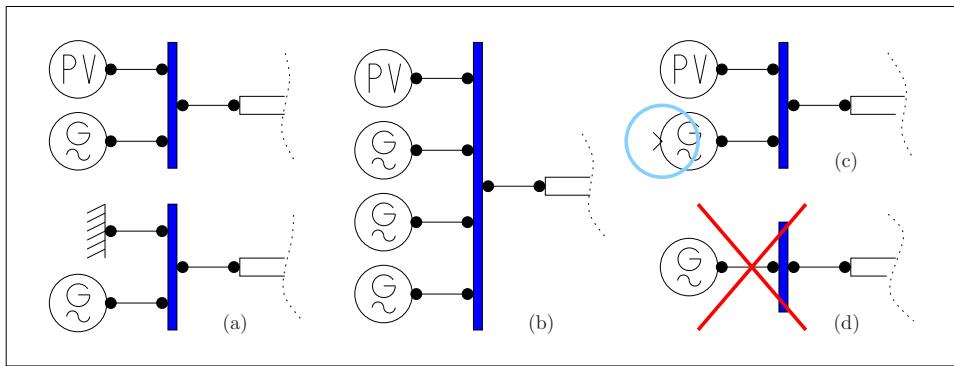


Figure 24.11: *Synchronous Machine* block usage. (a) and (b) Correct usage of synchronous machine blocks; (c) Not recommended usage of synchronous machine blocks. (d) Incorrect usage of synchronous machine blocks.

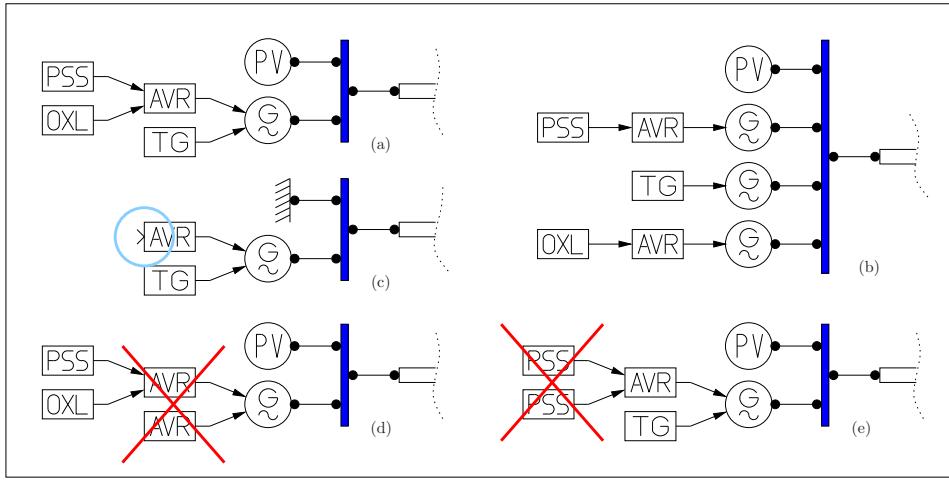


Figure 24.12: *Primary Regulator* block usage. (a) and (b) Correct usage of regulator blocks; (c) Not recommended usage of automatic voltage regulator blocks; (d) and (e) Incorrect usage of regulator blocks.

24.3.10 Secondary Voltage Regulation

Secondary Voltage Regulation (SVR) blocks are the Central Area Controller (CAC) and the Cluster Controller (CC) blocks. CAC or CC blocks must be used together. For each SVR system, there can be **only one** CAC block, while there is no limit to the number of CC blocks for each SVR system. The CAC input port has to be connected to a bus (pilot bus) where the voltage is controlled. The CAC output ports must be connected to CC blocks, and can be in any number. It is not recommended to leave unused output ports in CAC blocks. CC blocks can be connected directly to AVR blocks or to SVC blocks. In the latter case the *Link* block is needed to add the CC control channel to the SVC. Any number of SVCs or AVRs can be included in a SVR system. Furthermore, any model or control type of SVCs and AVRs is allowed. Figure 24.13 illustrates the usage of CAC, CC and Link blocks. Any other usage of these blocks is not allowed and will lead to unpredictable results or to error messages.

24.3.11 Under Load Tap Changers

Under Load Tap Changer (ULTC) blocks can be connected to two or three buses depending on the selected control type. Secondary voltage and reactive power controls (types 1 and 2) need only two buses, as the controlled bus is the secondary winding of the transformer. Remote voltage control (type 3) requires a connection to a third bus. When control type three is selected the shape of the ULTC block changes in order to allow a second input port. Figure 24.14 illustrates the usage of ULTC blocks. The ULTC is a standard block when using control types 1 and 2. In

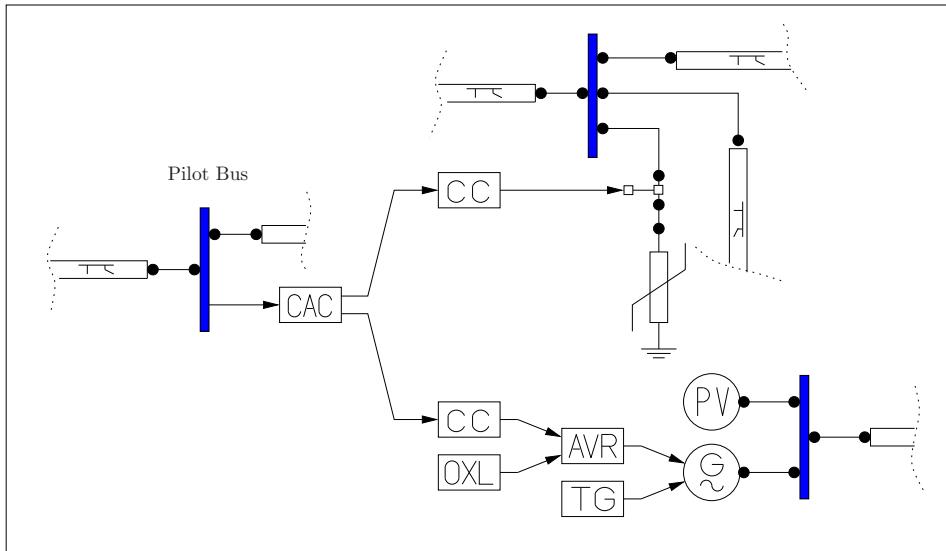


Figure 24.13: *Secondary Voltage Regulation* block usage.

the case of control type 3, it is a nonstandard block since the remote bus provides a signal, not a topological connection.

24.3.12 SVCs & STATCOMs

SVC & *STATCOM* blocks must be connected to a bus and need one PV generator block connected to the same bus. Note that slack generator blocks are not allowed in this case. If no PV generator is present, SVC or STATCOM state variables are not properly initialized and a warning message is displayed. Figure 24.15 illustrates the usage of SVC and STATCOM blocks.

24.3.13 Solid Oxide Fuel Cells

Solid Oxide Fuel Cell blocks must be connected to a bus and need one PV generator block connected to the same bus. Note that slack generator blocks are not allowed in this case. If no PV generator is present, fuel cell state variables are not properly initialized and a warning message is displayed. Figure 24.16 illustrates the fuel cell block usage.

24.3.14 Dynamic Shafts

Dynamic Shaft blocks must be connected to a synchronous machine. Observe that dynamic shafts blocks do not accept any input port, since the model implemented so far does not allow including a turbine governor when using a dynamic shaft.

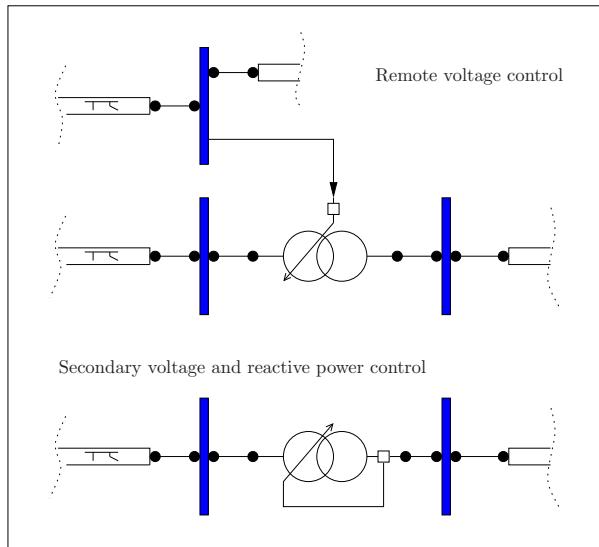


Figure 24.14: *Under Load Tap Changer* block usage.

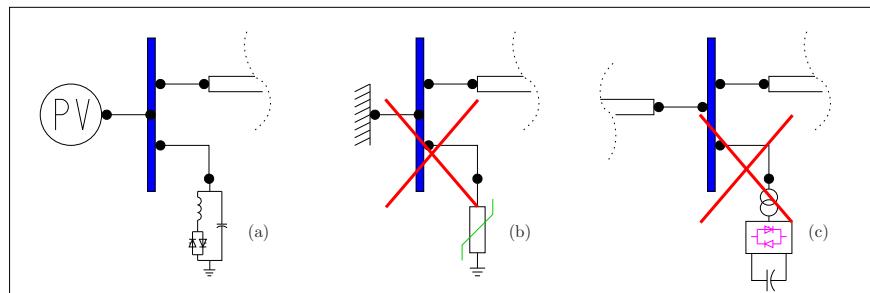


Figure 24.15: *SVC* block usage. (a) Correct usage of SVC blocks; (b) and (c) Incorrect usage of SVC blocks.

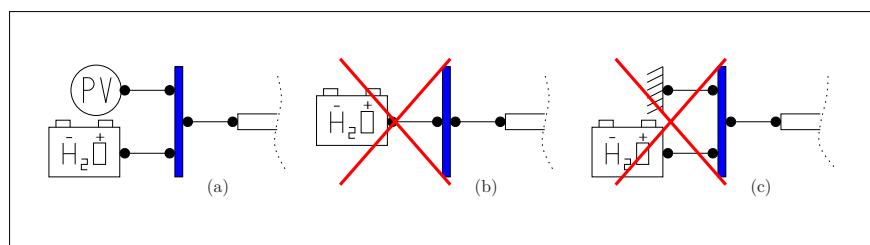


Figure 24.16: *Solid Oxide Fuel Cell* block usage. (a) Correct usage of Fuel Cell blocks; (b) and (c) Incorrect usage of Fuel Cell blocks.

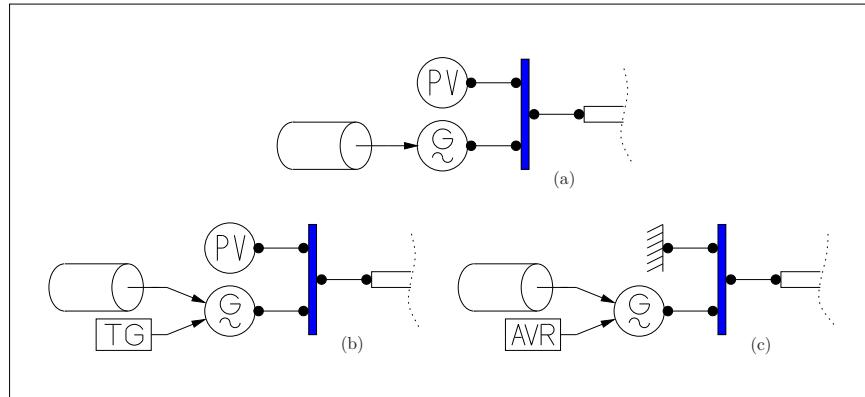


Figure 24.17: *Dynamic Shaft* block usage. (a-c) Correct usages of *Dynamic Shaft* blocks.

Thus, connecting a turbine governor and a dynamic shaft to the same synchronous machine does not give the expected results and it is not allowed. Furthermore, note that the interactions between AVRs and dynamic shaft when connected to the same generators have not been tested so far. Figure 24.17 illustrates the dynamic shaft block usage.

Chapter 25

Block Masks

As already mentioned in Chapter 23, the PSAT-SIMULINK library is not strictly correlated to PSAT internal functions or structures. As a matter of fact, one can write a MATLAB script file for defining PSAT data and never use the SIMULINK interface. However, it is often simpler and more user-friendly drawing a network than dealing with data matrices.

This chapter describes how masks are associated with blocks of the PSAT-SIMULINK library and how to create a new mask for a custom block.

25.1 Blocks vs. Global Structures

PSAT blocks provided within the SIMULINK library are hollow subsystems with a meaningful icon and with a mask which allows setting data. When using SIMULINK, one does not have to care about device indexing and can use the default values which comes with the masks.

The fact that SIMULINK blocks are independent from PSAT structures have pros and cons, as follows:

Pros

1. There can be more than one block associated to the same PSAT global structure, which can be useful to draw nicer networks (see Fig. 25.1).
2. Given a filter able to translate data in a format readable by PSAT, any other CAD tool could be used for drawing PSAT models.

Cons

1. SIMULINK models cannot be directly used as data files.
2. Values contained in the blocks must be interpreted and translated into PSAT global structures (this can be a lengthy process for big networks).

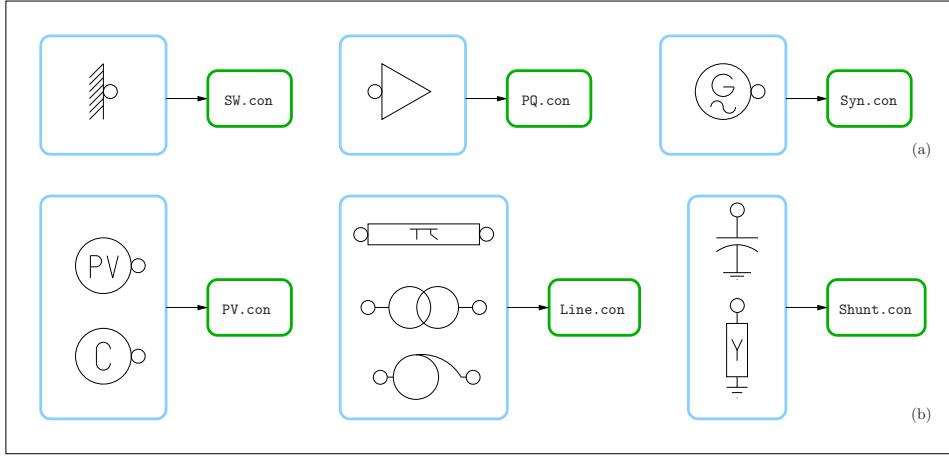


Figure 25.1: Correspondence between SIMULINK blocks and PSAT global structures. (a) examples of structures associated with only one SIMULINK block; (b) examples of structures associated with more than one SIMULINK block.

25.2 Editing Block Masks

SIMULINK allows defining new blocks, which are typically masked subsystems. This feature is extensively used in the PSAT-SIMULINK library, in order to define custom blocks.

Generally speaking, editing a block mask means setting up an initialization, an icon, and a documentation for the block. Only the initialization is strictly needed to set up a working block. However block icons are used in the PSAT library to emulate a power system diagram and a brief documentation helps reminding the component associated with the block.

Finally, the mask provides a field called *Mask Type*, which is used for defining the link between the PSAT blocks and PSAT global structures. This property tells the PSAT filter (i.e. the function `fm_sim`) which structure is associated with the block. It is always possible to know which structure is associated with the current SIMULINK block by simply opening its mask: the name just below the GUI window title and above the block documentation is the PSAT structure (see Fig. 25.2). Furthermore, one can modify at any time the *Mask Type* property, by unlinking the block and editing its mask. Of course this is not generally needed unless one wants to create a new block. At this aim refer to Section 25.4 in this chapter.

25.2.1 Mask Initialization

Each SIMULINK block mask must be initialized, i.e. it needs a set of parameters and functions which are launched a first time when the block is created and then each time any property of the block is changed. An example of block initialization

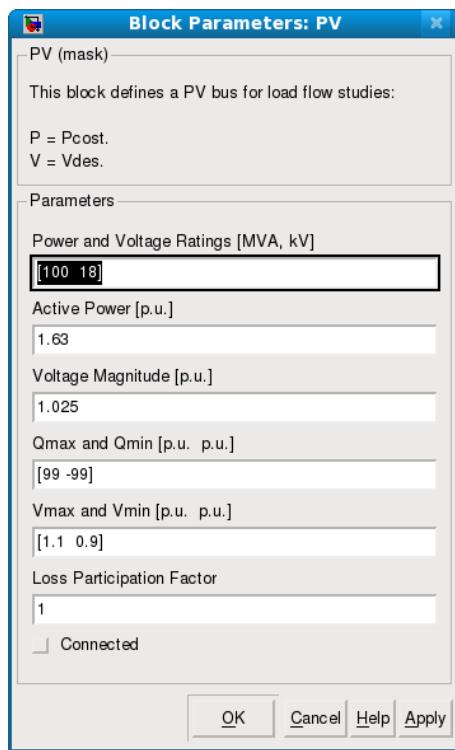


Figure 25.2: Mask GUI of a PSAT-SIMULINK block.

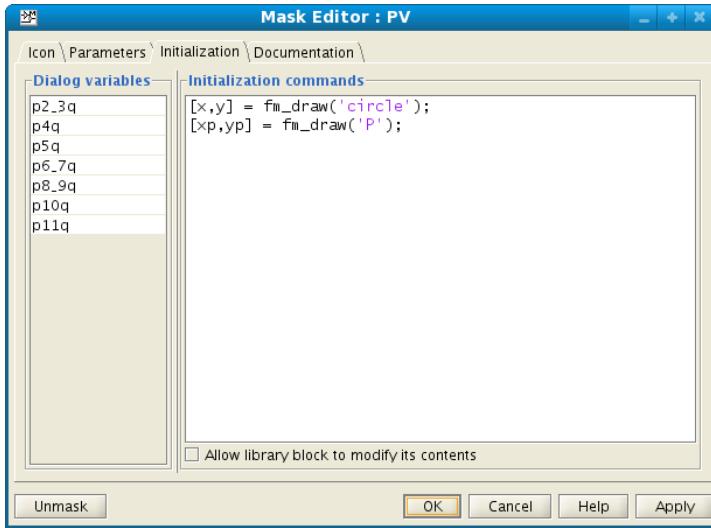


Figure 25.3: Mask initialization GUI for a PSAT-SIMULINK block.

is depicted in Fig. 25.3.

Each PSAT block uses the initialization GUI to define block parameters. Parameter names have a special syntax, which is fully described in Section 25.3.

25.2.2 Mask Icon

Mask icons are defined in the edit mask GUI as well. An example of block icon is depicted in Fig. 25.4. The icon is drawn by means of `plot` statements, which in some cases may be a lengthy process (e.g. when drawing circles). At this aim a few plotting utilities are provided in the function `fm_draw`, which is called at the initialization step (see Fig. 25.3).

Observe that more complicated block features, such as a variable icon which depends on parameter values cannot be obtained by means of the simple mask editing. In these cases a mask callback function has to be defined. The function `fm_block` takes care of these special “auto-adaptive” PSAT blocks.

Finally, blocks which have a variable number of input/output ports are handled at the initialization step by means of the `fm_inout` function.

25.2.3 Mask Documentation

Each PSAT block comes with a brief documentation. An example of block documentation is depicted in Fig. 25.5.

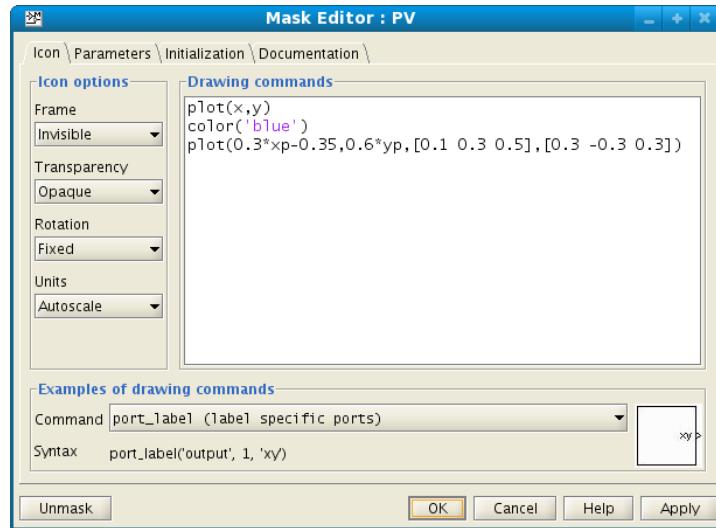


Figure 25.4: Mask icon GUI of a PSAT-SIMULINK block.

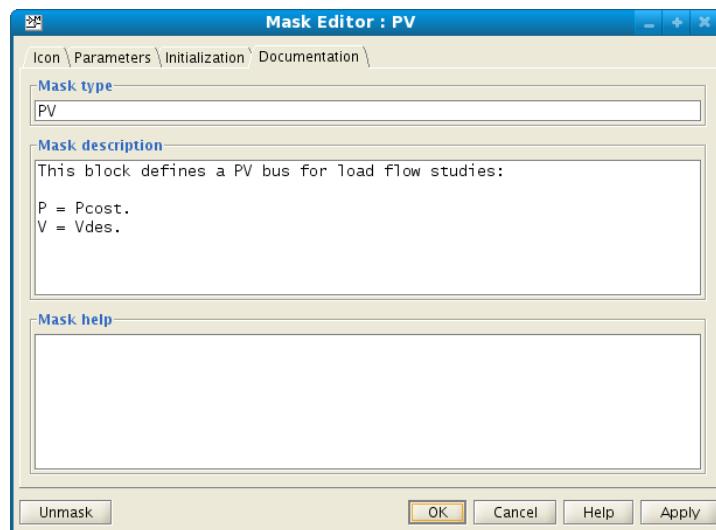


Figure 25.5: Mask documentation GUI of a PSAT-SIMULINK block.

Table 25.1: Mask parameter symbols

Pseudo Symbol	Meaning	Matlab Symbol
p	open array	[
q	close array]
_ ²	comma	,
x	list	:

Table 25.2: Example of well formed mask variable names

Variable name	MATLAB expression	Sample mask value
p3q	[3]	0.05
p3_4q	[3,4]	[0.05 1.00]
p3x5q	[3:5]	[0.05 1.00 1.25]
p3x5_7q	[3:5,7]	[0.05 1.00 1.25 0.333]

25.3 Syntax of Mask Parameter Names

Parameter names within block masks follow few simple rules, which make possible defining blocks independent from PSAT structures. When the parameters are loaded from the SIMULINK model, the function `fm_sim` takes care of assigning the parameters values to the correct structure and to fill up the correct columns and rows within the data matrix of the structure. The structure is the *Mask Type* property of the block, while the row number depends on the number of blocks of the same kind included in the SIMULINK model. Thus, the parameter names have just to specify in which columns data have to be stored. The trick consists of using a special syntax for defining arrays which can be easily converted into MATLAB expressions (using for example, regular expressions) and at the same time are well formed MATLAB variables. The symbols are depicted in Table 25.1 whereas a few examples of well formed mask variable names are depicted in Table 25.2.

There are also some keywords, which are associated with constant values, as depicted in Table 25.3. The keywords `in` and `out` when used as parameter names produce no effects, and are typically used with blocks which have a variable number of ports, such as buses and synchronous machines. Observe that keywords cannot be used within parameter names formed using the symbols of Table 25.1.

Mask values associated with mask parameter names must be consistent, i.e. have to be arrays of the dimension defined by the parameter names. A few examples are reported in the third column of Table 25.2

Observe that PSAT does not require that mask arrays are enclosed in brackets; however, to avoid warning messages generated by SIMULINK, it is recommended to use a correct MATLAB syntax for mask values.

²That is underscore, not dash.

Table 25.3: Mask parameter constants

Keyword	Value	Keyword	Value
on	1	saturday	6
off	0	sunday	7
omega	1	winter_week_day	1
power	2	winter_week_end	2
voltage	3	summer_week_day	3
monday	1	summer_week_end	4
tuesday	2	spring_fall_week_day	5
wednesday	3	spring_fall_week_end	6
thursday	4	in	none
friday	5	out	none

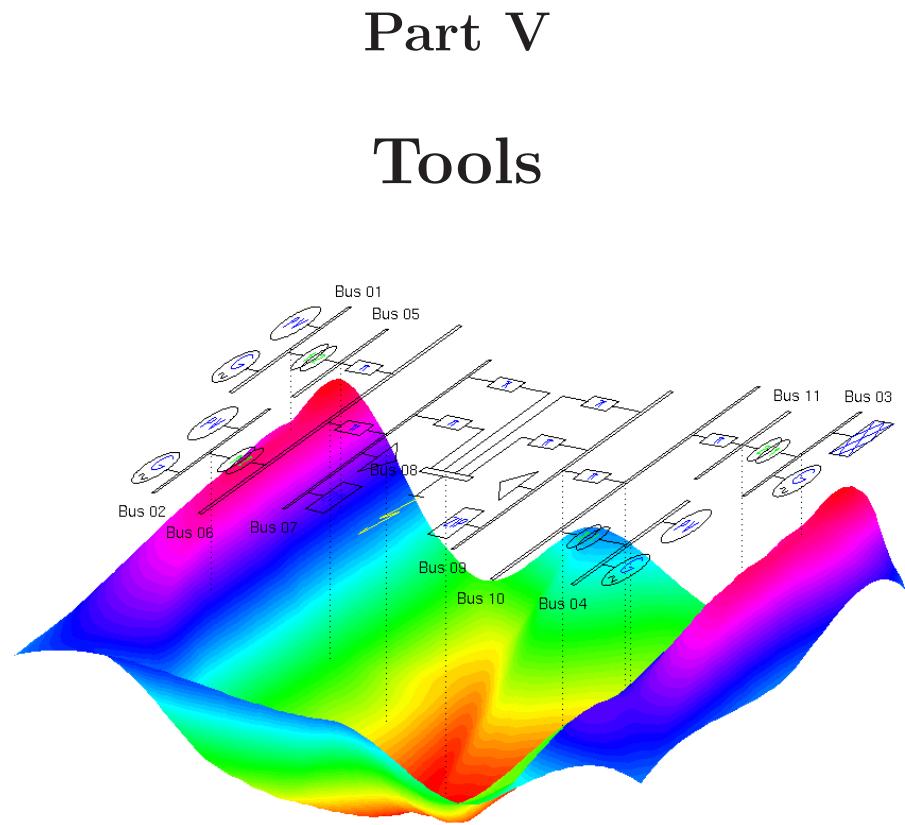
25.4 Remarks on Creating Custom Blocks

The easiest way to create a new PSAT block is to modify an existing one, as follows:

1. open and unlock the PSAT SIMULINK library;
2. copy the desired block;
3. unlink the copied block;
4. edit the block mask.

A few remarks on creating a new PSAT blocks follow:

1. PSAT blocks are typically hollow *Physical Modeling Components*. To be recognized as a PSAT block, the *PhysicalDomain* property must be set to *psat-domain*. and the block Tag must be set to *PSATblock*.
2. Some blocks also have control signals. In this case, the top level block is an hollow subsystems containing input and output ports.
3. The *Mask Type* property has always to be defined and has to be an existing PSAT structure.
4. Mask variables must follow the conventional syntax defined in the previous Section 25.3.
5. Since mask variables are going to fill up the data field of a PSAT structure, all columns (but the optional ones) of the resulting matrix of data have to be filled up.



Chapter 26

Data Format Conversion

PSAT is able to recognize and convert a variety of data formats commonly in use in power system research.¹

PSAT data files containing *only* static power flow data can be converted into the IEEE common data format and into the WSCC and EPRI ETMSP format.² PSAT data can be also converted into the ODM format that is a novel approach for power system data exchange [143, 144, 142].

Filters are written mostly in Perl language. The only filters that are written in MATLAB are those that convert MATLAB scripts or functions (e.g. PST and MATPOWER formats).

The conversion to and from PSAT may not be complete and may lead to unexpected results. In some cases, changes in the default PSAT settings are needed to reproduce results obtained by other power system software packages.

The conversion can be done from the command line or through the GUI for data format conversion, which can be launched using the *Tools/Data Format Conversion* menu in the main window. Figure 26.1 depicts the this GUI. Observe that SIMULINK models can be previewed in the GUI before being loaded.

The following filters have been implemented so far:³

`cepel2psat`: conversion from CEPEL data format;

`chapman2psat`: conversion from Chapman's data format [31];

`cyme2psat`: conversion from CYME power flow data format (CYMFLOW);

`digsilent2psat`: conversion from DIgSILENT data exchange format;

`epri2psat`: conversion from WSCC and EPRI's ETMSP data format;

¹Some of these filters have been kindly contributed by Juan Carlos Morataya R., Planificación y Control, EEGSA, Iberdrola, Guatemala. E-mail: JMOrataya@eegsa.net.

²Details on the IEEE Common Data Format can be find in [137]. Furthermore, a description of the IEEE CDF and on the EPRI ETMSP formats can be found at www.power.uwaterloo.ca/

³All filters can be found in the folder `psat/filters`.

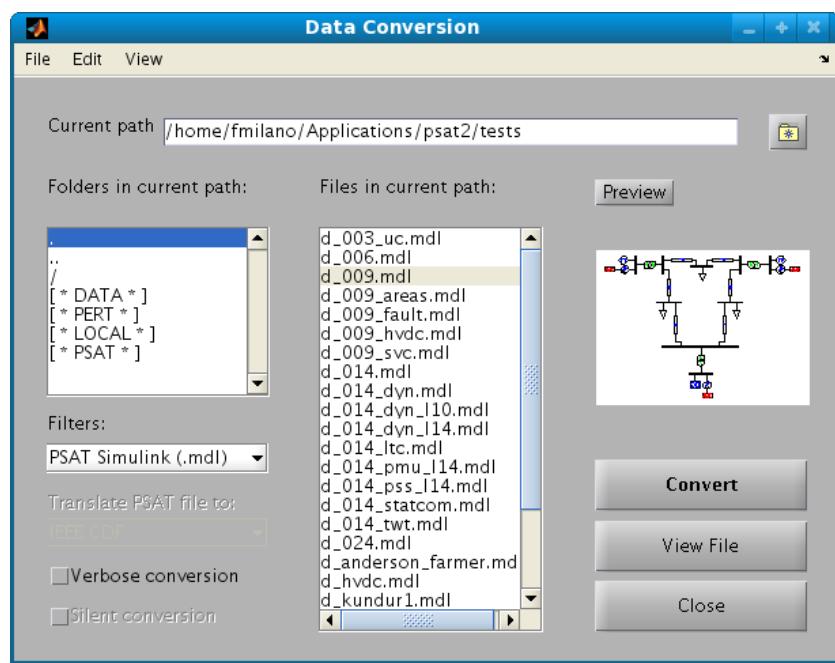


Figure 26.1: GUI for data format conversion.

eurostag2psat: conversion from Eurostag data format;
flowdemo2psat: conversion from FlowDemo.net data format;
ge2psat: conversion from General Electric data format;
ieee2psat: conversion from IEEE common data format;
inptc12psat: conversion from CESI INPTC1 data format;
ipss2psat: conversion from InterPSS XML data format;
ipssdat2psat: conversion from InterPSS plain data format;
matpower2psat.m: conversion from MATPOWER data format;
neplan2psat: conversion from NEPLAN data format;⁴
odm2psat: conversion from ODM data format;
pcflo2psat: conversion from PCFLO data format;
psap2psat: conversion from PSAP data format;⁵
psat2ieee.m: conversion to IEEE common data format;
psat2epri.m: conversion to EPRI/WSCC data format;
psat2odm.m: conversion to ODM format;
psse2psat: conversion from PSS/E data format (up to version 29);⁶
pst2psat.m: conversion from PST data format;
pwrworld2psat: conversion from POWERWORLD data format;
reds2psat: conversion from REDS data format;
simpow2psat: conversion from SIMPOW data format;
sim2psat.m: conversion from PSAT-SIMULINK models;
th2psat: conversion from Tsing Hua University data format;
ucte2psat: conversion from UCTE data format;
vst2psat: conversion from VST data format;
webflow2psat: conversion from WebFlow data format.

⁴This filter supports both comma and tab separated data formats.

⁵A description of the PSAP data format can be found at

www.ee.washington.edu/research/pstca/

⁶The filter should support PSS/E data format from version 26 to 30. A description of an old version of the PSS/E data format is available at www.ee.washington.edu/research/pstca/

Perl-based filters can be used from a command shell, as any UNIX application. The general syntax for perl-based filters is as follows:

```
$ <filter_name> [-v] [-h] [-a add_file] input_file [output_file]
```

where \$ is the shell prompt. The only mandatory argument is `input_file`. If no `output_file` is specified, the output file name will be automatically generated by the filter. Options are as follows:

`-v` : verbose conversion. For some filters, additional information is printed out during conversion.

`-h` : print a brief help and exit.

`-a` : define additional file needed for conversion. This option is only available for `neplan2psat` and `inptc12psat` filters, as follows:

`neplan2psat` : the additional file is a `.edt`. If the `-a` option is not used, the filter will assume that the `.edt` file has the same name as the `.ndt` file.

`inptc12psat` : the additional file is a COLAS ADD, typically with extension `.dat`. If the `-a` option is not used, the filter will assume there is no COLAS ADD file.

Chapter 27

Utilities

This chapter describes the GUIs and the properties of command history, 3D temperature map GUI, advanced settings, sparse matrix visualization, theme and text viewer settings.

27.1 Command History

The command history of all PSAT operations is contained in the structure `History`, which can be displayed and saved by means of the GUI depicted in Fig. 27.1. The GUI can be launched using the *Options/History* menu in the main window.

27.2 3D Temperature Map

The GUI for drawing 3D map can be used whenever the data is loaded from a SIMULINK model. After solving the power flow, the GUI is launched from the menu **View/Network Visualization** of the main PSAT window. The GUI allows drawing 3D temperature maps of relevant quantities of the system, namely bus voltages, line flows, generator rotor angles or speeds, LMPs and NCPs. The latter two are available only after solving the OPF analysis. The user can also interactively rotate the 3D plot, chose among a variety of color maps and set the transparency level. If the GUI is open during CPF analysis or time domain integration, the map is updated during the simulation. At the end of the simulation, the entire animation can be reproduced as a movie.¹ 3D plots can be saved as single frames or as movies.

27.3 Advanced Settings

General settings are adequate in most cases. Advanced settings are intended for those relatively rare circumstances in which the user needs to force some uncon-

¹In some cases, reproducing large simulations has no effect. This is likely a MATLAB issue.

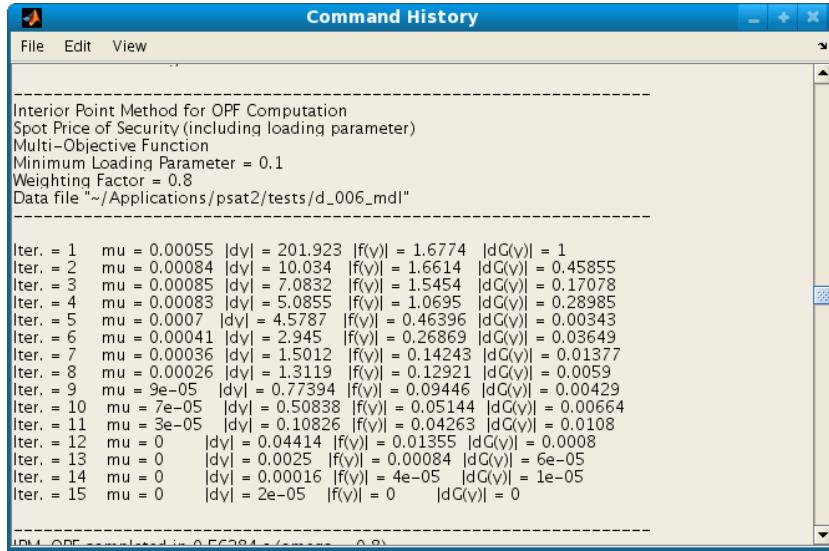


Figure 27.1: GUI for the command history.

ventional behavior of the program. These settings are:

1. *Reset pre-fault bus angles after fault clearing* (`Settings.resetangles`): Useful if there are convergence problems (e.g. voltage recovery) after fault clearing during time domain simulations. See also Section 14.1.
2. *Switch back to the NR method for small mismatches* (`Settings.switch2nr`): Only applies to robust power flow solvers. It can be used to reduce the number of iterations.
3. *Apply current options without asking for confirmation* (`Settings.donotask`): Force the usage of the actual settings even in case these could lead to inconsistencies.
4. *Skip updating of snapshots after power flow analysis* (`Settings.locksnap`): This option can help saving memory but some part of the program could not work properly.
5. *Enable Jacobian trajectory tracking* (`Settings.vs`): Enforce the storage of the determinant and the eigenvalues of the system state matrix \mathbf{A}_S during time domain simulation. It can dramatically slow down the performance of the program. Enforcing this option is deprecated.
6. *Number of PF iterations before switching PV to PQ* (`Settings.pv2pqniter`): Wait n iterations of the power flow solver before switching PV generators to PQ if a reactive power limit is reached. $n = 0$ is the default value. n should be less than the total iteration number of the power flow solver.

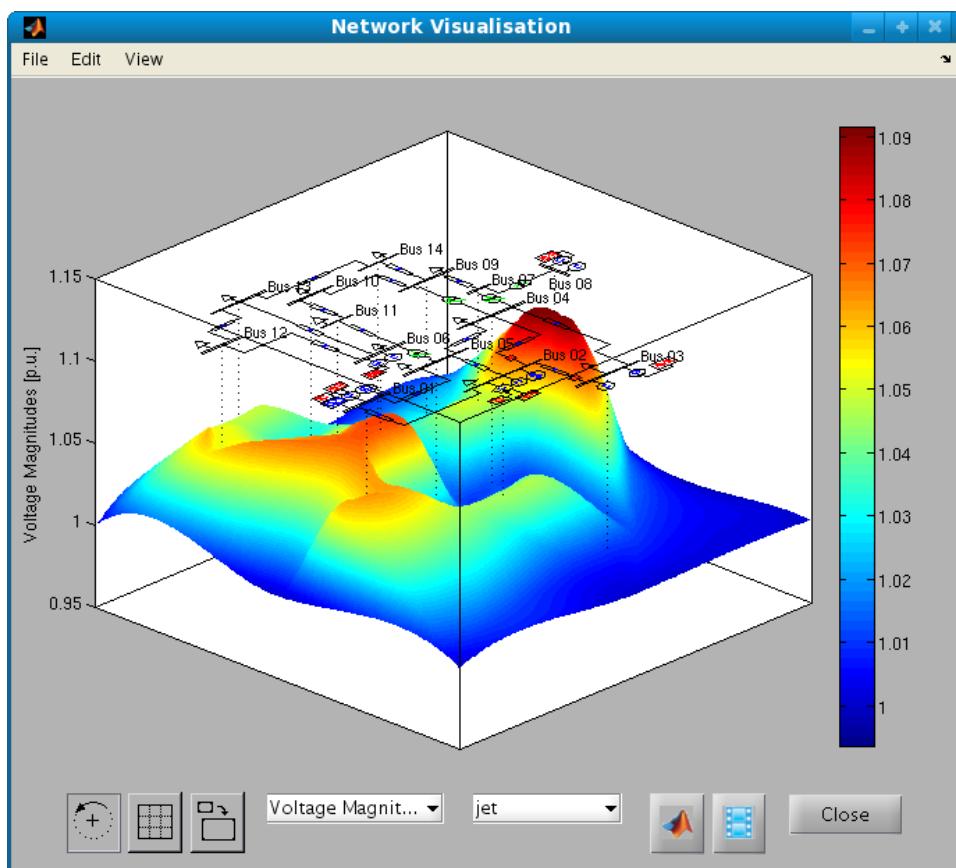


Figure 27.2: GUI for 3D map visualization.

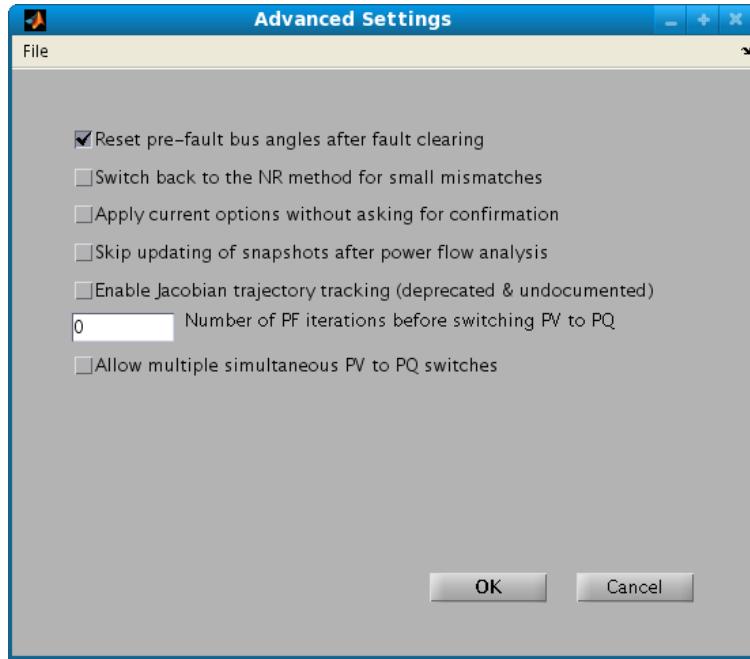


Figure 27.3: GUI for advanced settings.

7. *Allow multiple simultaneous PV to PQ switching* (`Settings.multipvswitch`): Allows more than one PV generator can be switched to PQ if a reactive power is reached. By default only one PV generator is switched per each iteration (this may lead to several iterations but is generally a robust method).

The GUI for advanced settings is depicted in Fig. 27.1. The GUI can be launched using the *Edit* menu of the main window.

27.4 Sparse Matrix Visualization

Figure 27.4 depicts the GUI for sparse matrix visualization that can be launched from the *View/ Sparse Matrix Visualization* menu in the main window.²

27.5 Themes

The graphical appearance of PSAT GUIs can be changed using the theme browser GUI, available in the *Option/Themes* in the main window. Figure 27.5 depicts this GUI displaying a preview of a theme provided with the toolbox. To add a new

²The figure illustrates the complete power flow Jacobian matrix of a 1254-bus test system.

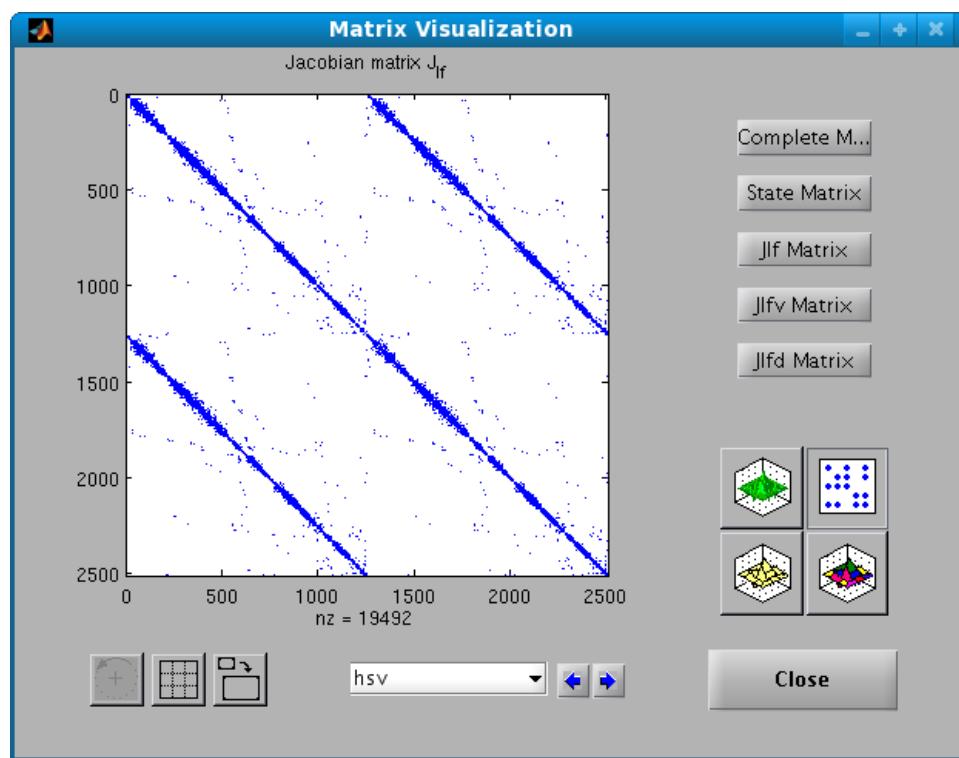


Figure 27.4: GUI for sparse matrix visualization.

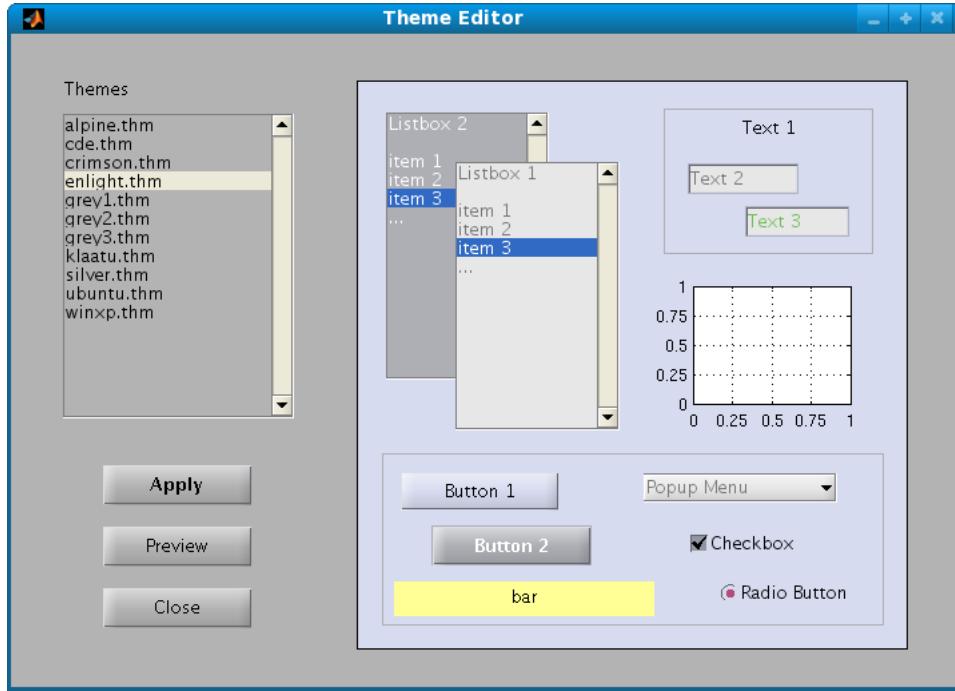


Figure 27.5: GUI for PSAT theme selection.

themes, just open and modify one of the sample files contained in the sub-folder **themes** within the main PSAT folder.

27.6 Text Viewer

Figure 27.6 depicts the GUI for selecting the text viewer used by PSAT for displaying reports generated by the routines. The programs are grouped based on the operating system, i.e. Unix (Solaris), Linux and Windows. Power flow results can be saved in three different formats, i.e. Microsoft Excel³, plain text (ASCII file) and L^AT_EX formatted plain text. This GUI can be launched from the *Options/Text Viewer* menu in the main window and from several other setting windows.

27.7 Benchmarks

The function **benchmark** provides an easy way to check whether the current PSAT version is working fine. If everything goes well, the function gives the following output:

³ActiveX® is used for exporting results to Microsoft Excel.



Figure 27.6: GUI for text viewer selection.

```
>> benchmark

    < P S A T >
Copyright (C) 2002-2008 Federico Milano
Version 2.1.3
July 1, 2008

PSAT comes with ABSOLUTELY NO WARRANTY; type 'gnuwarranty'
for details. This is free software, and you are welcome to
redistribute it under certain conditions; type 'gnulicense'
for details.

Host:          Matlab 7.6.0.324 (R2008a)
Session:       04-Nov-2008 17:09:57
Usage:         Command Line
Path:          /Users/fmilano/Applications/psat2

Test 1 - Power flow analysis.
* * * Newton-Rapshon, single slack bus: OK
* * * Newton-Rapshon, distributed slack bus: OK
* * * Fast decoupled power flow: OK
* * * Static report: OK

Test 2 - Continuation power flow analysis.
```

```
* * * SNB bifurcation: OK
* * * LIB bifurcation: OK
* * * HB bifurcation: OK

Test 3 - Optimal power flow analysis.
* * * Social benefit: OK
* * * Maximum loading condition: OK

Test 4 - Small signal stability analysis.
* * * Stable case: OK
* * * Unstable case: OK

Test 5 - Time domain simulation.
* * * Fault analysis 1: OK
* * * Fault analysis 2: OK

Test 6 - Interfaces.
* * * GAMS interface: OK
* * * UWPFLOW interface: OK
Testing completed.
```

The benchmark tests also produce five plots that should look like the ones that are displayed in Figure 27.7.

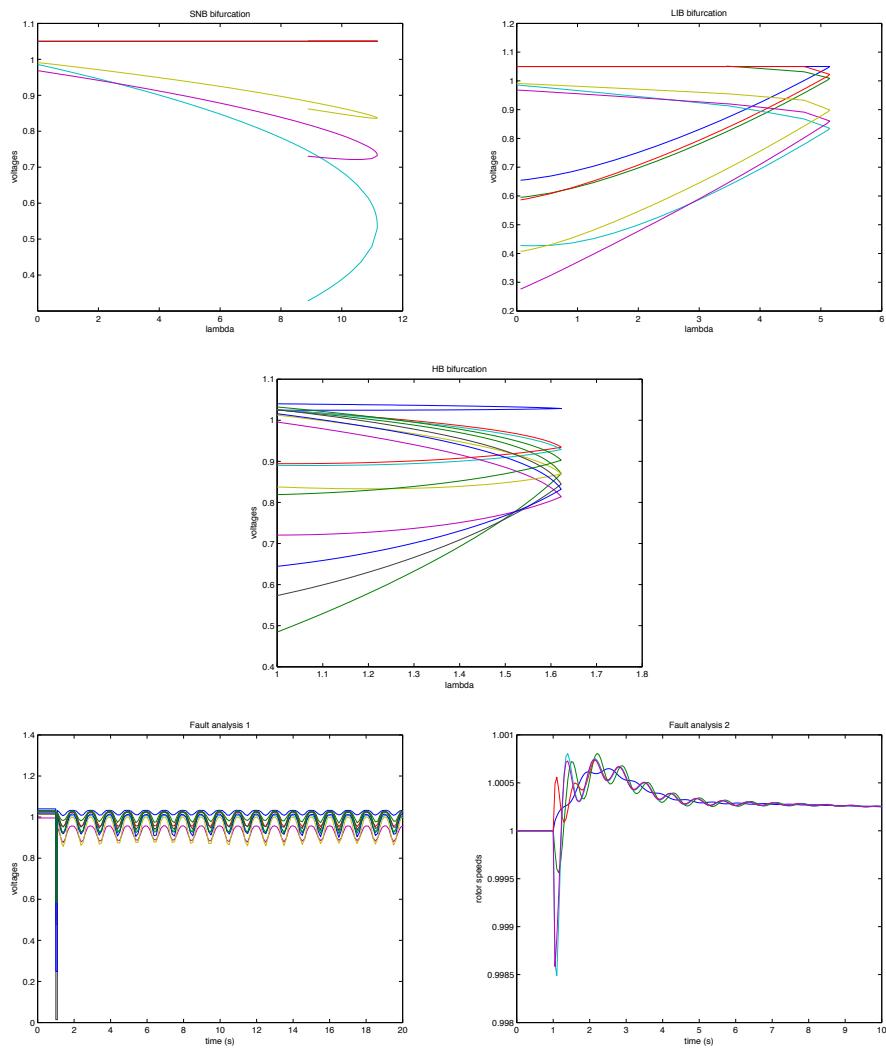


Figure 27.7: Output plots of the benchmark tests.

Chapter 28

Command Line Usage

A set of functions and script files for command line usage of PSAT have been added since PSAT version 1.3.0. These functions get rid of PSAT GUIs, which could be undesired when running PSAT on a remote server/host or when launching PSAT from within user defined routines. The command line usage of PSAT also allows speeding up operations.

28.1 Basics

The command line version of PSAT is launched as follows:

```
>> initpsat
```

This will initialize PSAT and display on the MATLAB workspace:

```
< P S A T >
Copyright (C) 2002-2008 Federico Milano
Version 2.1.3
July 1, 2008
```

```
PSAT comes with ABSOLUTELY NO WARRANTY; type 'gnuwarranty'
for details. This is free software, and you are welcome to
redistribute it under certain conditions; type 'gnulicense'
for details.
```

```
Host: Matlab 7.6.0.324 (R2008a)
Session: 03-Nov-2008 18:55:52
Usage: Command Line
Path: /Users/fmilano/Applications/psat2
```

Existing workspace variables are not cleared during the initialization, as it happens when launching the PSAT GUI. Clearing the workspace could not be the desired behavior as the command line version of PSAT can be used from within user defined

routines. However, observe that all user variables which have same names as a PSAT global variables will be overwritten. Refer to Chapter A for the complete list of PSAT global variables.

The scope of PSAT global variables will be the scope of the current workspace from where `initpsat` is called. If `initpsat` is called from within a user defined function, the scope will be the function workspace and the PSAT global variables will *not* be available in the MATLAB workspace. To set PSAT global variables in the common MATLAB workspace, `initpsat` must be launched from the MATLAB command line or from within a script file.¹

Initializing the PSAT variables is required only once for each workspace.

Following steps are setting up the data file and launching a PSAT routine. These operations can be done sequentially or at the same time by means of the function `runpsat`, as follows:

```
>> runpsat(datafile, 'data')
>> runpsat(routine)
```

or

```
>> runpsat(datafile, routine)
```

where *datafile* is a string containing the data file name, and *routine* is a string containing the conventional name of the routine to be executed. The data file can be both a PSAT script file or a PSAT SIMULINK model. In the latter case the extension `.mdl` is mandatory.

The difference between the two methods is that when calling only the routine the data file name will not be overwritten. The first method can be used if the data file under study does not change, while the user wants to perform several different analysis, as follows:

```
>> runpsat(datafile, 'data')
>> runpsat(routine1)
>> runpsat(routine2)
>> runpsat(routine3)
```

The second method can be used if there are several data files under study:

```
>> runpsat(datafile1, routine)
>> runpsat(datafile2, routine)
>> runpsat(datafile3, routine)
```

In the previous commands it is assumed that the data file is in the current directory (i.e. the one which is returned by the function `pwd`). To force PSAT to use a directory other than the current one, commands changes as follows:

```
>> runpsat(datafile, datapath, 'data')
>> runpsat(routine)
```

¹The latter should not have been launched from within a function.

Table 28.1: Routine Conventional Names for Command Line Usage.

String	Associated routine
pf	power flow analysis
cpf	continuation power flow analysis
snb	direct method for saddle-node bifurcations
lib	direct method for limit-induced bifurcations
cpfatz	evaluate ATC using CPF analysis
sensatz	evaluate ATC using sensitivity analysis
n1cont	N-1 contingency analysis
opf	optimal power flow analysis
sssa	small signal stability analysis
td	time domain simulation
pmu	PMU placement
gams	OPF analysis through the PSAT-GAMS interface
uw	CPF analysis through the PSAT-UWPFLOW interface

or

```
>> runpsat(datafile, datapath, routine)
```

where *datapath* is the absolute path of the data file.

The perturbation file can be set in a similar way as the data file. At this aim, the following commands are equivalent:

```
>> runpsat(pertfile, 'pert')
>> runpsat(pertfile, pertpath, 'pert')
>> runpsat(datafile, datapath, pertfile, pertpath, routine)
```

Observe that if setting both the data and the perturbation files, it is necessary to specify as well the absolute paths for both files.

The routine names are depicted in Table 28.1. Observe that if **runpsat** is launched with only one argument, say **option**, the following notations are equivalent:

```
>> runpsat('option')
>> runpsat option
```

Other command line options for **runpsat** are depicted in Table 28.2. The syntax for the **opensys** option is the same as the one for **data** and **pert** options.

If the PSAT variables are not needed anymore, the workspace can be cleared using the command:

```
>> closepsat
```

which will clear only PSAT global structures.

Table 28.2: General Options for Command Line Usage.

String	Associated routine
data	set data file
pert	set perturbation file
opensys	open solved case
savesys	save current system
log	write log file of the current session
pfrep	write current power flow solution
eigrep	write eigenvalue report file
pmurep	write PMU placement report file

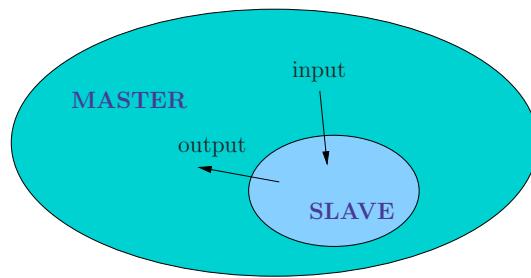


Figure 28.1: Master-slave architecture.

28.2 Advanced Usage

The standard usage of PSAT through GUIs monopolizes the MATLAB environment and makes difficult to include PSAT routine in other MATLAB programs and/or including new features to PSAT. These issues will be briefly commented in this section.

When using PSAT GUIs, PSAT runs as a master program and the user can initialize and launch each internal routine from the main window. Thus each routine is a slave program (see Figure 28.1). Using this architecture, the only way to include a new routine in PSAT is writing a function which interacts with the PSAT GUIs, shares some of the PSAT global structures and properly exchanges information with PSAT. However, users who want to run PSAT routines within their own algorithms generally need to get rid of GUIs. Thus, the best solution would be to use the user defined program as the master and launching PSAT only when needed, as a slave application. In this way the user only needs to know how to pass and get data to and from PSAT.

The latter can be easily solved by using PSAT global structures such as DAE, which mostly contains all variables of the current static solution (power flow, last CPF point, OPF), SSSA which contains the last small signal stability analysis solution, and Varout which contains the time domain simulation output, the continua-

Table 28.3: Structures to be modified to change default behavior.

Routine	Associated structure
Power Flow	Settings
Continuation Power Flow	CPF
SNB direct method	SNB
LIB direct method	LIB
Optimal Power Flow	OPF
Small Signal Stability Analysis	SSSA
Time Domain Simulation	Settings
PMU placement	PMU
PSAT-GAMS interface	GAMS
PSAT-UWPFLOW interface	UWPFLOW

tion curves or the Pareto's set. The structure DAE also contains the current system Jacobian matrices. Refer to Appendix A for details.

Passing data and options to PSAT is quite simple if the default behavior is convenient for the current application. Otherwise, one needs to edit the PSAT global structures and set the desired options. When using the standard version of PSAT, global structures are edited through the GUIs.

Editing global structures from the command line can be a lengthy process, especially if one needs repeating often the same settings. In this case it could be convenient to write a script file where these settings are listed altogether and then launching the script file. Table 28.3 depicts PSAT routines and the associated global structures which define routine options. A full description of these structures is presented in Appendix A.

28.3 Command Line Options

The default behavior of command line usage of PSAT can be adjusted by means of the structure `clpsat`, which contains a few options, as follows:²

`init` command line initialization status. It is 1 if PSAT is running with the standard GUI support, 0 otherwise. The value of this field should not be changed by the user and is initialized when launching PSAT.

`mesg` status of PSAT messages. If the value is 0, no message will be displayed on the MATLAB workspace. Default value is 1. Disabling message display will result in a little bit faster operations.

`refresh` if true (default), forces to repeat power flow before running further analysis independently on the power flow status. This implies that the base case solution is used as the initial solution for all routines.

²In the following the word *true* means the value of the variable is 1 and *false* means 0.

refreshsim if true, forces to reload SIMULINK model before running power flow independently on the SIMULINK model status. Default is false since in the command line usage it is assumed that the user does not want to or cannot use the SIMULINK graphical interface.

readfile if true, forces to read data file before running power flow. If the value is false (default), the data file is not reloaded (unless it has been modified), and slack generator, PV generator and PQ load data are reinitialized using their fields **store**. These data need to be reloaded since they might be modified during PSAT computations.

showopf if true, forces to display OPF result on the standard output. Default is false.

pq2z if true (default), forces to switch PQ loads to constant impedances before running time domain simulations.

viewrep if true, forces to display report files when created. Default is false, i.e. the report file is created silently.

For the sake of completeness, a summary of the fields of the **clpsat** structure is also depicted in Appendix A.

28.4 Example

The following script file gives a simple example of command line usage of PSAT.

```
% initialize PSAT
initpsat

% do not reload data file
clpsat.readfile = 0;

% set data file
runpsat('d_006.mdl','data')

% solve base case power flow
runpsat('pf')
voltages = DAE.y(1+Bus.n:2*Bus.n);

% increase base loading by 50%
for i = 1:10
    PQ.store(:,[4,5]) = (1+i/20)*[0.9, 0.6; 1, 0.7; 0.9, 0.6];
    PV.store(:,4) = (1+i/20)*[0.9; 0.6];
    runpsat('pf')
    voltages = [voltages, DAE.y(1+Bus.n:2*Bus.n)];
end
```

```
% clear PSAT global variables  
closepsat  
  
disp(voltages)
```

Firstly, PSAT is initialized and the `readfile` option is set to false. Then the file `d_006.mdl` is loaded (assuming that the file is in the current directory). Following instructions explain how to solve the base case power flow and a series of power flows with increased loads by means of an embedding algorithm. Finally the PSAT variables are cleared and the bus voltages printed on the workspace, as follows:

```
voltages =  
  
Columns 1 through 6  
  
1.0500    1.0500    1.0500    1.0500    1.0500    1.0500  
1.0500    1.0500    1.0500    1.0500    1.0500    1.0500  
1.0500    1.0500    1.0500    1.0500    1.0500    1.0500  
0.9859    0.9820    0.9781    0.9741    0.9700    0.9660  
0.9685    0.9633    0.9579    0.9525    0.9469    0.9413  
0.9912    0.9876    0.9840    0.9803    0.9765    0.9728  
  
Columns 7 through 11  
  
1.0500    1.0500    1.0500    1.0500    1.0500  
1.0500    1.0500    1.0500    1.0500    1.0500  
1.0500    1.0500    1.0500    1.0500    1.0500  
0.9618    0.9576    0.9533    0.9490    0.9446  
0.9356    0.9298    0.9239    0.9179    0.9118  
0.9689    0.9650    0.9611    0.9571    0.9531
```

Observe the usage of the `store` fields of the PV and PQ components. This allows changing the values of the system loading profile without reloading the data file.

Chapter 29

PSAT on GNU Octave

GNU OCTAVE¹ is a high-level language, primarily intended for numerical computations. It provides a convenient command line interface for solving linear and nonlinear problems numerically, and for performing other numerical experiments using a language that is mostly compatible with MATLAB . GNU OCTAVE is also freely redistributable software. You may redistribute it and/or modify it under the terms of the GNU GPL as published by the Free Software Foundation.

PSAT can be adapted to run on GNU OCTAVE. PSAT has been tested on GNU OCTAVE version 3.0.0 and 3.0.1 ² for Linux and Mac OS X. The extra packages provided by the Octave-forge community are not currently needed for running PSAT.³ The following restrictions apply:

1. Only the command line usage of PSAT is allowed.
2. There is no support for SIMULINK models.
3. Only a rudimentary plotting utility is available.

29.1 Setting up PSAT for Running on GNU Octave

PSAT functions have to be adapted before being able to use PSAT on GNU OCTAVE. The Perl filter `psat2octave` does the job automatically. The steps are as follows:

1. Decompress the PSAT tarball as described in Section 2.3.
2. Be sure that the folder `filters` within the main SPAT folder is in the search path or copy the file `psat2octave` to a folder that is in your search path.

¹GNU OCTAVE is available at www.octave.org.

²Earlier versions of GNU OCTAVE have not been tested.

³Octave-forge is available at <http://octave.sourceforge.net/>.

3. Make sure that the file `psat2octave` is executable.
4. Open a terminal and move to the PST folder.
5. Launch `psat2octave`. To launch the file, keep in mind that `psat2octave` is a Perl script. Thus, on Unix-like systems is sufficient to call the script right away, while on Windows, it may be necessary to use the command `perl psat2octave`.

The script `psat2octave` takes a while for completing all necessary changes. It is possible to display some messages during the process by using the *verbose* options:

```
>> psat2octave -v
```

or the *really verbose* option, as follows:

```
>> psat2octave -w
```

If everything goes well and there are no error messages, PSAT is ready to run on GNU OCTAVE. One can revert the conversion and come back to the original PSAT distribution using the command:

```
>> psat2octave -r
```

Other options of the `psat2octave` can be found printing out the help, as follows:

```
>> psat2octave -h
```

29.1.1 How does the conversion work?

PSAT deeply exploits MATLAB classes. Unfortunately, GNU OCTAVE does not currently support the definition of custom data types such as classes. The basic idea that is behind the script `psat2octave` is to downgrade all PSAT classes to a structure and a group of functions. The major problem is to solve the issue of function “overloading” that is a basic property of classes but cannot work on GNU OCTAVE. The script `psat2octave` basically modifies PSAT functions and class methods so that all methods become function with a unique name.

A minor issue is the not full compatibility of GNU OCTAVE function with the correspondent MATLAB functions. The small differences between the two environments are taken into account through embedded code in the PSAT functions.

29.2 Basic Commands

All commands provided by the command line usage (see Chapter 28) work well on GNU OCTAVE. However, on GNU OCTAVE, the syntax

```
>> runpsat command
```

is not allowed and one of the following functional forms

```
>> runpsat('command')
>> runpsat("command")
```

must be used. Furthermore, on GNU OCTAVE, both `initpsat` and `psat` launch the command line version of PSAT, which will result in the following message:

```
< P S A T >
Copyright (C) 2002-2008 Federico Milano
Version 2.1.0
May 1, 2008
```

```
PSAT comes with ABSOLUTELY NO WARRANTY; type 'gnuwarranty'
for details. This is free software, and you are welcome to
redistribute it under certain conditions; type 'gnulicense'
for details.
```

```
Host:          Octave 3.0.0
Session:       04-May-2008 12:19:59
Usage:         Command Line
Path:          /home/fmilano/temp/psat2
```

29.3 Plot Variables

The `runpsat` function admits the additional option `plot` on GNU OCTAVE. The routine will print a menu and wait for the user answer, as follows:

```
octave:100> runpsat('plot')
Plot variables:
```

```
[ 1] States
[ 2] Voltage Magnitudes
[ 3] Voltage Angles
[ 4] Active Powers
[ 5] Reactive Powers
[ 6] Generator speeds
[ 7] Generator angles
```

```
pick a number, any number:
```

Figure 29.1 depicts an example of plot obtained using GNU OCTAVE and gplot. The graphs refers to the generator speeds of the WSCC 9-bus example.

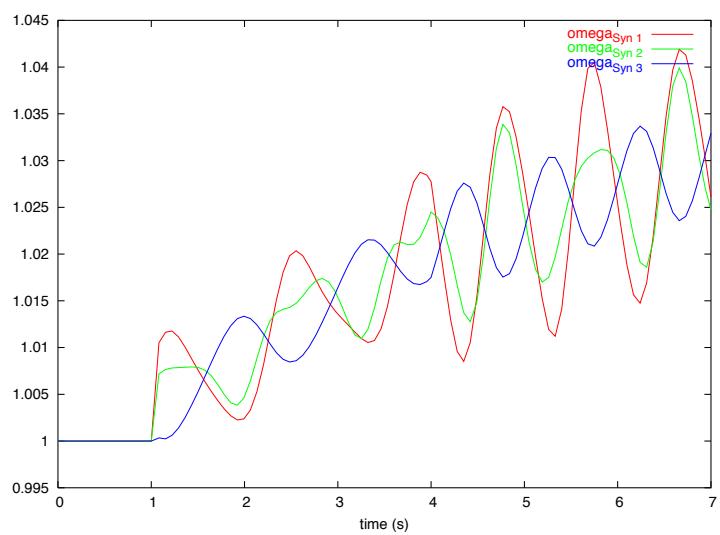
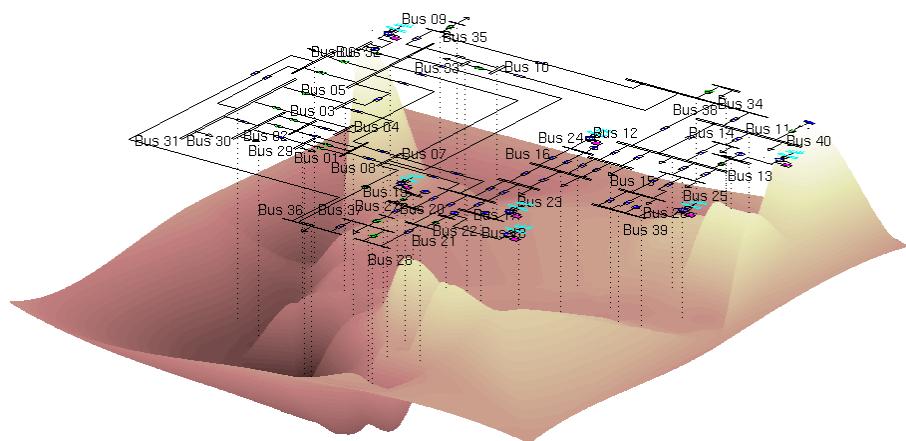


Figure 29.1: Example of graph obtained using GNU OCTAVE and gplot.

Part VI

Interfaces



Chapter 30

GAMS Interface

The General Algebraic Modeling System (GAMS) is a high-level modeling system for mathematical programming problems [17]. It consists of a language compiler and a variety of integrated high-performance solvers. GAMS is specifically designed for large and complex scale problems, and allows creating and maintaining models for a wide variety of applications and disciplines. GAMS is able to formulate models in many different types of problem classes, such as linear programming (LP), nonlinear programming (NLP), mixed-integer linear programming (MILP) and (relaxed) mixed-integer nonlinear programming (MINLP).

This chapter describes the routine and the GUI which interface PSAT to GAMS, an high-level language for the compact representation of large and complex models.

30.1 Getting Started

The use of the PSAT-GAMS interface requires you to have GAMS and a GAMS-MATLAB interface properly installed on your computer.

GAMS is available at:

www.gams.com

The website allows downloading a demo version that works properly for tiny examples. Test systems reported in the PSAT `tests` folder do not need a full pledged version of GAMS for being solved. How to install, program and use GAMS is not described here. Refer to the extensive GAMS user's guide [17] for details. In the following, it will be assumed that you have GAMS working on your computer.

The first step you have to solve is that GAMS is recognized as a command on your system. In other words, your GAMS folder must be set as an environment variable. How to set GAMS executable files as environment variables depends on the operating system, as follows:

Windows NT and **Windows 2000** look for Control Panel → System Properties → Advanced Options → Environment Variables. Then edit the “Path” by adding the full GAMS path.

Windows XP look for Control Panel → Performance and Maintenance → System. A windows with the title “System Properties” will show up. Select the “Advanced” tab and push the “Environment Variables” button. Then edit the “PATH” field by adding the full GAMS path.¹

Linux edit the `.bash_profile` file (or whatever file where your \$PATH variable is defined) in your home directory and add the full GAMS path in the \$PATH variable.

The second step is to properly set up the PSAT-GAMS interface. It is required the GAMS library `psatout.gms`, which can be found in the `~/psat/gams` folder. There are two ways to make sure that GAMS will find the library:

1. Copy the file `psatout.gms` into the folder `gamsPath/gams/inclib`. This operation generally requires to log as administrator.
2. Use the `~/psat/gams` path when running the PSAT-GAMS interface. The use of this path can be enforced by means of the PSAT-GAMS GUI (menu *Options/Include GAMS Call Options*). This may not work on all platforms (e.g. Windows). The user may also define a custom path (menu *Options/Edit GAMS Call Options*).

30.2 GAMS Solvers

OPF models used in PSAT are formulated as a set of non-linear equations. This forces the use of NLP solvers (e.g. CONOPT [44]) whose performances and results have been compared, when possible, to the ones obtained by means of the IPM implemented in PSAT. Furthermore, the solution of multi-period OPF needs a MINLP solver (e.g. DICOPT [51] and MINOS [96]), which basically works combining “relaxed” NLP with MIP master problem solutions. In large scale MINLP problems, the maximum number of integer iterations turns out to be the only possible stopping criterion. However, from the analysis of several multi-period OPF test cases, a maximum limit of 50000 integer iterations generally led to reasonable results.

30.3 PSAT-GAMS Interface

A bridge between GAMS and MATLAB allows using sophisticated nonlinear optimization tools with the visualization capabilities provided by MATLAB.

The existing MATLAB-GAMS Interface (MGI) [47] has been used as the main reference for creating the PSAT-GAMS Interface (PGI). However, the PGI does not make use of the MGI and thus you do not need to install the latter on your computer.

¹A known issue with Windows XP is that the PSAT folder needs to be the start-up folder for MATLAB. Here’s what you should do: 1) Go to your desktop in XP and right click on the MATLAB icon. 2) Indicate the full PSAT path in the destination field.

Main differences between MGI and PGI are as follows:

1. PGI is platform independent, while MGI is based on platform dependent *mex*-files.²
2. PGI is optimized for the use with PSAT only, while MGI is general purpose.
3. PGI comes with a complete GUI.
4. PGI does not require the user to know anything about GAMS programming language.
5. PGI is somewhat slower than MGI with regard to input/output file operations.

Figure 30.1 depicts the scheme of the PSAT-GAMS interface. The resulting software is a rather innovative tool able to set up large scale power system test cases, solve complex OPF problems and finally visualize results by means of a user-friendly GUI. Figure 30.2 depicts the PSAT-GAMS interface main window.

30.4 PSAT-GAMS Models

The current version of the PSAT-GAMS interface makes available five models, as follows:

1. Simple Auction
2. Market Clearing Mechanism
3. Standard OPF
4. Voltage Stability Constrained (VSC) OPF
5. Maximum Loading Condition

The VSC-OPF can be iterated for the weighting factor ω in order to produce a Pareto's set. Refer to [83] for a complete discussion of analytical models implemented in the PGI.

²A platform independent function `gams.m` is included in the PSAT distribution. This function can substitute the *mex*-files provided with the MGI tarball. However, be aware that the `gams.m` is generally slower than the correspondent *mex*-files. The PSAT distribution tarball also provides an enhanced version of the features of the GAMS library (`matout.gms`) which supports tables of any dimension. The improvements to `matout.gms` have been made in collaboration with M. C. Ferris.

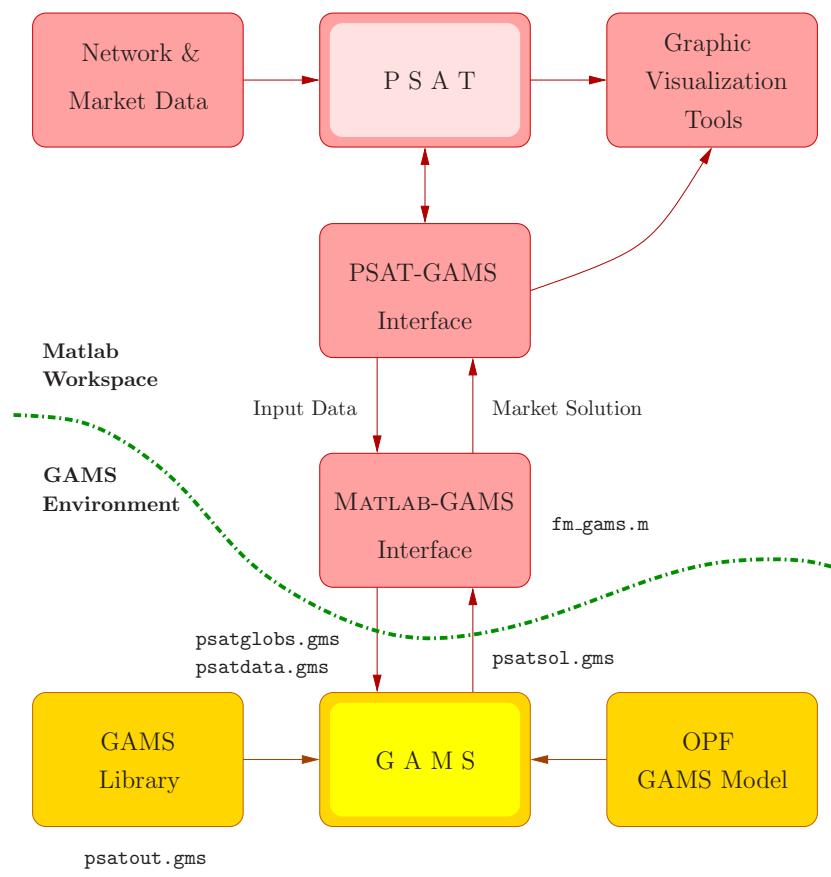


Figure 30.1: Structure of the PSAT-GAMS interface.

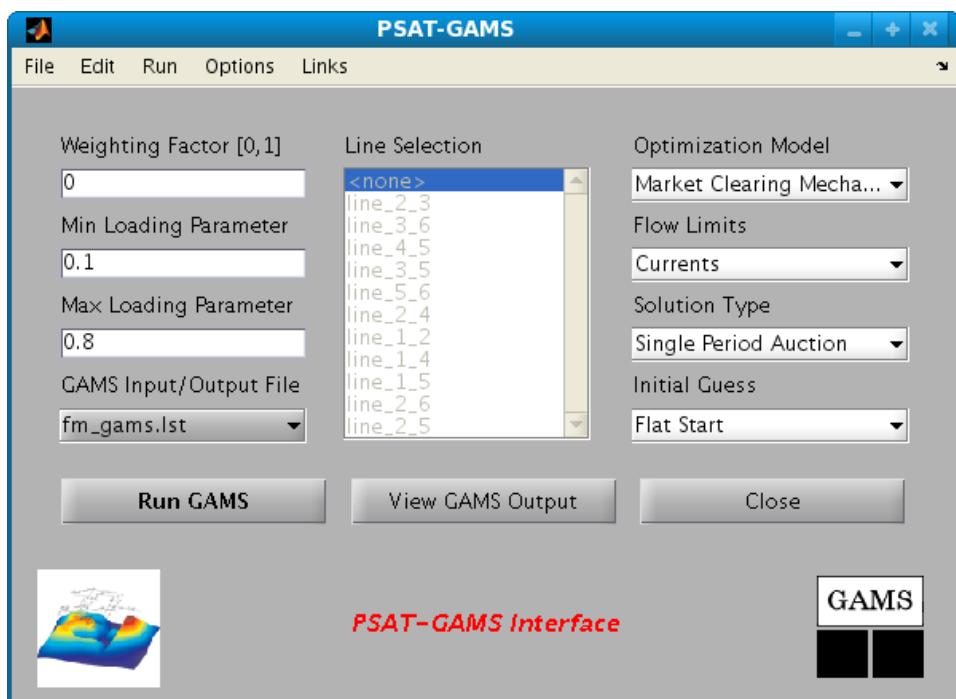


Figure 30.2: GUI of the PSAT-GAMS interface.

30.5 Multi-period Market Clearing Model

30.5.1 Notation

For the sake of clarity, constants, variables and sets used in the formulation of the multi-period market clearing models are reported below. Symbols used here follow mostly the nomenclature given in [41, 95, 40].

Constants:

$P_{S_i}^{\max}$	upper limit of the energy bid offered by unit i [MW];
$P_{S_i}^{\min}$	lower limit of the energy bid offered by unit i [MW];
$Q_{G_i}^{\max}$	upper limit of the reactive power support available at unit i [MVar];
$Q_{G_i}^{\min}$	lower limit of the reactive power support available at unit i [MVar];
P_{mn}^{\max}	flow limit in transmission line from bus m to n [MW];
x_{mn}	reactance of the transmission line from bus m to n [p.u.];
S_b	system base power [MVA];
T	scheduling time horizon (e.g. 24 hours);
DT_i	minimum down time of unit i [h];
UT_i	minimum up time of unit i [h];
SD_i	shut-down ramp limit of unit i [MW/h];
SU_i	start-up ramp limit of unit i [MW/h];
RD_i	ramp-down limit of unit i [MW/h];
RU_i	ramp-up limit of unit i [MW/h];
Γ_i	number of periods unit i must be on-line at the beginning of market horizon due to its minimum up time constraint [h];
Π_i	number of periods unit i must be off-line at the beginning of market horizon due to its minimum down time constraint [h];
α_i^0	time periods unit i has been on-line at the beginning of the market horizon (end of period 0) [h];
β_i^0	time periods unit i has been off-line at the beginning of the market horizon (end of period 0) [h];
$P_{D_j}^{\max}$	upper limit of the energy bid demanded by consumer j [MW];
$P_{D_j}^{\min}$	lower limit of the energy bid demanded by consumer j [MW];

Variables:

$\theta_b(t)$	voltage angle at bus b in period t [rad];
$P_{S_i}(t)$	power output of generation unit i in period t [MW];
$\bar{P}_{S_i}(t)$	maximum power output of generation unit i in period t [MW];
Q_{G_i}	reactive power output of unit i [MVar];
$P_{D_j}(t)$	power output of consumer j in period t [MW];
$P_{mn}(t)$	power flow from line m to line n in period t [MW];
$u_i(t)$	0/1 variable which is equal to 1 if unit i is on-line in period t ;
$w_i(t)$	0/1 variable which is equal to 1 if unit i is started-up at the beginning of period t ;
$z_i(t)$	0/1 variable which is equal to 1 if unit i is shut-down at the beginning of period t ;

Sets:

\mathcal{I}	set of indexes of generating units;
\mathcal{I}_b	subset of generating units connected at bus b ;
\mathcal{J}	set of indexes of consumers;
\mathcal{J}_b	subset of consumers connected at bus b ;
\mathcal{T}	set of indexes of periods of the market horizon;
\mathcal{B}	set of indexes of network buses;
\mathcal{N}	set of indexes of transmission lines;
\mathcal{N}_b	subset of transmission lines connected at bus b ;

30.5.2 Model Equations and Constraints

Multi-period OPF-based electricity markets are typically modeled as mixed integer linear programming problems. Equations are kept linear because of the complexity introduced by integer variables. Thus the nonlinear power flow equations are generally substituted by a power balance which may or may not include an approximated expression of network losses [41, 95].

The PSAT-GAMS interface includes ramp constraints as those that were described in [41], where the authors presents a detailed model of a multi-period auction for pool-based electricity markets. Model presented in [41] is linear, and take into account congestion in transmission lines. The PSAT-GAMS interface allows choosing between a simple power balance (“simple auction” model) and a power balance with transmission line flow limits (“market clearing mechanism” model). Both models are linear and do not take into account losses, as follows:

Simple auction:

$$\sum_{i \in \mathcal{I}} P_{S_i}(t) - \sum_{j \in \mathcal{J}} P_{D_j}(t) = 0 \quad \forall t \in \mathcal{T} \quad (30.1)$$

Market clearing model:

$$\sum_{i \in \mathcal{I}_b} P_{S_i}(t) - \sum_{j \in \mathcal{J}_b} P_{D_j}(t) - P_{m,n}(t) = 0 \quad \forall b \in \mathcal{B}, \quad \forall t \in \mathcal{T}, \quad (30.2)$$

$$\forall m, n \in \mathcal{N}, \quad \forall t \in \mathcal{T} \quad (30.3)$$

$$P_{m,n}(t) = \frac{S_b}{x_{mn}}(\theta_m - \theta_n) \quad \forall m, n \in \mathcal{N}, \quad \forall t \in \mathcal{T} \quad (30.4)$$

$$-P_{m,n}^{\max} \leq P_{m,n}(t) \leq P_{m,n}^{\min} \quad \forall m, n \in \mathcal{N}, \quad \forall t \in \mathcal{T} \quad (30.5)$$

The objective function as well as the feasibility region of generator powers have to be modified in order to take into account unit commitment of generation units and have to be extended to the scheduling time horizon T . (For instance, for daily-ahead market scheduling, $T = 24$ h.) Furthermore, a set of temporal constraints to

account for minimum up and down times, ramp up and down limits and start-up and shut-down ramp rates of generations unit has to be added to properly model thermal plants.

The objective function is:

$$\begin{aligned} \text{Max. } G = & \sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{J}} C_{D_j} P_{D_j}(t) \\ & - \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{I}} (C_{S_i} P_{S_i}(t) + C_{SU_i} w_i(t) + C_{SD_i} z_i(t)) \end{aligned} \quad (30.6)$$

where C_{SU} and C_{SD} are the start-up and shut-down costs of generating unit.

Supply bid blocks and generator reactive power limits have to take in account whether the generator is committed in the period t :

$$P_{S_i}^{\min} u_i(t) \leq P_{S_i}(t) \leq \bar{P}_{S_i}(t) \quad \forall i \in \mathcal{I}, \quad \forall t \in \mathcal{T} \quad (30.7)$$

$$Q_{G_i}^{\min} u_i(t) \leq Q_{G_i}(t) \leq Q_{G_i}^{\max} u_i(t) \quad \forall i \in \mathcal{I}, \quad \forall t \in \mathcal{T} \quad (30.8)$$

where maximum available power output limits $\bar{P}_{S_i}(t)$ are formulated in order to take into account the unit actual capacity, start-up ramp rate limits, shut-down ramp rate limits and rump-up limits, as follows:

$$\bar{P}_{S_i}(t) \leq P_{S_i}^{\max} [u_i(t) - z_i(t+1)] + z_i(t+1) S D_i \quad \forall i \in \mathcal{I}, \quad \forall t \in \mathcal{T} \quad (30.9)$$

$$\bar{P}_{S_i}(t) \leq P_{S_i}(t-1) + R U_i u_i(t-1) + S U_i w_i(t) \quad \forall i \in \mathcal{I}, \quad \forall t \in \mathcal{T}$$

The ramp-down rate limit and the shut-down ramp rate limit are modeled as follows:

$$P_{S_i}(t-1) \leq P_{S_i}(t) + R D_i u_i(t) + S D_i z_i(t) \quad \forall i \in \mathcal{I}, \quad \forall t \in \mathcal{T} \quad (30.10)$$

Equations (30.9) and (30.10) model start-up and shut-down constraints in a more detailed way than the one commonly used in the literature [77, 134], i.e.

$$P_{S_i}(t) - P_{S_i}(t-1) \leq R U_i \quad \forall i \in \mathcal{I}, \quad \forall t \in \mathcal{T} \quad (30.11)$$

$$P_{S_i}(t-1) - P_{S_i}(t) \leq R D_i \quad \forall i \in \mathcal{I}, \quad \forall t \in \mathcal{T}$$

since in (30.11) start-up and shut-down variables are used instead of ramp-up and ramp-down limits as in (30.9) and (30.10). Minimum on-line and off-line time constraints are formulated as presented in [41] and in [40]. These are as follows:

Minimum up time:

$$\begin{aligned} \sum_{t=1}^{\Gamma_i} (1 - u_i(t)) &= 0 \quad \forall i \in \mathcal{I} \\ \sum_{\tau=t}^{k+UT_i-1} u_i(\tau) &\geq U T_i w_i(t) \quad \forall i \in \mathcal{I}, \\ &\forall t = \Gamma_i + 1 \dots T - U T_i + 1 \\ \sum_{\tau=t}^T (u_i(\tau) - w_i(t)) &\geq 0 \quad \forall i \in \mathcal{I}, \\ &\forall t = T - U T_i + 2 \dots T \end{aligned} \quad (30.12)$$

Minimum down time:

$$\begin{aligned}
 \sum_{t=1}^{\Pi_i} u_i(t) &= 0 \quad \forall i \in \mathcal{I} \\
 \sum_{\tau=t}^{t+DT_i-1} (1 - u_i(\tau)) &\geq DT_i z_i(t) \quad \forall i \in \mathcal{I}, \\
 &\forall t = \Pi_i + 1 \dots T - DT_i + 1 \\
 \sum_{\tau=t}^T (1 - u_i(\tau) - z_i(t)) &\geq 0 \quad \forall i \in \mathcal{I}, \\
 &\forall t = T - DT_i + 2 \dots T
 \end{aligned} \tag{30.13}$$

where Γ_i and Π_i are the number of periods unit i must be on-line and off-line at the beginning of the time horizon respectively, as follows:

$$\Gamma_i = \min\{T, (UT_i - \alpha_i^0)u_i(0)\} \tag{30.14}$$

$$\Pi_i = \min\{T, (DT_i - \beta_i^0)(1 - u_i(0))\} \tag{30.15}$$

Finally, the start-up and the shut-down status of the units are managed as follows:

$$\begin{aligned}
 w_i(t) - z_i(t) &= u_i(t) - u_i(t-1) \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T} \\
 w_i(t) + z_i(t) &\leq 1 \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}
 \end{aligned} \tag{30.16}$$

Equations (30.16) are necessary to avoid simultaneous commitment and decommitment of a unit. Observe that a single-period market with unit commitment can be directly derived from (30.1)-(30.16) by imposing a scheduling time $T = 1$ h.

30.6 Example

This section illustrates how PSAT and the PSAT-GAMS interface works through a simple example. At this aim, let us consider the three-bus test system described in Appendix F.1.

Firstly the user has to set up the data in the PSAT format. This can be done by writing a MATLAB script file or, better, using the PSAT-SIMULINK library. Figure 30.3 depicts the resulting SIMULINK model which represents the three-bus test system. Each block of the diagram hides a mask where the user can set up the data associated with the correspondent component.

Once the model is completed, it has to be loaded in the MATLAB workspace. To load a file simply double click on this edit text, or use the first button of the tool bar, the menu *File/Open/Data File* or the shortcut <Ctrl-d> when the main window is active. The name of this file is always displayed in the edit text *Data File* of the main window.

Now, it is possible to solve the power flow, which can be launched by clicking on the “Power Flow” button in the main window. Power flow results can be visualized

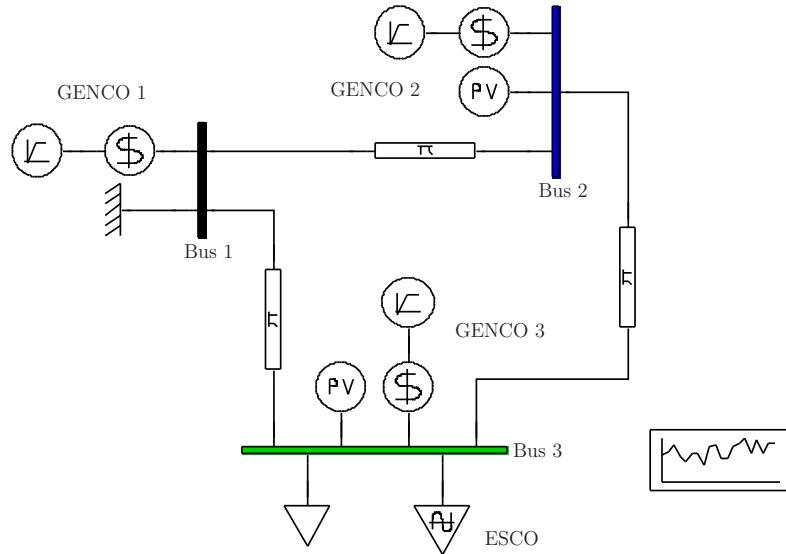


Figure 30.3: PSAT-SIMULINK model of the three-bus test system.

for a first inspection in the Static Report GUI (which can be launched by <Ctrl-v> from the main window) and saved in a report file.

After solving the base case power flow, PSAT is ready for further analysis. Observe that all variables, data and results are stored as global structures in the MATLAB workspace so that they are available for other routines and can be inspected at any time by the user.

For the sake of comparison, Tables 30.1 and 30.2 depict the solution of the single period OPF problem obtained with the IPM MATLAB routine and the PSAT-GAMS interface respectively. Tables 30.3, 30.4, and 30.5 depict the input and output files which are used for exchanging data between PSAT and GAMS.

Figure 30.4 illustrates the demand profile for a 5 hour time horizon, while Figures 30.5 and 30.6 depict the supply and LMP profiles with and without P_{mn}^{\max} limits. Observe that enforcing congestion limits leads not only to redistribute power supplies but also to split the market clearing price into nodal marginal prices.

Table 30.1: PSAT IPM-based OPF report for the three-bus test system.

OPTIMAL POWER FLOW REPORT (Standard OPF)						
P S A T 1.3.3						
Author: Federico Milano, (c) 2002-2005 e-mail: fmilano@thunderbox.uwaterloo.ca website: http://thunderbox.uwaterloo.ca/~fmilano						
File: ~/psatd/tests/d_unitcomm.mdl Date: 13-Jul-2005 09:19:39						
NETWORK STATISTICS						
Buses:	3					
Lines:	3					
Generators:	3					
Loads:	1					
Supplies:	3					
Demands:	1					
SOLUTION STATISTICS						
Objective Function [\$/h]:	1606.2045					
Active Limits:	7					
Number of Iterations:	12					
Barrier Parameter:	0					
Variable Mismatch:	0					
Power Flow Equation Mismatch:	0					
Objective Function Mismatch:	0					
POWER SUPPLIES						
Bus	mu min	Ps min [MW]	Ps [MW]	Ps max [MW]	mu max	
Bus1	0	10	52.4252	60	0	
Bus2	0	10	27.1322	60	0	
Bus3	0	10	20.4425	60	0	
POWER DEMANDS						
Bus	mu min	Pd min [MW]	Pd [MW]	Pd max [MW]	mu max	
Bus3	1294562.538	100	100	100	1294539.717	
REACTIVE POWERS						
Bus	mu min	Qg min [MVar]	Qg [MVar]	Qg max [MVar]	mu max	
Bus1	0	-150	0.82938	150	0	
Bus2	0	-20	0.55216	80	0	
Bus3	0	-20	61.3228	80	0	
VOLTAGES						
Bus	mu min	V min [p.u.]	V [p.u.]	V max [p.u.]	mu max	phase [rad]
Bus1	0	0.9	1.1	1.1	0.35091	0
Bus2	0	0.9	1.1	1.1	0.50728	-0.00697
Bus3	0	0.9	1.1	1.1	0.66381	-0.03637
POWER FLOW						
Bus	P [MW]	Q [MVar]	rho P [\$/MWh]	rho Q [\$/MVarh]	NCP [\$/MWh]	Pay [\$/h]
Bus1	52.4252	0.82937	20.285	0	0	-1063
Bus2	27.1322	0.55216	21.5529	0	1.2679	-585
Bus3	-79.5575	1.3228	22.8213	0	2.5362	1816
FLOW IN TRANSMISSION LINES						
From bus	To bus	I _{ij} [p.u.]	I _{ij} max [p.u.]	mu I _{ij}		
Bus1	Bus2	0.07666	0.4	0		
Bus1	Bus3	0.4	0.4	2.6146		
Bus2	Bus3	0.32335	0.4	0		
FLOW IN TRANSMISSION LINES						
From bus	To bus	I _{ji} [p.u.]	I _{ji} max [p.u.]	mu I _{ji}		
Bus2	Bus1	0.07666	0.4	0		
Bus3	Bus1	0.4	0.4	2.6146		
Bus3	Bus2	0.32335	0.4	0		
TOTALS						
TOTAL LOSSES [MW]:	0					
BID LOSSES [MW]:	0					
TOTAL DEMAND [MW]:	100					
TTL [MW]:	100					
IMO PAY [\$/h]:	167.3758					

Table 30.2: PSAT-GAMS OPF report for the three-bus test system.

PSAT-GAMS Interface						
<hr/>						
Standard OPF						
Single-Period Auction						
GAMS routine completed in 0.11565 s						
<hr/>						
Power Supplies						
<hr/>						
Bus	Ps	Ps max	Ps min			
<1>	[MW]	[MW]	[MW]			
1	52.4252	60	10			
2	27.1322	60	10			
3	20.4425	60	10			
<hr/>						
Power Demands						
<hr/>						
Bus	Pd	Pd max	Pd min			
<1>	[MW]	[MW]	[MW]			
3	100	100	100			
<hr/>						
Generator Reactive Powers						
<hr/>						
Bus	Qg	Qg max	Qg min			
<1>	[MVar]	[MVar]	[MVar]			
1	0.8294	150	-150			
2	0.5522	80	-20			
3	61.3228	80	-20			
<hr/>						
Power Flow Solution						
<hr/>						
Bus	V	theta	PG	PL	QG	QL
<1>	[p.u.]	[rad]	[MW]	[MW]	[MVar]	[MVar]
1	1.1000	0.0000	52.4252	0	0.8294	0
2	1.1000	-0.0070	27.1322	0	0.5522	0
3	1.1000	-0.0364	20.4425	100	61.3228	60
<hr/>						
Prices and Pays						
<hr/>						
Bus	LMP	NCP	Pay S	Pay D		
<1>	[\$/MWh]	[\$/MWh]	[\$/h]	[\$/h]		
1	20.2850	0.0000	-1063.4480	0.0000		
2	21.5529	1.1006	-584.7785	0.0000		
3	22.8213	2.4792	-466.5246	2282.1267		
<hr/>						
Flows on Transmission Lines						
<hr/>						
From Bus	To Bus	Iij	Iijmax	Iij margin	Iji	Ijimax
<1>	<j>	[p.u.]	[p.u.]	[p.u.]	[p.u.]	[p.u.]
1	2	0.0767	0.4000	0.3233	0.0767	0.4000
1	3	0.4000	0.4000	0.0000	0.4000	0.4000
2	3	0.3234	0.4000	0.0766	0.3234	0.4000
<hr/>						
Totals						
<hr/>						
Total Losses = 0 [MW]						
Bid Losses = 0 [MW]						
Total demand = 100 [MW]						
Total Transaction Level = 100 [MW]						
IMO Pay = 167.3758 [\$/h]						
<hr/>						
Check file "/psatd/fm_gams.lst for GAMS report.						
GAMS model status: locally optimal						
GAMS solver status: normal completion						
PSAT-GAMS Optimization Routine completed in 0.34138 s						

Table 30.3: Input file psatglobs.gms for the three-bus test system.

```
$setglobal nBus '3'
$setglobal nLine '3'
$setglobal nPs '3'
$setglobal nD '1'
$setglobal nSW '1'
$setglobal nPV '2'
$setglobal nBusref '1'
$setglobal control '3'
$setglobal flow '1'
```

Table 30.4: Input file `psatdata.gms` for the three-bus test system.

\$onempty	\$kill Pd_idx	1.VO 1.000000
\$kill Gh	parameter Pd_idx /	2.VO 1.000000
parameter Gh /	3..1 1.000000	3.VO 1.000000
/;	/;	1.Qgmax 1.500000
\$kill Bh	\$kill S	2.Qgmax 0.800000
parameter Bh /	parameter S /	3.Qgmax 0.800000
1..1 -20.000000	1..Pmax 0.600000	1.Qgmin -1.500000
2..1 10.000000	2..Pmax 0.600000	2.Qgmin -0.200000
3..1 10.000000	3..Pmax 0.600000	3.Qgmin -0.200000
1..2 10.000000	1..Pmin 0.100000	1.Vmax 1.100000
2..2 -20.000000	2..Pmin 0.100000	2.Vmax 1.100000
3..2 10.000000	3..Pmin 0.100000	3.Vmax 1.100000
1..3 10.000000	1..Csa 0.060000	1.Vmin 0.900000
2..3 10.000000	2..Csa 0.040000	2.Vmin 0.900000
3..3 -20.000000	3..Csa 0.080000	3.Vmin 0.900000
/;	1..Csb 9.800000	1.ksw 1.000000
\$kill Li	2..Csb 10.700000	2.kpv 1.000000
parameter Li /	3..Csb 12.600000	3.kpv 1.000000
1..1 1.000000	1..Csc 10.000000	/;
2..1 1.000000	2..Csc 20.000000	\$kill N
3..2 1.000000	3..Csc 25.000000	parameter N /
/;	1..ksu 1.000000	1.b -10.000000
\$kill Lj	2..ksu 1.000000	2.b -10.000000
parameter Lj /	3..ksu 1.000000	3.b -10.000000
1..2 1.000000	/;	1.Pjmax 0.400000
2..3 1.000000	\$kill D	2.Pjmax 0.400000
3..3 1.000000	parameter D /	3.Pjmax 0.400000
/;	1..Pd0 1.000000	1.Pjmax 0.400000
\$kill Ps_idx	1..Pdmax 1.000000	2.Pjmax 0.400000
parameter Ps_idx /	1..Pdmin 1.000000	3.Pjmax 0.400000
1..1 1.000000	1..tgphi 0.600000	/;
2..2 1.000000	/;	\$offempty
3..3 1.000000	\$kill X	parameter X /
/;		

Table 30.5: Output file `psatsol.m` for the three-bus test system.

nout = nout + 1;	varargout{nout} = zeros(3,1);	varargout{nout}(3) = 2.2821267457505E+01;
varargout{nout}(1) = 5.2425221822350E-01;	varargout{nout}(2) = 2.7132243262640E-01;	nout = nout + 1;
varargout{nout}(3) = 2.0442534915010E-01;	nout = nout + 1;	varargout{nout} = zeros(3,1);
varargout{nout} = zeros(1,1);	varargout{nout}(1) = 1.0000000000000E+00;	varargout{nout}(1) = 7.6659512222775E-02;
varargout{nout}(1) = 1.0000000000000E+00;	nout = nout + 1;	varargout{nout}(2) = 4.0000000000000E-01;
varargout{nout} = zeros(3,1);	varargout{nout}(2) = 3.2335073143047E-01;	varargout{nout}(3) = 3.2335073143047E-01;
varargout{nout}(1) = 1.1000000000000E+00;	nout = nout + 1;	varargout{nout} = zeros(3,1);
varargout{nout}(2) = 1.1000000000000E+00;	varargout{nout}(1) = 7.6659512222775E-02;	varargout{nout}(1) = 3.2335073143047E-01;
varargout{nout}(3) = 1.1000000000000E+00;	varargout{nout}(2) = 4.0000000000000E-01;	varargout{nout}(2) = -5.0727651287779E-01;
nout = nout + 1;	varargout{nout}(3) = 3.2335073143047E-01;	varargout{nout}(3) = -6.6380624368687E-01;
varargout{nout} = zeros(3,1);	nout = nout + 1;	nout = nout + 1;
varargout{nout}(1) = -6.9690606686979E-03;	varargout{nout} = zeros(3,1);	varargout{nout} = zeros(3,1);
varargout{nout}(2) = -3.6365640167873E-02;	varargout{nout}(1) = -3.5091452691537E-01;	varargout{nout}(2) = -5.0727651287779E-01;
nout = nout + 1;	varargout{nout}(2) = -6.6380624368687E-01;	varargout{nout}(3) = -6.6380624368687E-01;
varargout{nout} = zeros(3,1);	nout = nout + 1;	nout = nout + 1;
varargout{nout}(1) = 8.2938340399436E-03;	varargout{nout} = zeros(3,1);	varargout{nout} = zeros(3,1);
varargout{nout}(2) = 5.5216188162457E-03;	nout = nout + 1;	nout = nout + 1;
varargout{nout}(3) = 6.1322778477527E-01;	varargout{nout} = 2.0000000000000E+00;	varargout{nout} = 2.0000000000000E+00;
nout = nout + 1;	nout = nout + 1;	nout = nout + 1;
varargout{nout} = zeros(3,1);	varargout{nout}(1) = 2.0285044364470E+01;	varargout{nout} = 1.0000000000000E+00;
varargout{nout}(1) = 2.0285044364470E+01;	varargout{nout}(2) = 2.1552697306532E+01;	

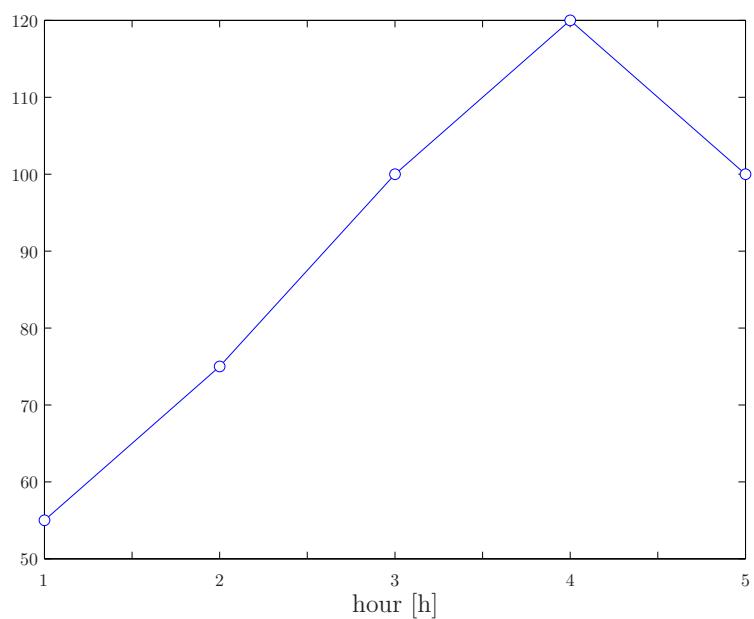


Figure 30.4: Demand profile for the multi-period auction.

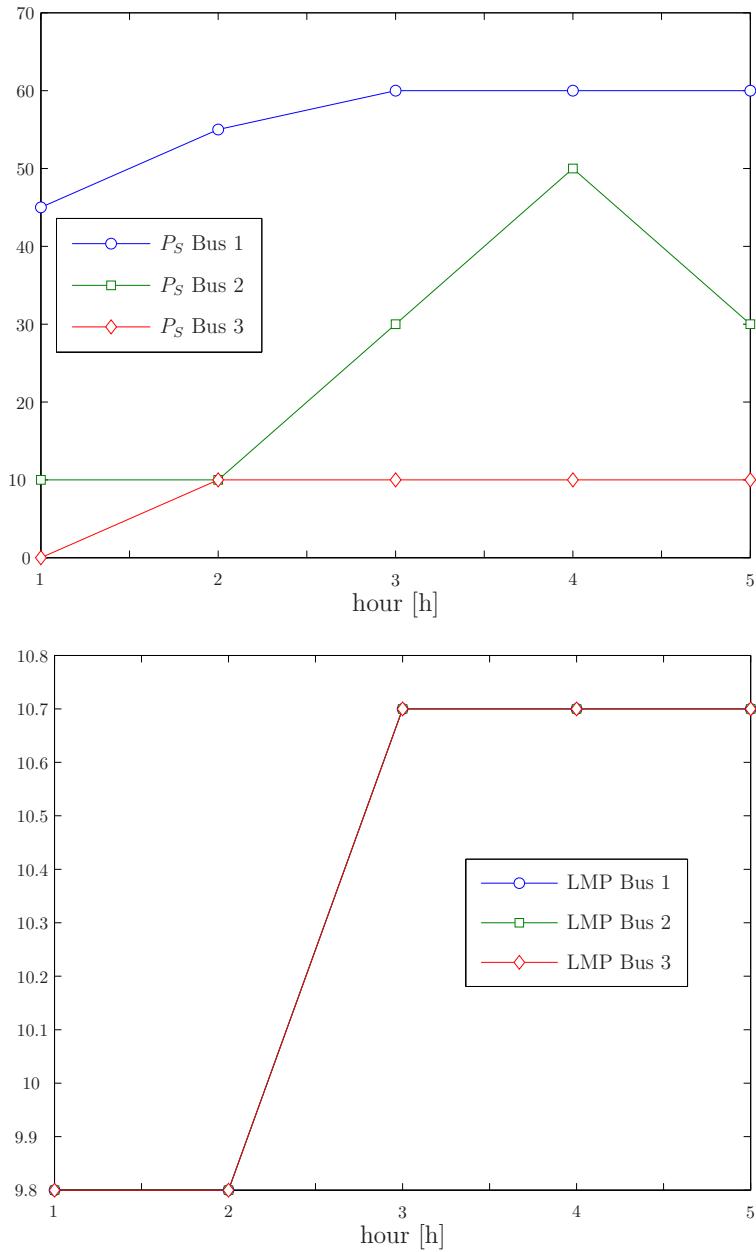


Figure 30.5: Supply and LMP profiles for the multi-period auction without P_{mn}^{\max} limits.

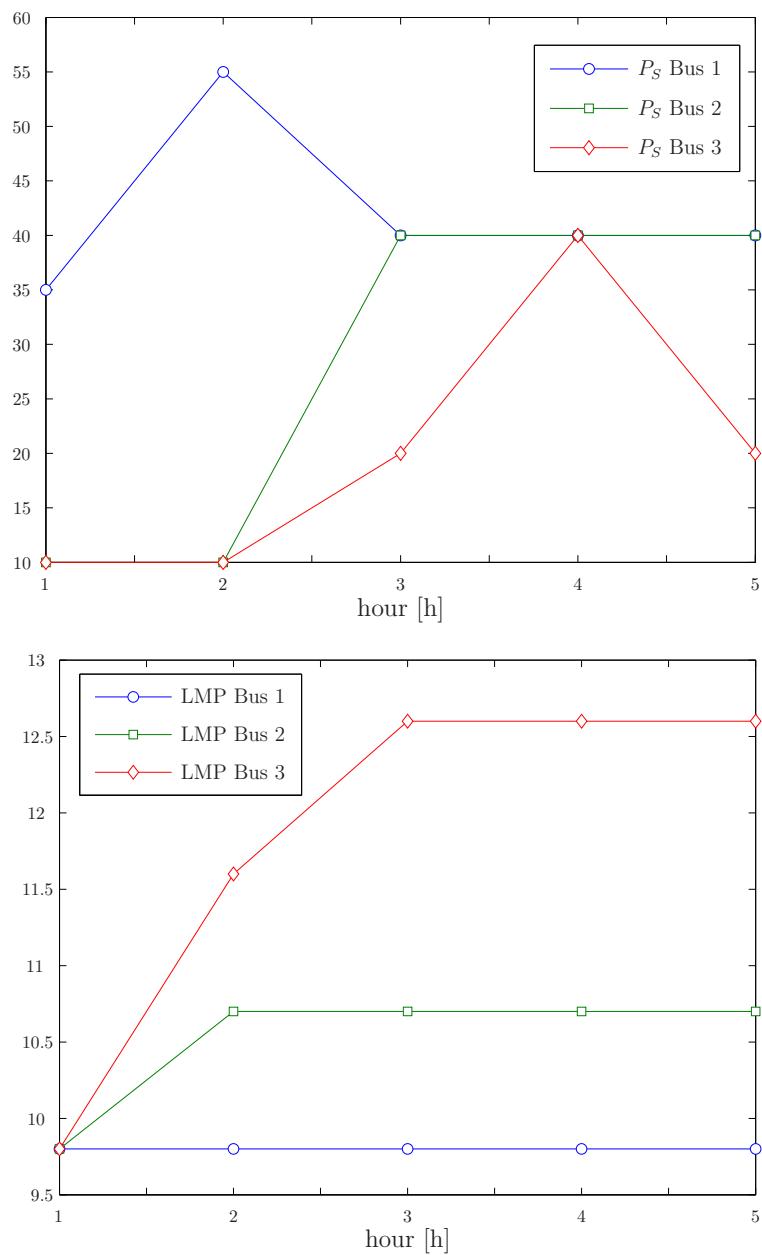


Figure 30.6: Supply and LMP profiles for the multi-period auction with P_{mn}^{\max} limits.

Chapter 31

UWPFLOW Interface

UWPFLOW is an open source program for sophisticated continuation power flow analysis [22]. It consists of a set of C functions and libraries designed for voltage stability analysis of power systems, including voltage dependent loads, HVDC, FACTS and secondary voltage control.

This chapter describes the PSAT-UWPFLOW interface, which allows exporting PSAT models to UWPFLOW, and provides a simple example. The interface is currently in an early stage; refer to Section 31.3 for limitations and ToDos.

31.1 Getting Started

The use of the PSAT-UWPFLOW interface requires you have UWPFLOW installed on your computer. UWPFLOW is freely available at

www.power.uwaterloo.ca

Unix and Linux¹ users have just to follow installation instructions provided with the UWPFLOW tarball. Windows users have an extra work to do in order to get the PSAT-UWPFLOW interface properly working, as follows:

1. After installing the Windows version of UWPFLOW, look for the UWPFLOW folder and rename the `uwpfow.exe` (e.g. `uwpfow_ide.exe`). Remember to change the path in the UWPFLOW desktop icon, if you have one. These changes do not affect the Windows version of UWPFLOW, which will just keep working fine.
2. Move to the UWPFLOW source folder and compile UWPFLOW from scratch. I used `make` and `gcc` for win32 provided by CygWin.² If you are using `gcc` as C compiler, remember to modify the UWPFLOW `makefile`, i.e. change

¹On some Linux platforms, such as Red Hat, UWPFLOW may produce segmentation faults when trying to display results. To avoid that, comment line 569 `/*fclose(OutputHomot);*/` of file `writesol.c`. Then compile UWPFLOW.

²available at www.cygwin.com

the first line as follows: `CC = gcc`. If you are using the CygWin package, the compiler will produce two files, `uwpflow.exe` and `cygwin1.dll`.³

3. Copy the UWPFLOW executable file(s) created at the previous step in a Windows system folder, such as `C:\Windows\system32`.

31.2 Graphical User Interface

Figure 31.1 depicts the GUI of the PSAT-UWPFLOW interface. The user has just to set the desired options and then push the Run button. The GUI may be also used just as a generator of the command line for UWPFLOW.

31.3 Limitations and ToDos

The PSAT-UWPFLOW interface is in very early stage and is currently able to export very simple power flow models. That means voltage dependent loads, HVDC, FACTS and secondary voltage control are not supported yet. UWPFLOW does not support dynamic models; thus the interface will work successfully with networks containing **only** lines, slack generators, PV generators, PQ loads and shunt admittances (i.e. the components described in Chapter 12).

The interface allows exporting PSAT models to UWPFLOW and running UWPFLOW with the proper options, but not the other way round, i.e. UWPFLOW results and the power flow solutions are not loaded in PSAT. The user is always asked if UWPFLOW power flow solution should be loaded in PSAT. Continuation power flow solutions, i.e. nose curves, are plotted in a MATLAB window.

A list of ToDos follows:

1. make possible to export voltage dependent loads, HVDC, FACTS and secondary voltage control;
2. make possible to load all UWPFLOW results in PSAT;
3. make possible to visualize UWPFLOW nose curves in PSAT;
4. add a batch file support to run sophisticated UWPFLOW sessions.

Improvements to the PSAT-UWPFLOW interface will be included in future versions of PSAT.

³Users which are not familiar with Unix-like systems, could find a little bit confusing dealing with the `make` and `gcc` utilities. If you have no clue on how to compile UWPFLOW from scratch, the executable files can be found in the PSAT Forum file repository (see Appendix H). However, be aware that the reference UWPFLOW version will remain the one that is distributed by Prof. C. Cañizares.

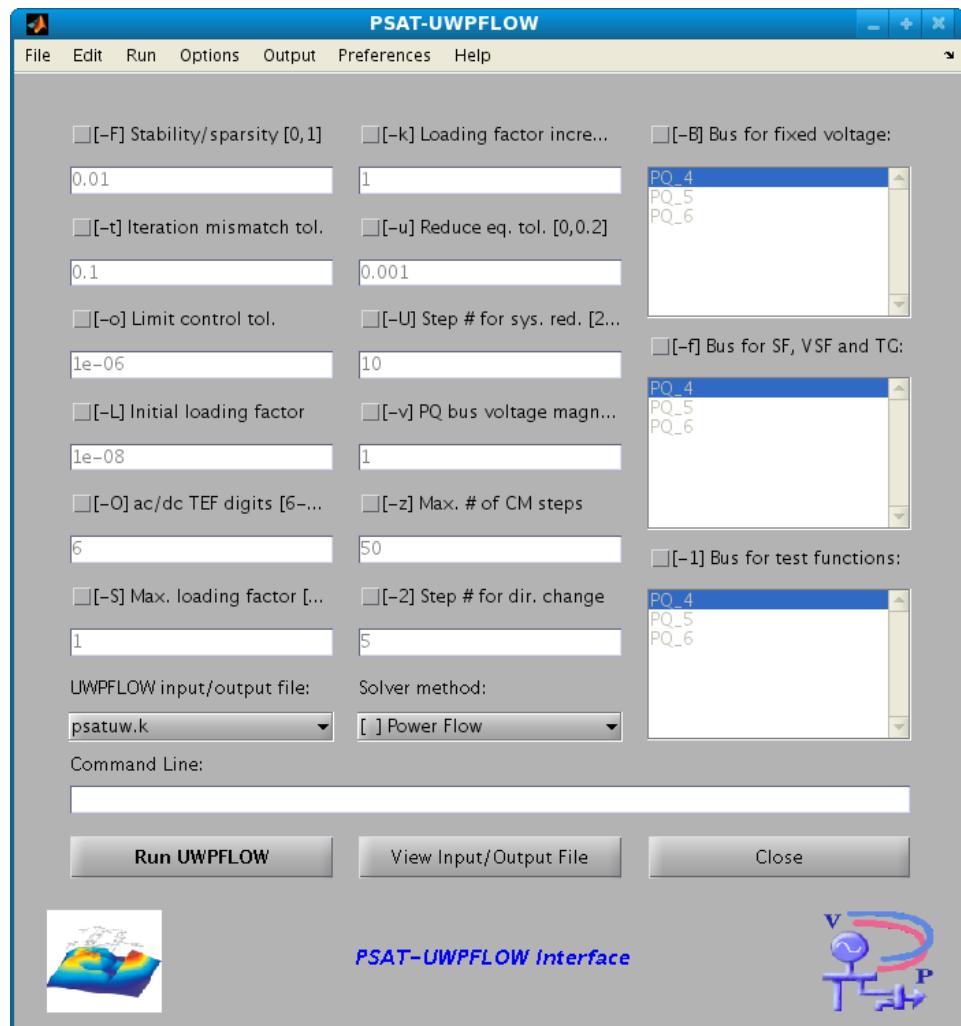


Figure 31.1: GUI of the PSAT-UWPFLOW interface.

31.4 Example

This section illustrates the usage of the PSAT-UWPFLOW interface by means of the 6-bus test system, whose data are reported in Appendix F.2.

The network has to be loaded in PSAT and the power flow has to be solved, as usual. At this point, the PSAT-UWPFLOW interface can be launched.

For example, in order to run the power flow, simply select *Power Flow* in the *Solver Method* pop-up menu and, if needed, set the desired UWPFLOW options. Pushing the *Run UWPFLOW* button will launch the PSAT-UWPFLOW interface, which creates the UWPFLOW command line, as follows:

```
uwpflow -I d_006.mdl.cf psatuw.pf
```

The interface will also write a IEEE common data format file containing the current system data, as depicted in Table 31.1.⁴

Results are stored in the file `psatuw.pf`,⁵ which is located in the same folder as the PSAT data file (see Table 31.2). At the end of the computations, the user may chose to load these results in PSAT. Observe that in order to load results, it is used the file `psatuw.cf`⁶ which is in the IEEE common data format.⁷ Because of the limited number of digits available for voltages, the UWPFLOW solution can present “big” equation mismatches when loaded in PSAT.

To run the continuation power flow, select *Continuation Method* in the *Solver Method* pop-up menu and, if needed, set the desired UWPFLOW options. For the 6-bus test system, the command line will result as follows:

```
uwpflow -I d_006.mdl.cf psatuw.pf -cpsatuw.cpf -Kpsatuw.k
```

Observe that along with the `d_006.mdl.cf` file, the interface has to take care of the input file `psatuw.k`, which provide power direction for the continuation method. At this aim, **Supply** and **Demand** data are used. If **Supply** and **Demand** data are not defined, base case powers will be used, i.e. the powers of slack and PV generators and PQ loads. Table 31.3 depicts the `psatuw.k` file for the 6-bus test system.

In the case of continuation methods, UWPFLOW writes a file containing the loading parameters and the most significant voltages, as depicted in Table 31.4. When this file is created, the PSAT-UWPFLOW interface will load and display the data in a MATLAB figure, as depicted in Fig. 31.2. Observe that nose curves obtained by means of the PSAT-UWPFLOW interface are not internally loaded in PSAT and cannot be plotted using the PSAT GUI for plotting results.

⁴The PSAT-UWPFLOW interface also performs a few syntax checks of the resulting UWPFLOW command line. In some cases some options are added in order to build a well formed command line.

⁵The user may chose another name for this file using the menu *Preferences/Modify Input-Output File Name* of the PSAT-UWPFLOW interface.

⁶Or whatever is the “Input-Output File Name” chosen by the user.

⁷This file is always created by the PSAT-UWPFLOW interface, by means of the `-w` option. This option is not shown in the resulting command line, unless the user set the option in the PSAT-UWPFLOW interface.

Table 31.1: IEEE CDF file to be used within UWPFLOW (d_006.mdl.cf)

11/16/03 PSAT ARCHIVE 100.00 2003 W 6-Bus 11-Line System																	
BUS DATA FOLLOW 6 ITEMS																	
1	Bus1	1	0	2	1.0500	0.000	0.0000	0.0000	90.000	0.000	400.00	1.0500	150.00	-150.00	0.0000	0.0000	1
2	Bus2	1	0	3	1.0500	0.000	0.0000	0.0000	140.000	0.000	400.00	1.0500	150.00	-150.00	0.0000	0.0000	2
3	Bus3	1	0	2	1.0500	0.000	0.0000	0.0000	60.000	0.000	400.00	1.0500	150.00	-150.00	0.0000	0.0000	3
4	Bus4	1	0	1	1.0000	0.000	90.0000	60.0000	0.000	0.000	400.00	0.0000	0.00	0.00	0.0000	0.0000	4
5	Bus5	1	0	1	1.0000	0.000	100.0000	70.0000	0.000	0.000	400.00	0.0000	0.00	0.00	0.0000	0.0000	5
6	Bus6	1	0	1	1.0000	0.000	90.0000	60.0000	0.000	0.000	400.00	0.0000	0.00	0.00	0.0000	0.0000	6
-999																	
BRANCH DATA FOLLOW 11 ITEMS																	
2	3	1	1	1	0	0.0500000	0.2500000	0.0600000	0	0	0	0	0.0000	0.0000	0.0000	0.0000	0.0000
3	6	1	1	1	0	0.0200000	0.1000000	0.0200000	0	0	0	0	0.0000	0.0000	0.0000	0.0000	1.3973
4	5	1	1	1	0	0.2000000	0.4000000	0.0800000	0	0	0	0	0.0000	0.0000	0.0000	0.0000	0.1796
3	5	1	1	1	0	0.1200000	0.2600000	0.0500000	0	0	0	0	0.0000	0.0000	0.0000	0.0000	0.6585
5	6	1	1	1	0	0.1000000	0.3000000	0.0600000	0	0	0	0	0.0000	0.0000	0.0000	0.0000	0.2000
2	4	1	1	1	0	0.0500000	0.1000000	0.0200000	0	0	0	0	0.0000	0.0000	0.0000	0.0000	1.3740
1	2	1	1	1	0	0.1000000	0.2000000	0.0400000	0	0	0	0	0.0000	0.0000	0.0000	0.0000	0.2591
1	4	1	1	1	0	0.0500000	0.2000000	0.0400000	0	0	0	0	0.0000	0.0000	0.0000	0.0000	0.9193
1	5	1	1	1	0	0.0800000	0.3000000	0.0600000	0	0	0	0	0.0000	0.0000	0.0000	0.0000	0.8478
2	6	1	1	1	0	0.0700000	0.2000000	0.0500000	0	0	0	0	0.0000	0.0000	0.0000	0.0000	0.9147
2	5	1	1	1	0	0.1000000	0.3000000	0.0400000	0	0	0	0	0.0000	0.0000	0.0000	0.0000	0.7114
-999																	
LOSS ZONES FOLLOW 1 ITEMS																	
1 6-Bus																	
-99																	
INTERCHANGE DATA FOLLOW 1 ITEMS																	
1	2	Bus2	0.00	999.99	6Bus	6-Bus 11-Line System											
-9																	
TIE LINES FOLLOW 0 ITEMS																	
-999																	
END OF DATA																	

Table 31.2: UWPFLOW power flow results (psatuw.pf)

U.E.P. Solution:
6-Bus 11-Line System

Loading factor -> 5.39346e-05
AC buses -> 6
PV buses -> 0
X buses -> 0
Z buses -> 0
AC elem. -> 11
V Reg. Trf. -> 0
PQ Reg. Trf. -> 0
DC buses -> 0
DC lines -> 0
SVCs -> 0
TSCCs -> 0
STATCOMs -> 0
No. Areas -> 0
Iterations -> 30 (Maximum = 50)
Max. p.u. mismatch -> 9.374e-07 (Tolerance = 0.0001)
Reference Bus(es) -> 2 Bus2 (Angle= 0.00 deg.)

***** AC RESULTS *****

		L=lower limit		H=higher limit		O=over limit		U=under limit									
A	i	Bus	V(pu)	V(kV)	Pg(MW)	Pload	Pshuntl	j	Bus	C	Pij	Plosses	ij (A)	kVi/kVj	T	Controlled Bus	
n		Name	d(deg)	d(rad)	Qg(MVAR)	Qload	Qshuntl		Name	r	Qij	Qlosses		a(deg)	k	Name	
0	1	Bus1	0.6536	261.43	114.55	0.00	0.00	5	Bus5	1	47.63	15.41	197.91				
			3.70	0.0646	150.00H	0.00	0.00				75.91	56.28					
									4	Bus4	1	48.95	7.74	178.02			
											64.05	29.72					
									2	Bus2	1	17.97	1.03	45.47			
											10.04	0.51					
0	2	Bus2	0.5940	237.58	170.69	0.00	0.00	5	Bus5	1	39.83	13.11	163.88				
			0.00	0.0000	150.00H	0.00	0.00				54.42	38.48					
									6	Bus6	1	53.32	9.42	166.12			
											42.77	26.63					
									1	Bus1	1	-16.94	1.03	47.24			
											-9.53	0.51					
									4	Bus4	1	75.27	13.81	239.30			
											63.50	27.08					
									3	Bus3	1	19.21	0.52	46.77			
											-1.16	0.53					
0	3	Bus3	0.5838	233.54	84.55	0.00	0.00	5	Bus5	1	41.06	16.02	165.11				
			-7.97	-0.1391	150.00H	0.00	0.00				52.67	33.68					
									6	Bus6	1	62.17	7.67	281.99			
											95.63	37.86					
									2	Bus2	1	-18.69	0.52	46.39			
											1.69	0.53					
0	4	Bus4	0.4293	171.74	0.00	90.00	0.00	1	Bus1	1	-41.21	7.74	180.32				
			-9.79	-0.1708	0.00	60.00	0.00				-34.33	29.72					
									2	Bus2	1	-61.46	13.81	240.17			
											-36.42	27.08					
									5	Bus5	1	12.67	3.17	55.87			
											10.75	5.32					
0	5	Bus5	0.2711	108.42	0.00	100.00	0.00	2	Bus2	1	-26.72	13.11	165.67				
			-23.56	-0.4112	0.00	70.00	0.00				-15.94	38.48					
									1	Bus1	1	-32.22	15.41	200.92			
											-19.63	56.28					
									6	Bus6	1	-6.52	1.88	63.55			
											-10.00	4.93					
									3	Bus3	1	-25.04	16.02	167.37			
											-19.00	33.68					
									4	Bus4	1	-9.50	3.17	58.27			
											-5.43	5.32					
0	6	Bus6	0.4049	161.97	0.00	90.00	0.00	2	Bus2	1	-43.90	9.42	167.99				
			-18.44	-0.3219	0.00	60.00	0.00				-17.14	25.63					
									5	Bus5	1	8.40	1.88	61.04			
											14.92	4.93					
									3	Bus3	1	-54.50	7.67	283.12			
											-57.78	37.86					

Table 31.3: Input file which defines power directions in UWPFLOW (psatuw.k)

```

C      6 BUS AC TEST SYSTEM
C Generation and Load Directions
C
C This file contains the generation (DPg) and load (Pnl, Qnl, and optional
C Pzl and Qzl) direction, and the maximum P generation (PgMax) needed for
C finding the bifurcation point. Since the IEEE Common Format does not
C allow for the definition of PgMax, this value is ignored in this file
C by making it equal to 0.
C
C The file must be read with the -K option whenever one wants to do
C bifurcation studies (-c, -C, -H and -B options).
C The unformatted data is given in the following order:
C
C BusNumber BusName   DPg     Pnl     Qnl     PgMax [ Smax Vmax Vmin Pzl Qzl ]
  1          0        0.20000  0.00000  0.00000  0       0    1.10000  0.90000
  2          0        0.25000  0.00000  0.00000  0       0    1.10000  0.90000
  3          0        0.20000  0.00000  0.00000  0       0    1.10000  0.90000
  4          0        0.00000  0.25000  0.16665  0       0    1.10000  0.90000
  5          0        0.00000  0.10000  0.07000  0       0    1.10000  0.90000
  6          0        0.00000  0.20000  0.06667  0       0    1.10000  0.90000

```

Table 31.4: UWPFLOW output file with CPF results (psatuw.cpf)

L.F.	V6	V5	V4	V3	V2	V1
0.0000	.99121	.96854	.98592	1.0500	1.0500	1.0500
2.9005	.96112	.92851	.92593	1.0500	1.0500	1.0500
3.2382	.95736	.92347	.91824	1.0500	1.0500	1.0500
3.4006	.95554	.92101	.91448	1.0500	1.0500	1.0500
3.4803	.95464	.91979	.91262	1.0500	1.0500	1.0500
3.5595	.95374	.91858	.91076	1.0500	1.0500	1.0500
3.5595	.95374	.91858	.91076	1.0500	1.0500	1.0500
4.1562	.94322	.90507	.88943	1.0500	1.0410	1.0500
4.4277	.93824	.89864	.87920	1.0500	1.0366	1.0500
4.5574	.93582	.89550	.87418	1.0500	1.0345	1.0500
4.6209	.93462	.89395	.87169	1.0500	1.0334	1.0500
4.6524	.93402	.89318	.87045	1.0500	1.0329	1.0500
4.6680	.93372	.89279	.86983	1.0500	1.0326	1.0500
4.6836	.93342	.89240	.86920	1.0500	1.0324	1.0500
4.6836	.93342	.89240	.86920	1.0500	1.0324	1.0500
4.9291	.91430	.87509	.85116	1.0345	1.0196	1.0500
5.0380	.90517	.86683	.84258	1.0271	1.0136	1.0500
5.0894	.90069	.86279	.83838	1.0235	1.0106	1.0500
5.1145	.89849	.86080	.83631	1.0217	1.0091	1.0500
5.1392	.89628	.85880	.83425	1.0199	1.0076	1.0500
5.1392	.89628	.85880	.83425	1.0199	1.0076	1.0500
5.1392	.89628	.85880	.83425	1.0199	1.0076	1.0500
5.1392	.89628	.85880	.83425	1.0199	1.0076	1.0500
5.1392	.89628	.85880	.83425	1.0199	1.0076	1.0500
5.1392	.89628	.85880	.83425	1.0199	1.0076	1.0500
5.1392	.89628	.85880	.83425	1.0199	1.0076	1.0500
1.5719	.50769	.43186	.47080	.67418	.66722	.72318
.78340	.44492	.34784	.43371	.62063	.62063	.67933
.39080	.42091	.30787	.42720	.59947	.60438	.66391
.19516	.41181	.28897	.42726	.59091	.59843	.65811
.09753	.40808	.27988	.42806	.58719	.59601	.65568
-.0001	.40492	.27106	.42935	.58385	.59396	.65357

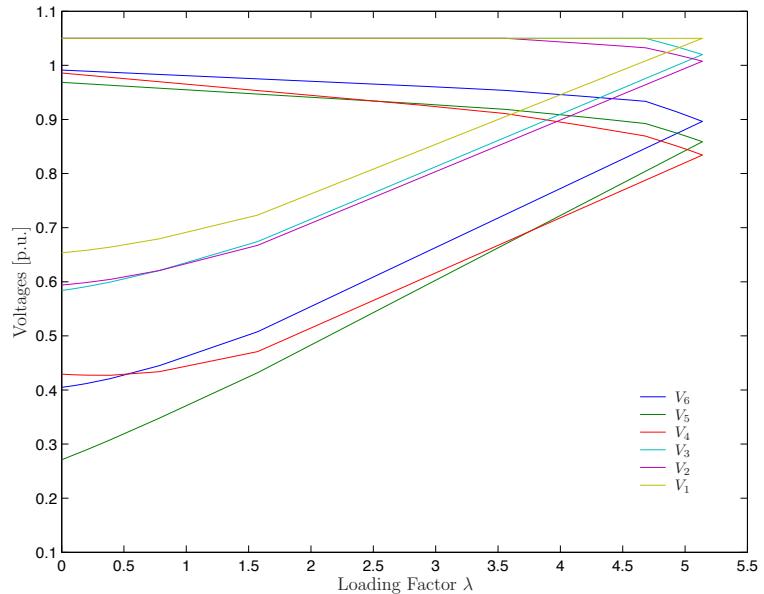
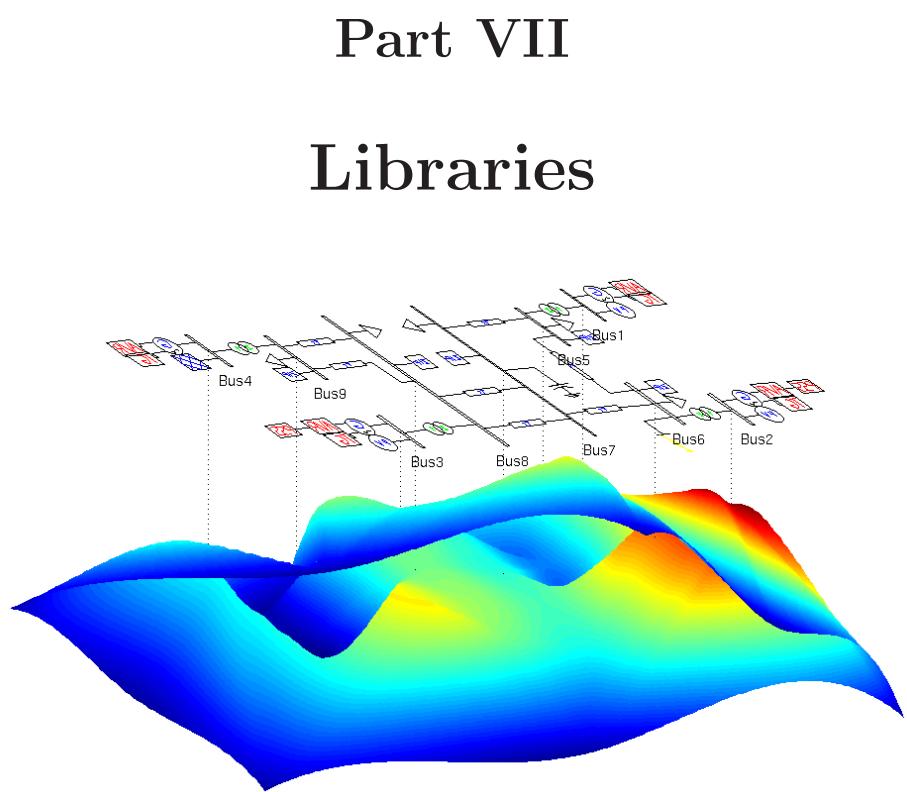


Figure 31.2: UWPFLOW nose curves for the 6-bus test systems. Results are obtained enforcing reactive power limits of generators. Compare these results with the PSAT CPF results depicted in Fig. 6.9.



Chapter 32

Linear Analysis

The linear analysis library computes input and output (algebraic) matrices \mathbf{A} , \mathbf{B} , \mathbf{C} and \mathbf{D} typically used in control theory, as follows:

$$\begin{aligned}\Delta \dot{\mathbf{x}} &= \mathbf{A} \Delta \mathbf{x} + \mathbf{B} \Delta \mathbf{u} \\ \Delta \mathbf{y} &= \mathbf{C} \Delta \mathbf{x} + \mathbf{D} \Delta \mathbf{u}\end{aligned}\tag{32.1}$$

where $\mathbf{x} \in \mathbb{R}^n$ are the state variables, $\mathbf{y} \in \mathbb{R}^m$ are the output variables, and $\mathbf{u} \in \mathbb{R}^p$ the input variables. Observe that in (32.1), \mathbf{y} are the algebraic variables as defined in (5.1). The input variables \mathbf{u} are the reference voltages of AVR, the reference speeds of turbine governors, and the reference voltages of SVC regulators.¹

32.1 Description

Input and output matrices are set up by the function `fm_abcd.m`. This function can be launched at the MATLAB prompt after solving the power flow analysis. Given the system equations:

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}, \mathbf{y}, \mathbf{u}) \\ \mathbf{0} &= \mathbf{g}(\mathbf{x}, \mathbf{y}, \mathbf{u})\end{aligned}\tag{32.2}$$

The linearization leads to the following expressions:

$$\begin{aligned}\Delta \dot{\mathbf{x}} &= f_x \Delta \mathbf{x} + f_y \Delta \mathbf{y} + f_u \Delta \mathbf{u} \\ \mathbf{0} &= g_x \Delta \mathbf{x} + g_y \Delta \mathbf{y} + g_u \Delta \mathbf{u}\end{aligned}\tag{32.3}$$

From the second equation of (32.3), one has:

$$\Delta \mathbf{y} = -g_y^{-1} g_x \Delta \mathbf{x} - g_y^{-1} g_u \Delta \mathbf{u}\tag{32.4}$$

¹Other input variables can be defined adapting device classes and the library functions. Input variables of FACTS other than SVCs are currently under development.

Thus, the input and output matrices are:

$$\mathbf{A} = \mathbf{f}_x - \mathbf{f}_y g_y^{-1} g_x \quad (32.5)$$

$$\mathbf{B} = \mathbf{f}_u - \mathbf{f}_y g_y^{-1} g_u \quad (32.6)$$

$$\mathbf{C} = -g_y^{-1} g_x \quad (32.7)$$

$$\mathbf{D} = -g_y^{-1} g_u \quad (32.8)$$

The function `fm_abcd.m` also computes matrices that link the algebraic variables \mathbf{y} to transmission line currents and active power and reactive power flows. The nonlinear equations that link line flows \mathbf{z} and system variables can be written as follows:

$$\mathbf{z} = \mathbf{h}(\mathbf{y}) \quad (32.9)$$

where it has to be noted that there is no dependence from state and input variables \mathbf{x} and \mathbf{u} . From the linearization of (32.9), one has:

$$\Delta \mathbf{z} = \mathbf{h}_y \Delta \mathbf{y} \quad (32.10)$$

To obtain as output variables $\Delta \mathbf{z}$ instead of $\Delta \mathbf{y}$, one has to use modified $\tilde{\mathbf{C}}$ and $\tilde{\mathbf{D}}$ matrices, as follows:

$$\tilde{\mathbf{C}} = \mathbf{h}_y \mathbf{C} = -\mathbf{h}_y g_y^{-1} g_x \quad (32.11)$$

$$\tilde{\mathbf{D}} = \mathbf{h}_y \mathbf{D} = -\mathbf{h}_y g_y^{-1} g_u \quad (32.12)$$

In case variables \mathbf{z} depend on \mathbf{x} , \mathbf{y} and \mathbf{u} , the expressions of matrices \mathbf{C} and \mathbf{D} is a little more complex, as follows. Let us say that:

$$\mathbf{z} = \mathbf{h}(\mathbf{x}, \mathbf{y}, \mathbf{u}) \quad (32.13)$$

then the linearization leads to:

$$\Delta \mathbf{z} = \mathbf{h}_x \Delta \mathbf{x} + \mathbf{h}_y \Delta \mathbf{y} + \mathbf{h}_u \Delta \mathbf{u} \quad (32.14)$$

and, finally:

$$\hat{\mathbf{C}} = \mathbf{h}_x + \mathbf{h}_y \mathbf{C} = \mathbf{h}_x - \mathbf{h}_y g_y^{-1} g_x \quad (32.15)$$

$$\hat{\mathbf{D}} = \mathbf{h}_u + \mathbf{h}_y \mathbf{D} = \mathbf{h}_u - \mathbf{h}_y g_y^{-1} g_u \quad (32.16)$$

Note that the function `fm_abcd.m` uses Jacobian matrices computed analytically.

Input and output matrices linear analysis library are stored in the structure `LA`, which has the following fields:

1. `a`: state matrix \mathbf{A} .
2. `b_avr`: matrix \mathbf{B} for exciter reference voltages \mathbf{v}_{ref} .
3. `b_tg`: matrix \mathbf{B} for governor reference speeds ω_{ref} .

4. $\mathbf{b_svc}$: matrix \mathbf{B} for SVC reference voltages.
5. $\mathbf{c_y}$: matrix \mathbf{C} for algebraic variables.
6. $\mathbf{d_avr}$: matrix \mathbf{D} for exciter reference voltages \mathbf{v}_{ref} .
7. $\mathbf{d_tg}$: matrix \mathbf{D} for governor reference speeds ω_{ref} .
8. $\mathbf{d_svc}$: matrix \mathbf{D} for SVC reference voltages \mathbf{v}_{ref} .
9. $\mathbf{h_ps}$: matrix \mathbf{H} for active power flows \mathbf{p}_{ij} .
10. $\mathbf{h_qs}$: matrix \mathbf{H} for reactive power flows \mathbf{q}_{ij} .
11. $\mathbf{h_pr}$: matrix \mathbf{H} for active power flows \mathbf{p}_{ji} .
12. $\mathbf{h_qr}$: matrix \mathbf{H} for reactive power flows \mathbf{q}_{ji} .
13. $\mathbf{h_is}$: matrix \mathbf{H} for current flows \mathbf{i}_{ij} .
14. $\mathbf{h_ir}$: matrix \mathbf{H} for current flows \mathbf{i}_{ji} .

32.2 Example

This example illustrates the use of the linear analysis library through the WSCC 9-bus test system described in [111] (see also the Appendix F.3).

The following code assumes that the command line version of PSAT is used and that the data file `d_009_mdl.m` is in the local path. The perturbation consists in a 2% variation of the voltage reference of the AVR connected to generator 2.

```
% set the data file
runpsat('d_009_mdl.m','data')
% run the power flow
runpsat('pf')
% compute in and out variables
fm_abcd;
% initialize the time vector
t = 0:0.01:20;
% set up the a 2% step in the AVR voltage reference
u(1:length(t)) = 0.02;
% set up the initial state variables
x0 = zeros(DAE.n,1);
% store the initial values of bus voltages
% and power flows in transmission lines
v0 = DAE.y(Bus.v);
[ps0,qs0,pr0,qr0] = fm_flows;

% run linear time domain simulation
[dy,dt,dx] = lsim(ss(LA.a,LA.b_avr(:,2), ...
```

```

        LA.c_y,LA.d_avr(:,2)),u,t,x0);
% note the use of matrices H for getting power flows
[dp,dt,dx] = lsim(ss(LA.a,LA.b_avr(:,2),LA.h_ps*LA.c_y, ...
LA.h_ps*LA.d_avr(:,2)),u,t,x0);
[dq,dt,dx] = lsim(ss(LA.a,LA.b_avr(:,2),LA.h_qs*LA.c_y, ...
LA.h_qs*LA.d_avr(:,2)),u,t,x0);

% add initial values to the simulation results
vmat = ones(length(dt),Bus.n);
v = dy(:,Bus.v) + vmat*diag(v0);
qmat = ones(length(dt),Line.n);
q = dq + qmat*diag(qs0);
pmat = ones(length(dt),Line.n);
p = dp + pmat*diag(ps0);

% plot some results
figure
plot(t,v(:,[6 7]))
figure
plot(t,q(:,6))
figure
plot(t,p(:,7))

```

The consistency of the results can be checked by running a time domain simulation with the same perturbation. For example, Figures 32.1 and 32.2 compare voltages and reactive powers obtained using the linear analysis and the time domain simulation.

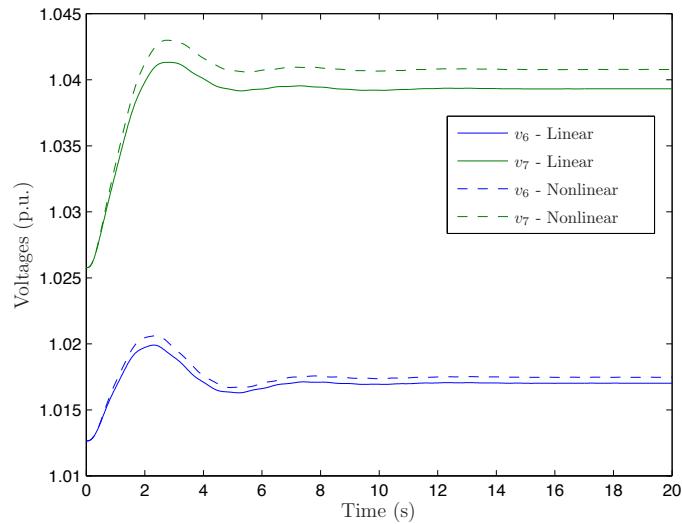


Figure 32.1: Linear analysis of the WSCC system: voltages at buses 6 and 7.

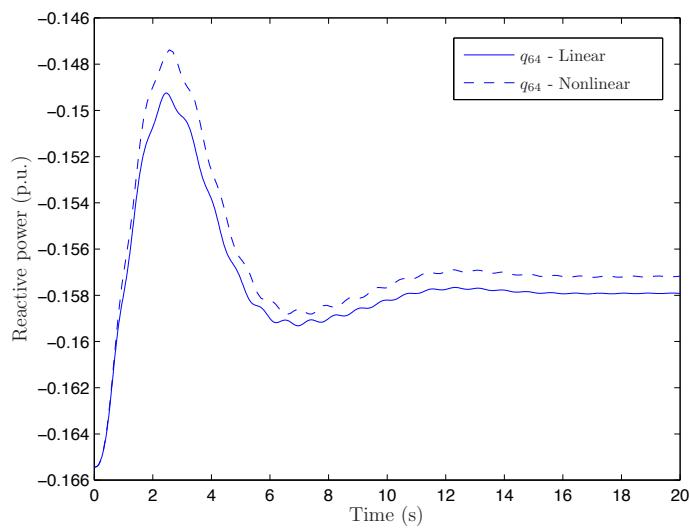


Figure 32.2: Linear analysis of the WSCC system: reactive power of line 6-4.

Chapter 33

Numerical Differentiation

PSAT provides the script `numjacs.m` for computing numerical Jacobian matrices f_x , f_y , g_x , and g_y . The script creates the matrices F_x , F_y , G_x , and G_y that can be accessed from the MATLAB prompt. Table 33.1 shows the comparison of the eigenvalues of the WSCC 9-bus system computing using numerical and analytical Jacobian matrices.

The script `numjacs.m` can be used to double-check the correctness of analytical Jacobian matrices that are normally used in PSAT. The check can be done automatically using the script `checkjac.m`, which computes the maximum absolute and relative errors between the analytical and the numerical Jacobian matrices. For example, for the WSCC test system, the output of `checkjac.m` is as follows:

```
% maximum absolute errors
Fx abs err = 3.9112e-06
Fy abs err = 3.9112e-06
Gx abs err = 2.1476e-07
Gy abs err = 1.7183e-06

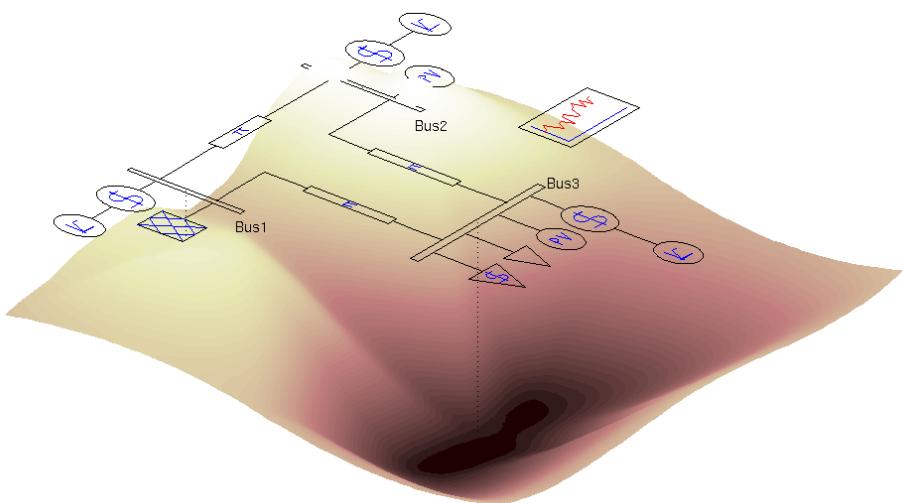
% maximum relative errors
Fx rel err = 2.8743e-07
Fy rel err = 2.5827e-07
Gx rel err = 2.5743e-06
Gy rel err = 2.2366e-06
```

Table 33.1: State matrix eigenvalues for the WSCC 9-bus test system

Numerical State Matrix	Analytical State Matrix
-999.999951348	-999.999954762
-999.999973362	-999.999976528
-999.999979448	-999.999983351
$-0.70749883906 \pm 11.60651343150i$	$-0.70749888190 \pm 11.60651347991i$
$-0.18645447282 \pm 7.63240176503i$	$-0.18645448595 \pm 7.63240191527i$
$-5.48380341424 \pm 7.94648257921i$	$-5.48380340701 \pm 7.94648256512i$
$-5.21800507874 \pm 7.81342706646i$	$-5.21800510927 \pm 7.81342704488i$
$-5.32109571324 \pm 7.91898907306i$	$-5.32109570704 \pm 7.91898904706i$
-5.19713348603	-5.19713340998
-3.40392005505	-3.40391968965
$-0.44279102644 \pm 1.21198161044i$	$-0.44279104934 \pm 1.21198178773i$
$-0.43828641145 \pm 0.74014689900i$	$-0.43828642958 \pm 0.74014692649i$
$-0.42482821305 \pm 0.49684500482i$	$-0.42482821493 \pm 0.49684501086i$
$-0.00000006519 \pm 0.00006233819i$	-0.00000038885
-3.22580645161	-3.22580645161

Part VIII

Appendices



Appendix A

Global Structures & Classes

This appendix lists all global structures used in PSAT and provides a detailed description of their fields. If the structures and the associated fields are described elsewhere, only the section number is reported.

A.1 General Settings

General settings and parameters for power flow computations and time domain simulations are stored in the structure `Settings`, whose fields are as follows:

absvalues Use of absolute/per unit values when writing the report file of the current case solution.

`on` Use absolute values.

`off` Use per unit values (default).

beep Beep control.

`0` Disabled (default).

`1` Enabled.

checkdelta Stop time domain simulation if the maximum relative difference between two generator rotor angles is greater than `Settings.deltadelta`.

`0` Disabled (default).

`1` Enabled.

chunk Initial dimension of output arrays.

coi Use COI during time domain simulations.

`0` Disabled (default).

`1` Enabled.

color Default GUI colors.

conv System base conversion and checks.

0 Disabled.

1 Enabled (default).

date Release date of the current PSAT version.

deltadelta Maximum allowed relative rotor angle difference used for checking the transient stability during time domain simulations [deg]. Default value is 180°.

deltat Time step for time domain integrations [s].

deltatmax Maximum time step [s].

deltatmin Minimum time step [s].

distrsw Set distributed slack bus model.

0 Disabled (default).

1 Enabled.

donotask Apply current options and user preferences without asking for confirmation. This option can be “tricky” unless the user really knows what he is doing. This option has no effect for command line usage where the preferences of the structure **clpsat** take the precedence.

0 Disabled (default).

1 Enabled.

dynmit Maximum number of iteration for dynamic analyses.

dyntol Error tolerance for dynamic analyses.

error Maximum error mismatch of power flow analysis.

export Extension of report file .

txt Plain text (default).

xls Excel format.

hml HTML format.

tex L^AT_EX format.

fixt Set fixed time step.

0 Disabled (default).

1 Enabled.

forcepq Force PQ loads to constant powers regardless the bus voltage level or other options (e.g. **pq2z**). This option is silently and implicitly enforced for CPF and OPF analyses.

- 0 Disabled (default).
- 1 Enabled.

format Data file format number (default 1).

freq System frequency rating in Hz (default 50 Hz).

hostver MATLAB or GNU OCTAVE version of the current session.

init Power flow status.

- 1 Power flow not converged.
- 0 Power flow not solved yet.
- 1 Power flow completed.
- 2 Time domain simulation completed.

iter Number of iterations of the last power flow computation (default 20).

lfmit Maximum number of iteration for static analyses (default 20).

lftime Elapsed time for power flow computations.

lftol Error tolerance for static analyses (default 10^{-5}).

local Defines the folder where to write the function **fm_call.m**. Use 0 only if the main PSAT folder is writable.

- 0 Use folder **Path.psat**.
- 1 Use folder **Path.local** (default).

locksnap Skip initialization and updating of the **Snapshot** structure after power flow analysis. This may lead to inconsistencies if solving OPF, LIB, ATC and $N - 1$ contingency analyses.

- 0 Initialize snapshots (default).
- 1 Skip initialization.

matlab True if the current PSAT session is running on MATLAB .

- 0 PSAT is not running on MATLAB .
- 1 PSAT is running on MATLAB .

maxvar Maximum number of variables displayed in the static report GUI (default value is 1500).

maxsimin Maximum number of input ports of a SIMULINK bus block (default value is 15).

maxsimout Maximum number of output ports of a SIMULINK bus block (default value is 15).

method Integration method.

- 1 Forward Euler's method.
- 2 Trapezoidal method (default).

multipvswitch Allows more than one PV generator can be switched to PQ if a reactive power is reached.

- 0 Disabled (default).
- 1 Enabled.

mv Model version of the currently loaded SIMULINK model.

mva System power rating in MVA (default 100 MVA).

noarrows Defines if the arrows have to be removed when exporting PSAT-SIMULINK model to eps files.

- 0 Leaves arrows there.
- 1 Removes arrows (default).

nseries Number of series components defined in the current system. It is the sum of the number of lines, load tap changers, phase shifters and HVDC lines.

octave True if the current PSAT session is running on GNU OCTAVE .

- 0 PSAT is not running on GNU OCTAVE .
- 1 PSAT is running on GNU OCTAVE .

ok Output of the **fm_choice** dialog box.

- 0 Yes.
- 1 No.

pfsolver Select power flow solver.

- 1 Newton-Raphson's method (default).
- 2 XB variation of fast decoupled power flow.
- 3 BX variation of fast decoupled power flow.
- 4 Runge-Kutta-based continuous Newton's method.
- 5 Iwamoto's method.
- 6 Simple robust method.

platform Computer architecture or platform: UNIX, MAC, or PC.

plot Plot during time domain simulations.

- 0 Disabled (default).
- 1 Enabled.

plottype Select variable to be plot during time domain simulations.

- 1 State variables.
- 2 Bus voltage magnitudes.
- 3 Bus voltage phases.
- 4 Real powers injected at buses.
- 5 Reactive powers injected at buses.

pq2z Convert PQ load to constant impedances. This option is applied **only** for time domain simulations. The option **forcepq** overwrite this option.

- 0 Disabled (default).
- 1 Enabled.

pv2pq Generator reactive power limit control during power flow analysis.

- 0 Disabled (default).
- 1 Enabled.

pv2pqniter Integer that represents the number of iterations that the power flow solver has to wait before switching PV generators to PQ ones if a reactive power limit is reached. The default value is 0.

report Style of the power flow report.

- 0 Bus report and line report are separated (default).
- 1 The line flows are embedded in the bus report.

resetangles After clearing a fault, reset bus angles to the pre-fault values. Disabling this behavior can in some cases improve the convergence of the post-fault point during time domain simulations.

- 0 Disabled.
- 1 Enabled (default).

show Display iteration status and messages.

- 0 Disabled.
- 1 Enabled.

showlf Display report GUI after power flow solution.

- 0 Disabled (default).
- 1 Enabled.

simtd Display and update voltages in SIMULINK models during time domain simulations.

- 0 Do not display/update (default).
- 1 Display/update.

static Discard dynamic component data.

- 0 Disabled (default).
- 1 Enabled.

status Display convergence error of the current iteration on the main window.

- 0 Disabled.
- 1 Enabled.

switch2nr In power flow analysis, switch a robust power flow method to the standard NR method if the tolerance error is smaller than 10^{-2} .

- 0 Disabled (default).
- 1 Enabled.

t0 Initial simulation time [s].

tf Final simulation time [s].

tstep Fixed time step value [s].

tviewer Current text viewer.

usedegree Use degrees when plotting output angles of time domain simulation.

- 0 Disabled (default).
- 1 Enabled.

usehertz Use Hertz when plotting output rotor speeds of time domain simulation.

- 0 Disabled (default).
- 1 Enabled.

userelspeed Use relative values when plotting output rotor speeds of time domain simulation.

0 Disabled (default).
1 Enabled.

version Current PSAT version.

vs Compute eigenvalues of the state matrix during time domain simulations (deprecated).

violations Enforce limit violation checks when writing the report file of the current case solution.

on Disabled (default).
off Enabled.

xlabel Label for plotting variables.

zoom Zoom plotting variables.

0 Disabled (default).
1 Enabled.

A.2 Other Settings

Comp Device general settings.

funct	Cell array of all device functions.
n	Total number of installed devices.
number	Cell array of all device .n fields.
prop	Device properties.

Fig Handles of the GUI windows. The handle value is 0 if the associated window is not open. The handle names are as follows:

about	PSAT information.
advanced	Advanced settings.
author	Author's pic.
clock	Analogical watch window.
cpf	Continuation power flow settings.
dir	File browser and data format conversion.
eigen	Small signal stability analysis.
gams	PSAT-GAMS interface.
hist	Command history.
laprint	LATEX settings (deprecated).
lib	Limit-induced bifurcation settings.
license	Displays the program license.
line	GUI for editing the plotted line properties.
main	PSAT main window.

matrix	GUI for Jacobian matrix visualization.
opf	Optimal power flow settings.
plot	GUI for plotting variables.
plotsel	GUI for selecting output variables for TDs.
pmu	PMU placement.
simset	GUI for setting SIMULINK model properties.
setting	General settings.
snap	GUI for setting snapshots.
snb	Direct method for SNB.
stat	Power flow report GUI.
theme	Theme browser.
threed	3D system visualization.
tviewer	GUI for selecting the text viewer.
uwpflow	GUI for the PSAT-UWPFLOW. interface
warranty	GUI that displays the warranty conditions.

File Data and disturbance file names, as follows:

data	Current data file name.
modify	Data modification time.
pert	Current disturbance file name.
temp	Temporary file name (internal use).

Hd1 Handles of the most used graphic objects.

axes	PSAT logo axis in the main window.
bar	Axis for the progress bar in the main window.
frame	Frame of message text in the main window.
hist	Command history list-box in the command history GUI.
pert	Handle of perturbation function.
status	Axis for convergence status in the main window.
text	Message static text in the main window.

History Command history text and settings.

Backgrounder	Background color of the command history GUI.
FontName	Name of the font of the command history GUI.
FontSize	Size of the font of the command history GUI.
FontAngle	Angle of the font of the command history GUI.
FontWeight	Weight of the font of the command history GUI.
ForegroundColor	Foreground color of the command history GUI.
index	Number of the last row where string was found.
Max	Maximum number of rows of the text cell array.
string	String for text search within the command history.
text	Cell array of the last $n = \text{Max}$ commands.
workspace	Enable displaying messages on the MATLAB workspace.

Path Path strings of the most commonly used folders, as follows:

build	Absolute path of the secondary folder build .
data	Current data file path.
filters	Absolute path of the secondary folder filters .
images	Absolute path of the secondary folder images .
local	Current workspace path.
pert	Current disturbance file path.
psat	PSAT path.
themes	Absolute path of the secondary folder themes .
temp	Temporary path name (internal use).

Source Cell arrays containing the current data file and the current disturbance file.

This structure is used for saving outputs on disk. The fields are as follows:

data	Data file cell array.
description	Case description (<i>not used</i>).
pert	Disturbance file cell array.

Snapshot snapshot data.

Fx	Jacobian matrix f_x .
Fy	Jacobian matrix f_y .
Gx	Jacobian matrix g_x .
Gy	Jacobian matrix g_y .
it	Flag for internal usage.
name	Cell array of snapshot names.
Ploss	Total real losses of the current power flow solution.
Pg	Vector of generator real powers injected at buses.
P1	Vector of load real powers absorbed from buses.
Qg	Vector of generator reactive powers injected at buses.
Q1	Vector of load reactive powers absorbed from buses.
Qloss	Total reactive losses of the current power flow solution.
time	Array of times associated to the defined snapshots.
x	Vector of state variables.
y	Vector of algebraic variables.
Ybus	Network admittance matrix.

Theme Properties and settings for the appearance of the GUIs.

color01	Background color 1.
color02	Background color 2.
color03	List box color 1 (used also for special buttons).
color04	List box color 2.
color05	Text color 1.
color06	Text color 2.

<code>color07</code>	Text color 3.
<code>color08</code>	Progress bar color.
<code>color09</code>	Text color for special buttons.
<code>color10</code>	Text color for special list boxes.
<code>color11</code>	Axis color.
<code>font01</code>	Font name for buttons and static texts.
<code>font02</code>	Font name for edit texts.
<code>font03</code>	Font name for command line in the main window.
<code>font04</code>	Font name for command history in the main window.
<code>hdl</code>	Handles of graphical objects in the theme manager GUI.

A.3 System Properties and Settings

`CPF` Continuation power flow settings.

`areaannualgrowth` Area loading direction rates.

`flow` Select transmission line flow.

- 1 Currents i_{ij} .
- 2 Active powers p_{ij} .
- 3 Apparent powers s_{ij} .

`hopf` Check for change of sign of pair of complex conjugate eigenvalues (Hopf bifurcation points).

- 0 Disabled (default).
- 1 Enabled.

`ilim` Check transmission line flow limits.

- 0 Disabled.
- 1 Enabled.

`init` Solution status of continuation power flow.

- 0 To be solved yet.
- 1 Solved continuation power flow.
- 2 Solved ATC analysis.
- 3 Solved $N - 1$ contingency analysis.
- 4 Solved continuation OPF (PSAT-GAMS interface).

`kg` Distributed slack bus variable.

`lambda` Loading parameter.

`linit` Initial value of the loading parameter λ .

`method` Method for corrector step.

- 1 Perpendicular intersection.
- 2 Local parametrization.

negload Include negative active power loads in CPF analysis.

0 Disabled (default).
1 Enabled.

nump Maximum number of points to be computed.

pmax Maximum power flow limits. This field is filled up by the function **fm_n1cont** as a result of the $N - 1$ contingency criterion.

onlynegload Use only negative active power loads in CPF analysis.

0 Disabled (default).
1 Enabled.

onlypqgen Use only PQ generators in CPF analysis.

0 Disabled (default).
1 Enabled.

qlim Check generator reactive power limits.

0 Disabled.
1 Enabled.

regionannualgrowth Region loading direction rates.

sbus Slack bus model.

0 Distributed slack bus.
1 Single slack bus.

show Show iteration status on main window.

0 Disabled.
1 Enabled.

step Step size control.

stepcut Step size control.

0 Disabled.
1 Enabled (default).

tolc Corrector step tolerance.

tolf Error tolerance for transmission line flows.

tolv Error tolerance for bus voltages.

type Select end criterion for the the continuation power flow. If “complete nose curve” is set, the routine stops either if the maximum number of points is reached or if $\lambda = 0$.

1 Complete nose curve.
2 Stop when a bifurcation is encountered.
3 Stop when the first enforced limit is encountered.

vlim Check voltage limits.

- 0 Disabled.
- 1 Enabled.

DAE Differential and algebraic equations, functions and Jacobians matrices. Fields are as follows:

Ac	Complete DAE Jacobian matrix.
f	Differential equations \mathbf{f} .
Fk	Jacobian matrix of differential equations \mathbf{f}_{k_G} .
Fl	Jacobian matrix of differential equations \mathbf{f}_λ .
Fx	Jacobian matrix of differential equations \mathbf{f}_x .
Fy	Jacobian matrix of differential equations \mathbf{f}_y .
g	Algebraic equations \mathbf{g} .
Gk	Jacobian matrix of algebraic equations \mathbf{g}_{k_G} .
Gl	Jacobian matrix of algebraic equations \mathbf{g}_λ .
Gx	Jacobian matrix of algebraic equations \mathbf{g}_x .
Gy	Jacobian matrix of algebraic equations \mathbf{g}_y .
kg	Variable for distributing losses among generators.
m	Number of algebraic variables m .
n	Number of state variables n .
npf	Dynamic order during power flow n_{PF} .
tn	Vector of DAE for time domain simulations.
t	Current simulation time (-1 for static analysis).
x	State variables \mathbf{x} .
y	Algebraic variables \mathbf{y} .

LIB Settings for limit-induced bifurcation (direct method).

bus	Generation and load direction buses.
dndl	Sensitivity coefficients \mathbf{p}_λ .
lambda	Loading parameter λ .
selbus	Bus number where applying the limit.
slack	Enable distributed slack bus.

- 0 Single slack bus.
- 1 Distributed slack bus.

type LIB type.

- 1 Maximum voltage v^{\max} .
- 2 Minimum voltage v^{\min} .
- 3 Maximum reactive power q^{\max} .
- 4 Minimum reactive power q^{\min} .

OPF Optimal power flow settings and outputs.

atc Maximum loading condition for the current OPF solution.

basepg Include base case generation powers.

- 0 Disabled.
- 1 Enabled.

basepl Include base case load powers.

- 0 Disabled.
- 1 Enabled.

conv OPF method convergence status.

- 0 OPF routine did not converge.
- 1 OPF routine converged.

deltat Time step in minutes of the daily forecast (*not used*).

dF Equality constraint mismatch.

dG Objective function mismatch.

dy Algebraic variable mismatch.

enflow Enforce flow limit inequalities.

- 0 Disabled.
- 1 Enabled.

enreac Enforce generator reactive power inequalities.

- 0 Disabled.
- 1 Enabled.

envolt Enforce voltage limit inequalities.

- 0 Disabled.
- 1 Enabled.

eps1 Error tolerance of the power flow equations.

eps2 Error tolerance of the objective function.

eps_mu Error tolerance of the barrier parameter μ_s .

flatstart Set initial guess of system variables.

- 1 Flat start ($\mathbf{v} = \mathbf{1}$ and $\boldsymbol{\theta} = \mathbf{0}$).
- 2 Actual power flow solution.

flow Type of flows used for the flow constraints in the transmission lines.

- 1 Currents i_{ij} .
- 2 Active power flows p_{ij} .
- 3 Apparent power flows s_{ij} .

fun Current selected OPF function.

gamma Safety factor γ .

gpc Active power injections for the critical loading condition.

gqc Reactive power injections for the critical loading condition.

guess Vector of values for initializing the OPF routine.

init OPF solution status.

- 0 Not solved yet.
- 1 Standard OPF has been solved.
- 2 Multi-objective OPF has been solved.
- 3 Pareto's set OPF has been solved.

iter Number of iterations of the OPF method.

line Number of the line to be deleted for $N - 1$ contingency evaluations in the maximum loading condition system.

lmax Maximum value of the loading parameter λ .

lmin Minimum value of the loading parameter λ .

lmin_s String containing the minimum value of the loading parameter λ .

LMP Locational Marginal Prices of the current solution.

method Method used for computing the variable directions and increments.

- 1 Newton's directions.
- 2 Merhotra's Predictor/Corrector.

ms Barrier parameter μ_s .

NCP Nodal Congestion Prices of the current solution.

obj Value of the objective function.

omega Weighting factor ω (can be a vector).

omega_s String containing the weighting factor ω .

report Cell array of the OPF solution.

show Display the convergence error of the OPF routine.

- 0 Disabled.
- 1 Enabled.

sigma Centering parameter σ .

tiebreak Tiebreak term in the objective function.

- 0 Disabled.
- 1 Enabled.

type Type of OPF problem to be solved.

- 1 Single OPF (if ω is a vector, the first value is used).
- 2 Pareto's set (one solution for each value of the vector ω).
- 3 Daily forecast (*not implemented yet*).
- 4 ATC by CPF (*development status*) .
- 5 ATC by sensitivity analysis (*development status*).

- vmax** Maximum voltage limit for zero-injection buses, i.e. buses at which there is no generator or load connected (default 1.2 p.u.).
- vmin** Minimum voltage limit for zero-injection buses, i.e. buses at which there is no generator or load connected (default 0.8 p.u.).
- w** Actual value of the weighting factor ω .
- wp** Dummy variable containing the value of the weighting factor ω .
- PMU** Settings for PMU placement algorithms.
- angle** Cell array of estimated angles.
 - location** Cell array of PMU placement.
 - measc** Number of measured currents.
 - measv** Number of measured voltages.
 - method** Method type.
 - 1 Depth first.
 - 2 Graphic theoretic procedure.
 - 3 Annealing-bisecting search method.
 - 4 Recursive security N algorithm.
 - 5 Single-shot security N algorithm.
 - 6 Recursive security $N-1$ algorithm.
 - 7 Single-shot security $N-1$ algorithm.
 - noobs** Current number of non-observable buses.
 - number** Current number of PMU.
 - pseudo** Number of pseudo-measured currents.
 - report** Structure containing the full PMU placement report.
 - voltage** Cell array of estimated voltages.
- SNB** Settings for saddle-node bifurcation analysis (direct method).
- bus** Generation and load direction buses.
 - dndl** Sensitivity coefficients p_λ .
 - lambda** Loading parameter λ .
 - slack** Enable distributed slack bus.
 - 0 Single slack bus.
 - 1 Distributed slack bus.
- SSSA** Settings for small signal stability analysis.
- eigs** Vector of eigenvalues mu .
 - map** Map type.

1 S -map.
 2 Participation factor map.
 3 Z -map.

matrix Matrix type.

- 1 Reduced dynamic power flow Jacobian $\tilde{\mathbf{J}}_{LFD}$.
- 2 Reduced complete power flow Jacobian $\tilde{\mathbf{J}}_{LFV}$.
- 3 Reduced standard power flow Jacobian $\tilde{\mathbf{J}}_{LF}$.
- 4 State matrix \mathbf{A}_S

method Eigenvalue computation method.

- 1 All eigenvalues.
- 2 Largest magnitude.
- 3 Smallest magnitude.
- 4 Largest real part.
- 5 Smallest real part.
- 6 Largest imaginary part.
- 7 Smallest imaginary part.

neig Number of eigenvalues to be computed (applies only if **method** ≠ 1).

pf Matrix of participation factors.

report Structure containing the small signal stability analysis report.

A.4 Outputs and Variable Names

Varname System variable **TEX** and plain names. Formatted **TEX** names are used for creating legends in the plotting variable GUI. Fields are as follows:

areas	Indexes of the selected areas.
compx	Names of components with state variables.
compy	Names of components with algebraic variables.
custom	True if custom selection of plot variables.
idx	Indexes of selected plot variables.
Iij	True if selecting all current power flows.
fixed	True if fixed selection of plot variables.
fnamex	Formatted names of all state variables.
fnamey	Formatted names of all algebraic variables.
fvars	Formatted names of output variables.
nvars	Total number of output variables.
P	True if selecting all active power bus injections.
Pij	True if selecting all active power flows.
pos	Vector of positions of plotted variables.
Q	True if selecting all reactive power bus injections.
Qij	True if selecting all reactive power flows.
regions	Indexes of the selected regions.

Sij	True if selecting all apparent power flows.
unamex	Unformatted names of all state variables.
unamey	Names of all algebraic variables.
uvars	Unformatted names of output variables.
x	True if selecting all state variables.
y	True if selecting all algebraic voltages.

Varout Output of time domain simulations. Fields are as follows:

alpha	Transparency level for 3D visualization.
caxis	Set voltage limits for 3D visualization.
idx	Indexes of currently stored output variables.
hdl	Handles of the network scheme in the 3D plots.
movie	3D movie of the simulation.
surf	Handle of the surface plot object.
t	Time vector.
vars	Output variables.
xb	<i>x</i> -axis grid data for 3D visualization.
yb	<i>y</i> -axis grid data for 3D visualization.
zlevel	High of network scheme in 3D visualization.

A.5 Models

Power Flow Data

Areas	Interchange area.	Table 12.11
Bus	Bus numbers and voltage ratings.	Table 12.1
Line	Transmission line and transformer.	Tables 12.2-12.4
Lines	Alternative transmission line.	Table 12.3
PV	PV generator.	Table 12.7
PQ	Constant power load.	Table 12.8
PQgen	Constant power generator.	Table 12.9
Regions	Region.	Table 12.11
Shunt	Shunt admittance.	Table 12.10
SW	Slack bus.	Table 12.6
Twt	Three-winding transformer.	Table 12.5

CPF and OPF Data

Demand	Power demand.	Table 13.4
Rmpg	Generator ramp.	Table 13.3
Rmpl	Power demand ramp.	Table 13.6
Rsrv	Generator power reserve.	Table 13.2
Supply	Power supply.	Table 13.1
Vltn	Violation parameters.	<i>not used...</i>
YpdP	Demand profile.	Table 13.5

Faults & Breakers

Breaker	Transmission line breaker.	Table 14.2
Fault	Transmission line fault.	Table 14.1

Measurements

Busfreq	Bus frequency measurement.	Table 15.1
Pmu	Phasor measurement units.	Table 15.2

Loads

Exload	Exponential recovery load.	Table 16.6
F1	Frequency dependent load.	Table 16.3
Jimma	Jimma's load.	Table 16.8
Mixload	Mixed load.	Table 16.9
Mn	Voltage dependent load.	Table 16.1
P1	ZIP (polynomial) load.	Table 16.2
Tap	Voltage dependent load with tap changer.	Table 16.5
Thload	Thermodynamically controlled load.	Table 16.7

Machines

COI	Center of inertia.	Table 17.1
Ind	Induction machine.	Table 17.3
Syn	Synchronous machine.	Table 17.1

Controls

CAC	Central Area Controller.	Table 18.8
Cluster	Cluster Controller.	Table 18.9
Exc	Automatic Voltage Regulator.	Tables 18.3-18.5
Oxl	Over-excitation Limiter.	Table 18.7
Pod	Power Oscillation Damper.	Table 18.10
Pss	Power System Stabilizer.	Table 18.6
Tg	Turbine Governor.	Table 18.1-18.2

Regulating Transformers

Ltc	Load tap changer.	Table 19.1
Phs	Phase shifting transformer.	Table 19.2

FACTS

Hvdc	High Voltage DC transmission system.	Table 20.7
Sssc	Static Synchronous Source Series Compensator.	Table 20.5
Statcom	Static Var Compensator.	Table 20.4
Svc	Static Var Compensator.	Tables 20.1-20.2

Tcsc	Thyristor Controlled Series Capacitor.	Table 20.3
Upfc	Unified Power Flow Controller.	Table 20.6

Wind Turbines

Cswt	Constant speed wind turbine.	Table 21.4
Ddsg	Direct drive synchronous generator.	Table 21.6
Dfig	Doubly fed induction generator.	Table 21.5
Wind	Wind models.	Table 21.1

Other Models

Mass	Synchronous machine dynamic shaft.	Table 22.1
Sofc	Solid Oxide Fuel Cell.	Table 22.3
SSR	Sub-synchronous resonance model.	Table 22.2

A.6 Command Line Usage

clpsat Structure for command line usage of PSAT.

init Command line initialization status.

- 0 PSAT is running with the standard GUIs.
- 1 Command line PSAT is active (default).

msg Status of PSAT messages.

- 0 No message.
- 1 Messages will be displayed in the current output (default).

pq2z If true, force to switch PQ loads to constant impedances before running time domain simulations.

- 0 False.
- 1 True (default).

readfile If true, force to read data file before running power flow.

- 0 False.
- 1 True (default).

refresh If true, force to repeat power flow before running further analysis independently on the power flow status.

- 0 False.
- 1 True (default).

refreshsim If true, force to reload SIMULINK model before running power flow independently on the SIMULINK model status.

- 0 False (default).
- 1 True.

showopf If true, force to display OPF result on the standard output running power flow.

- 0 False (default).
- 1 true.

viewrep If true, force to visualize report files when created.

- 0 False (default).
- 1 True.

A.7 Interfaces

GAMS Parameters and settings for the PSAT-GAMS interface:

basepg Use base generator powers in OPF .

- 0 Disabled.
- 1 Enabled (default).

basepl Use base load powers in OPF .

- 0 Disabled.
- 1 Enabled (default).

flatstart Set initial guess of system variables.

- 1 Use flat start as initial guess ($V = 1$ and $\theta = 0$).
- 2 Use current power flow solution as initial guess.

flow Flow type in transmission lines

- 0 None.
- 1 Currents.
- 2 Active powers.
- 3 Apparent powers.

ldir Command line options for GAMS calls.

libinclude Use command line options.

- 0 Disabled.
- 1 Enabled.

line Number of line to be taken out in $N - 1$ contingency analysis.

lmax Maximum value of λ (float).

lmin Minimum value of λ (float).

lmin_s Minimum value of λ (string).

loaddir Use load direction when solving maximum loading condition OPF.

- 0 Disabled.
- 1 Enabled.

method Select OPF method.

- 1 Simple auction.
- 2 Market clearing mechanism.
- 3 Standard OPF.
- 4 VSC-OPF.
- 5 Maximum loading condition.
- 6 Continuation OPF.

omega Weighting factor ω values (float).

omega_s Weighting factor ω values (string).

show Display results and logs.

- 0 Disabled.
- 1 Enabled.

type Solution type.

- 1 Single period auction.
- 2 Multi-period auction.
- 3 Pareto's set auction.
- 4 Unit commitment auction.

UWPFLOW Parameters, option and settings for the PSAT-UWPFLOW interface.

command Generation and load direction buses.

file Name of output files (default **psatuw**).

method Loading parameter λ value.

- 1 Power flow.
- 2 Continuation power flow.
- 3 Direct method.
- 4 Parametrized continuation method.

opt List of UWPFLOW options. Refer to UWPFLOW documentation for details [22].

status Generation and load direction buses.

A.8 Classes

ARclass	Class for Area and Regions devices.
AVclass	Class for Exc devices.
BFclass	Class for Busfreq devices.
BKclass	Class for Breaker devices.
BUclass	Class for Bus devices.
CCclass	Class for Cac devices.
CIclass	Class for COI devices.
CLclass	Class for Cluster devices.

CSclass	Class for Cswt devices.
DDclass	Class for Ddsg devices.
DFclass	Class for Dfig devices.
DMclass	Class for Demand devices.
DSclass	Class for Mass devices.
ELclass	Class for Exload devices.
FCclass	Class for Sofc devices.
FLclass	Class for F1 devices.
FTclass	Class for Fault devices.
HVclass	Class for Hvdc devices.
IMclass	Class for Ind devices.
JIclass	Class for Jimma devices.
LNclass	Class for Line devices.
LSclass	Class for Lines devices.
LTclass	Class for Ltc devices.
MNclass	Class for Mn devices.
MXclass	Class for Mixload devices.
OXclass	Class for Oxl devices.
PHclass	Class for Phs devices.
PLclass	Class for P1 devices.
PMclass	Class for Pmu devices.
POclass	Class for Pod devices.
PQclass	Class for PQ and PQgen devices.
PSclass	Class for Pss devices.
PVclass	Class for PV devices.
RGclass	Class for Rmpg devices.
RLclass	Class for Rmpl devices.
RSclass	Class for Rsrv devices.
SHclass	Class for Shunt devices.
SRclass	Class for Ssr devices.
SSclass	Class for Sssc devices.
STclass	Class for Statcom devices.
SUclass	Class for Supply devices.
SVclass	Class for Svc devices.
SWclass	Class for SW devices.
SYclass	Class for Syn devices.
TCclass	Class for Tcsc devices.
TGclass	Class for Tg devices.
THclass	Class for Thload devices.
TPclass	Class for Tap devices.
TWclass	Class for Twt devices.
UPclass	Class for Upfc devices.
VLclass	Class for Vltn devices.
WNclass	Class for Wind devices.
YPclass	Class for Ypdp devices.

Appendix B

Matlab Functions

This appendix lists the MATLAB scripts and functions of the PSAT folder. The list is also available on-line (`Contents.m`) by typing

```
>> help psat
```

General Functions and GUIs

<code>fm_main</code>	Main GUI.
<code>fm_set</code>	General settings and utilities.
<code>fm_var</code>	Definition of global variables.
<code>psat</code>	Main script that launches PSAT.

Power Flow

<code>fm_base</code>	Report of component quantities on system bases.
<code>fm_dynidx</code>	Indexes of state variables (after power flow).
<code>fm_dyncf</code>	Indexes of state variables (before power flow).
<code>fm_flows</code>	Network current/power flows and connectivity.
<code>fm_inilf</code>	Reset variables for power flow computations.
<code>fm_ncomp</code>	Indexes of components.
<code>fm_nrlf</code>	Power flow with fixed state variables.
<code>fm_report</code>	Writes power flow report files.
<code>fm_restore</code>	Reset all components data using the <code>store</code> matrices.
<code>fm_spf</code>	Standard power flow routine.
<code>fm_stat</code>	GUI for displaying power flow results.
<code>fm_xfirst</code>	Initial guess of state variables.

Direct Methods

<code>fm_limit</code>	Limit-induced bifurcation routine.
<code>fm_snb</code>	Saddle-node bifurcation routine.
<code>fm_snbf</code>	GUI for saddle-node bifurcations.
<code>fm_snbf</code>	GUI for limit-induced bifurcations.

Continuation Power Flow (CPF)

<code>fm_cpf</code>	Continuation power flow.
<code>fm_cpffig</code>	GUI for continuation power flow.
<code>fm_n1cont</code>	$N - 1$ contingency computations.

Optimal Power Flow (OPF)

<code>fm_atc</code>	Available transfer capability computations.
<code>fm_opffig</code>	GUI for optimal power flow.
<code>fm_opfm</code>	Optimal power flow.
<code>fm_opfrep</code>	Writes optimal power flow report files.
<code>fm_opfsdr</code>	VS constrained optimal power flow.
<code>fm_pareto</code>	Pareto's set computations.

Small Signal Stability Analysis

<code>fm_eigen</code>	Eigenvalue computations.
<code>fm_eigfig</code>	GUI for eigenvalue computations.

Time Domain Simulation

<code>fm_int</code>	Time domain simulation.
<code>fm_out</code>	Time domain simulation output.
<code>fm_snap</code>	GUI for snapshot settings.
<code>fm_tstep</code>	Definition of time step for transient computations.

Utilities Functions

<code>autorun</code>	Secure routine launch.
<code>benchmark</code>	Check the PSAT routines using benchmark tests.
<code>fm_errv</code>	Check component voltage rating.
<code>fm_filenum</code>	Enumeration of output files.
<code>fm_genstatus</code>	Static generator status.
<code>fm_getxy</code>	Get x and y indexes within a network zone.
<code>fm_idx</code>	Definition of variable names.
<code>fm_iidx</code>	Find bus interconnections.
<code>fm_laprint</code>	Export graphics to <i>eps</i> and L ^A T _E X files.
<code>fm qlim</code>	Get static generator reactive power limits.
<code>fm_rmgcn</code>	Find and remove static generators connected to a bus.
<code>fm_setgy</code>	Delete rows and columns of Jacobian matrix g_y .
<code>fm_status</code>	Display convergence error status on main GUI.
<code>fm_strjoin</code>	Platform independent clone of the <code>strcat</code> function.
<code>fm_vlim</code>	Get bus voltage limits.
<code>fm_windup</code>	Set windup hard limits.
<code>fvar</code>	Convert variables in strings.
<code>pgrep</code>	Search .m files for string.

<code>psatdomain</code>	Dummy function for the PMC SIMULINK library.
<code>psed</code>	Substitute string in <i>.m</i> files.
<code>settings</code>	Define customized settings (optional).
<code>sizefig</code>	Determine figure size.

Simulink Library and Functions

<code>fm_block</code>	Set SIMULINK block parameters.
<code>fm_draw</code>	Draw SIMULINK block icons.
<code>fm_inout</code>	Create and delete SIMULINK block input/output ports.
<code>fm_lib</code>	SIMULINK library.
<code>fm_maskrotate</code>	Manipulation of block masks.
<code>fm_simrep</code>	Power flow report for SIMULINK models.
<code>fm_simsave</code>	Save a SIMULINK 5, 4.1 or 4 model as a SIMULINK 3 model.
<code>fm_simset</code>	GUI for SIMULINK model settings.

Data File Conversion

<code>fm_dir</code>	Browser for data conversion.
<code>fm_dirset</code>	Utilities for data conversion.
<code>filters/cepel2psat</code>	CEPEL to PSAT filter (perl file).
<code>filters/chapman2psat</code>	Chapman's format to PSAT filter (perl file).
<code>filters/cloneblock</code>	Update obsolete PSAT-SIMULINK blocks.
<code>filters/cyme2psat</code>	CYMFLOW to PSAT filter (perl file).
<code>filters/digsilent2psat</code>	DIgSILENT to PSAT filter (perl file).
<code>filters/epri2psat</code>	EPRI to PSAT filter (perl file).
<code>filters/eurostag2psat</code>	Eurostag to PSAT filter (perl file).
<code>filters/flowdemo2psat</code>	FlowDemo.net to PSAT filter (perl file).
<code>filters/ge2psat</code>	GE to PSAT filter (perl file).
<code>filters/ieee2psat</code>	IEEE CDF to PSAT filter (perl file).
<code>filters/inptc12psat</code>	CESI INPTC1 to PSAT filter (perl file).
<code>filters/ipss2psat</code>	InterPSS to PSAT filter (perl file).
<code>filters/ipssdat2psat</code>	InterPSS plain text to PSAT filter (perl file).
<code>filters/matpower2psat</code>	MATPOWER to PSAT filter (m-file).
<code>filters/neplan2psat</code>	NEPLAN to PSAT filter (perl file).
<code>filters/odm2psat</code>	ODM to PSAT filter (perl file).
<code>filters/pcf1o2psat</code>	PCFLO to PSAT filter (perl file).
<code>filters/psap2psat</code>	PSAP to PSAT filter (perl file).
<code>filters/psat2epri</code>	PSAT to EPRI filter (m-file).
<code>filters/psat2ieee</code>	PSAT to IEEE filter (m-file).
<code>filters/psat2octave</code>	adapt PSAT for GNU OCTAVE (perl file).
<code>filters/psat2odm</code>	PSAT to ODM filter (m-file).
<code>filters/psse2psat</code>	PSS/E to PSAT filter (perl file).
<code>filters/pst2psat</code>	PST to PSAT filter (m-file).
<code>filters/pwrworld2psat</code>	POWERWORLD to PSAT filter (perl file).
<code>filters/sim2psat</code>	SIMULINK to PSAT filter (m-file).

<code>filters/simpow2psat</code>	SIMPOW to PSAT filter (perl file).
<code>filters/th2psat</code>	Tsing Hua Univ. to PSAT filter (perl file).
<code>filters/ucte2psat</code>	UCTE to PSAT filter (perl file).
<code>filters/vst2psat</code>	VST to PSAT filter (perl file).
<code>filters/webflow2psat</code>	WebFlow to PSAT filter (perl file).

Plotting Utilities

<code>fm_axesdlg</code>	GUI for axes properties settings.
<code>fm_bar</code>	Plots status bar on main window.
<code>fm_linedlg</code>	GUI for line properties settings.
<code>fm_linenlist</code>	GUI for line list browser.
<code>fm_matrix</code>	GUI for sparse matrix visualization.
<code>fm_plot</code>	General function for plotting results.
<code>fm_plotfig</code>	GUI for plotting results.
<code>fm_threed</code>	GUI for plotting 3D maps.
<code>fm_view</code>	General function for sparse matrix visualization.

Command History

<code>fm_disp</code>	Command, message and error display.
<code>fm_hist</code>	GUI for command history visualization.
<code>fm_text</code>	Command history general functions and utilities.
<code>fval</code>	Message line for variable manipulation.

Output

<code>fm_write</code>	Call function for writing output results.
<code>fm_writehtm</code>	Write output results in HTML format.
<code>fm_writetex</code>	Write output results in L ^A T _E X format.
<code>fm_writetxt</code>	Write output results in plain text.
<code>fm_writexls</code>	Write output results in Excel format.

Themes

<code>fm_mat</code>	Background for GUI images.
<code>fm_theme</code>	Theme manager.
<code>fm_themefig</code>	GUI of theme manager.

Other GUI Utilities

<code>fm_about</code>	About PSAT.
<code>fm_advanced</code>	GUI for advanced settings.
<code>fm_author</code>	Author's pic.
<code>fm_choice</code>	Dialog box.
<code>fm_clock</code>	Analogical watch.
<code>fm_enter</code>	Welcome GUI.

<code>fm_iview</code>	Image viewer.
<code>fm_setting</code>	GUI for general settings.
<code>fm_tviewer</code>	GUI for text viewer selection.

GNU License Functions

<code>gnulicense</code>	Type the GNU GPL .
<code>gnuwarranty</code>	Type the “no warranty” conditions.
<code>fm_license</code>	GUI for the GNU GPL .
<code>fm_warranty</code>	GUI for the “no warranty” conditions.

PMU Placement Functions

<code>fm_annealing</code>	Annealing method for PMU placement.
<code>fm_lssest</code>	Linear static state estimation.
<code>fm_mintree</code>	Minimum tree search.
<code>fm_pmufig</code>	GUI for PMU placement.
<code>fm_pmuloc</code>	PMU placement manager.
<code>fm_pmun1</code>	PMU placement for device outages.
<code>fm_pmurec</code>	Recursive method for PMU placement.
<code>fm_pmurep</code>	Write PMU placement report.
<code>fm_pmutry</code>	Filter for zero-injection buses.
<code>fm_spantree</code>	Spanning tree of existing PMUs.

Command Line Usage

<code>closepsat</code>	Clear all PSAT global variables from workspace.
<code>initpsat</code>	Initialize PSAT global variables.
<code>runpsat</code>	Launch PSAT routine.

Interface Functions

<code>fm_gams</code>	GAMS interface for single-period OPF.
<code>fm_gamsfig</code>	GUI of the GAMS interface.
<code>fm_uwfig</code>	GUI of the UWPFLOW interface.
<code>fm_uwpflow</code>	UWPFLOW interface.

Linear Analysis Functions

<code>fm_abcd</code>	Compute input/output matrices \mathbf{A} , \mathbf{B} , \mathbf{C} , \mathbf{D} .
----------------------	---

Numerical Differentiation Functions

<code>checkjac</code>	Compare numeric and analytic Jacobian matrices.
<code>numjacs</code>	Evaluate numeric Jacobian matrices f_x , f_y , g_x and g_y .

Appendix C

Other Files and Folders

This appendix presents files other than MATLAB functions and scripts that comes with PSAT.

.ini Files

comp definition of component functions, associated structures and a number of Boolean variables for defining the calls of the functions. The format is as follows:

Device name	cols. 1-20
Call algebraic equations	col. 22
Call algebraic Jacobians	col. 24
Call differential equations	col. 26
Call state Jacobians	col. 28
Call hard limits	col. 30
Call during power flow	col. 32
Call initialization	col. 34
Call if computing shunt powers	col. 36
Call if computing series flows	col. 38

history Settings for the command history. The file is updated only if command history settings are saved.

namevarx Definition of state variables names, formatted names in a L^AT_EX syntax and associated component structure names. The variable names are also fields for the correspondent structures. The format is as follows:

Variable name	cols. 1-19
Variable formatted name	cols. 21-29
Component structure name	cols. 41-...

namevary Definition of algebraic variables names, formatted names in a LATEX syntax and associated component structure names. The variable names are also fields for the correspondent structures. The format is the same as for the file `namevarx.ini`.

service Contains a list of variables that are common to different components, such as the generator field voltage or the reference voltage of the excitation systems.

.gms Files

`fm_gams.gms` Single-period OPF routines.

`fm_gams2.gms` Multi-period OPF routines.

`gams/matout.gms` MATLAB-GAMS interface library.

`gams/psatout.gms` PSAT-GAMS interface library.

`psatdata.gms` Input data for the PSAT-GAMS interface.

`psatglobs.gms` Global variables for the PSAT-GAMS Interface.

`psatout.m` Output data for the PSAT-GAMS interface (*m*-file).

Perl Filters

`filters/cepel2psat` Filter for the CEPEL data format.

`filters/chapman2psat` Filter for the Chapman's data format.

`filters/cyme2psat` Filter for the CYMFLOW data format.

`filters/digsilent2psat` Filter for the DIgSILENT data format.

`filters/epri2psat` Filter for the EPRI data format.

`filters/eurostag2psat` Filter for the Eurostag data format.

`filters/flowdemo2psat` Filter for the FlowDemo.net data format.

`filters/ieee2psat` Filter for the IEEE CDF data format.

`filters/inptc12psat` Filter for the CESI INPTC1 data format.

`filters/odm2psat` Filter for the ODM data format.

`filters/pcflo2psat` Filter for the PCFLO data format.

`filters/pcflo2psat` Filter for the PCFLO data format.

`filters/psap2psat` Filter for the PECO-PSAP data format.

filters/psse2psat Filter for the PSS/E 29 data format.

filters/pwrworld2psat Filter for the POWERWORLD auxiliary file format.

filters/red2psat Filter for the REDS file format.

filters/simpow2psat Filter for the SIMPOW file format.

filters/th2psat Filter for the TH data format.

filters/ucte2psat Filter for the UCTE data format.

filters/vst2psat Filter for the VST data format.

filters/webflow2psat Filter for the WebFlow data format.

GNU General Public License

gnulicense.txt Original plain text of the GNU GPL.

Secondary Folders

images Contains the image files used by the graphical user interfaces.

build Contains the MATLAB script files defining the user defined components (*not used*).

themes Contains the themes for customizing the appearance of the graphical user interface.

filters Contains the Perl filters for data format conversions.

gams Contains the PSAT-GAMS interface functions and libraries.

Appendix D

Third Party Matlab Code

A few functions included in the MATLAB distribution have been modified and/or partially rewritten. These functions are:

`imageview.m` changed in `fm_iview.m`.

`inputdlg.m` changed in `fm_input.m`.

`scribeaxesdlg.m` changed in `fm_axesdlg.m`.

`scribelinedlg.m` changed in `fm_linedlg.m`.

`strcat.m` changed in `fm_strjoin.m`.

Appendix E

Power System Software

This appendix provides a list of selected websites of power system software packages. Any help in maintaining this list as complete and updated as possible is greatly appreciated.

ANA's Software	www.cepel.br/servicos/descprog.shtm
ASPEN	www.aspeninc.com
ATP/EMTP	www.emtp.org
CAPE	www.electrocon.com
CDEGS	www.sestech.com
CYME	www.cy.me.com
DCOPFJ	www.econ.iastate.edu/tesfatsi/DCOPFJHome.htm
DEW	www.samsix.com/dew.htm
DIgSILENT	www.digsilent.de
DINIS	www.dinis.com
DMS	www.dmsgroup.co.yu
DSA PowerTools	www.powertechlabs.com
EDSA	www.edsa.com
EMTP-RV	www.emtp.com
ESA Easy Power	www.easypower.com
ETAP	www.etap.com
EUROSTAG	www.eurostag.be
FENDI	www.martinole.org/Fendi/
FlowDemo.net	flowdemo.net
GE-PSLF	www.gepower.com/prod_serv/products/utility_software/en/ge_psif/index.htm
INTELLICON	www.intellicon.biz
InterPSS	www.interpss.org
IPSA	www.ipsa-power.com
MATPOWER	www.pserc.cornell.edu/matpower
MICROTRAN	www.microtran.com
MiPower	www.mipowersoftware.com

NEPLAN	www.neplan.ch
Optimal Aempfast	www.otii.com/aempfast.html
OpeDSS	sourceforge.net/projects/electricdss
PET	www.ece.neu.edu/~abur/pet.html
POM	www.vrenergy.com
POWERWORLD	www.powerworld.com
PSASP	www.psasp.com.cn
PSAT	www.uclm.es/area/gsee/Web/Federico/psat.htm
PSCAD/EMTDC	www.pscad.com
PSS/E	www.pti-us.com
PST	www.eagle.ca/~cherry/pst.htm
QuickStab(R)	www.scscc-us.com
SCOPE	www.nexant.com
SKM Power* Tools	www.skm.com
SIMPOW	www.stri.se
SIMPOWERSYSTEMS	www.mathworks.com/products/simpower/
SPARD(R)	www.energyco.com
SynerGEE	www.advantica.biz
TRANSMISSION 2000	www.cai-engr.com/T2000.htm
UWPFLOW	thunderbox.uwaterloo.ca/~claudio/software/pflow.html
VST	power.ece.drexel.edu/index_files/vst.htm
WebFlow	pw.elec.kitami-it.ac.jp/ueda/demo/

Other useful links are as follows:

- IEEE Task Force on Open Source Software for Power Systems, available at:

ewh.ieee.org/cmte/psace/CAMS_taskforce/index.htm

- IEEE PES PEEC Digital Educational Resources, available at:

www.ece.mtu.edu/faculty/ljbohman/peec/Dig_Rsor.htm

- IEEE Power Systems Test Case Archive, available at:

www.ee.washington.edu/research/pstca/

- Power Systems Dynamic Test Cases Archive, available at:

psdyn.ece.wisc.edu/IEEE_benchmarks/index.htm

- Open-Source Software for Electricity Market Research, Teaching, and Training, available at:

www.econ.iastate.edu/tesfatsi/ElectricOSS.htm

- REpository for Distribution Systems (REDS), available at:

www.ece.ndsu.nodak.edu/~cris/reds.html

Appendix F

Test System Data

This appendix provides the data of the test systems used in the examples of this manual. These are 3-bus, 6-bus, 9-bus, and 14-bus systems. Data are reported in the PSAT data format and were generated by the SIMULINK models provided with the toolbox.¹

F.1 3-bus Test System

Figure F.1 depicts a three-bus test case that represents three generation companies (GENCOs) and one energy supply companies (ESCOs) that provide supply and demand bids. The complete data set for this system is as follows:

```
Bus.con = [ ...
    1 400 1 0 1 1;
    2 400 1 0 2 1;
    3 400 1 0 3 1;
];

Line.con = [ ...
    1 2 100 400 60 0 0 0 0.1 0 0 0 0.4 0.4 0 1;
    1 3 100 400 60 0 0 0 0.1 0 0 0 0.4 0.4 0 1;
    2 3 100 400 60 0 0 0 0.1 0 0 0 0.4 0.4 0 1;
];

SW.con = [ ...
    1 100 400 1 0 1.5 -1.5 1.1 0.9 0.4 1 1 1;
];

PV.con = [ ...
    2 100 400 0.4 1 0.8 -0.2 1.1 0.9 1 1;
    3 100 400 0.4 1 0.8 -0.2 1.1 0.9 1 1;
];

PQ.con = [ ...
    3 100 400 1 0.6 1.2 0.8 1 1;
```

¹The SIMULINK models are placed in the folder `tests` within the PSAT main folder.

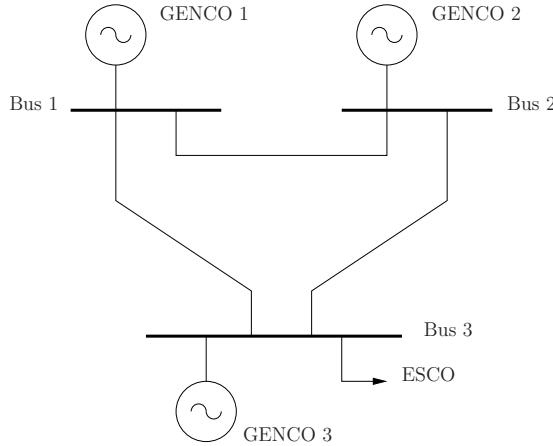


Figure F.1: 3-bus test system.

```

];
Demand.con = [ ...
 3 100 1 0.6 1 1 0 0 0 0 0 0 0 0 0 0 0 1;
];

Supply.con = [ ...
 1 100 0 0.6 0.1 0 6 9.8 0.1 0 0 0 0 0 1 1.5 -1.5 0 0 1;
 2 100 0 0.6 0.1 0 4 10.7 0.2 0 0 0 0 0 1 0.8 -0.2 0 0 1;
 3 100 0 0.6 0.1 0 8 12.6 0.25 0 0 0 0 0 1 0.8 -0.2 0 0 1;
];

Rmpg.con = [ ...
 2 100 0.1 0.1 2 2 5 0 1 1;
 1 100 0.05 0.05 2 2 5 0 1 1;
 3 100 0.15 0.15 2 2 0 5 1 1;
];

Ypdp.con = [ ...
 55 75 100 120 100;
];

Bus.names = {...
 'Bus1'; 'Bus2'; 'Bus3'};

```

F.2 6-bus Test System

Figure F.2 depicts the 6-bus test case, which is extracted from [115], representing three generation companies (GENCOs) and three energy supply companies (ESCOs) that provide supply and demand bids. The complete data of this system are

as follows:

```

Bus.con = [ ...
    1 400 1 0 2 1;
    2 400 1 0 2 1;
    3 400 1 0 2 1;
    4 400 1 0 2 1;
    5 400 1 0 2 1;
    6 400 1 0 2 1;
];

Line.con = [ ...
    2 3 100 400 60 0 0 0.05 0.25 0.06 0 0 0.3082 0 0 1;
    3 6 100 400 60 0 0 0.02 0.1 0.02 0 0 1.3973 0 0 1;
    4 5 100 400 60 0 0 0.2 0.4 0.08 0 0 0.1796 0 0 1;
    3 5 100 400 60 0 0 0.12 0.26 0.05 0 0 0.6585 0 0 1;
    5 6 100 400 60 0 0 0.1 0.3 0.06 0 0 0.2 0 0 1;
    2 4 100 400 60 0 0 0.05 0.1 0.02 0 0 1.374 0 0 1;
    1 2 100 400 60 0 0 0.1 0.2 0.04 0 0 0.2591 0 0 1;
    1 4 100 400 60 0 0 0.05 0.2 0.04 0 0 0.9193 0 0 1;
    1 5 100 400 60 0 0 0.08 0.3 0.06 0 0 0.8478 0 0 1;
    2 6 100 400 60 0 0 0.07 0.2 0.05 0 0 0.9147 0 0 1;
    2 5 100 400 60 0 0 0.1 0.3 0.04 0 0 0.7114 0 0 1;
];

SW.con = [ ...
    2 100 400 1.05 0 1.5 -1.5 1.1 0.9 1.4 1 1 1;
];

PV.con = [ ...
    1 100 400 0.9 1.05 1.5 -1.5 1.1 0.9 1 1;
    3 100 400 0.6 1.05 1.5 -1.5 1.1 0.9 1 1;
];

PQ.con = [ ...
    4 100 400 0.9 0.6 1.1 0.9 0 1;
    5 100 400 1 0.7 1.1 0.9 0 1;
    6 100 400 0.9 0.6 1.1 0.9 0 1;
];

Demand.con = [ ...
    6 100 0.2 0.066666 0.2 1e-05 0 0 9.5 0 0 0 0 0 0 0 0 0 1;
    5 100 0.1 0.07 0.1 1e-05 0 0 10.5 0 0 0 0 0 0 0 0 0 1;
    4 100 0.25 0.166665 0.25 1e-05 0 0 12 0 0 0 0 0 0 0 0 0 1;
];

Supply.con = [ ...
    1 100 0.2 0.2 1e-05 0 0 9.7 0 0 0 0 0 0 1 1.5 -1.5 0 0 1;
    2 100 0.25 0.25 1e-05 0 0 8.8 0 0 0 0 0 0 1 1.5 -1.5 0 0 1;
    3 100 0.2 0.2 1e-05 0 0 7 0 0 0 0 0 0 1 1.5 -1.5 0 0 1;
];

Bus.names = {...
    'Bus1'; 'Bus2'; 'Bus3'; 'Bus4'; 'Bus5';
    'Bus6'};

```

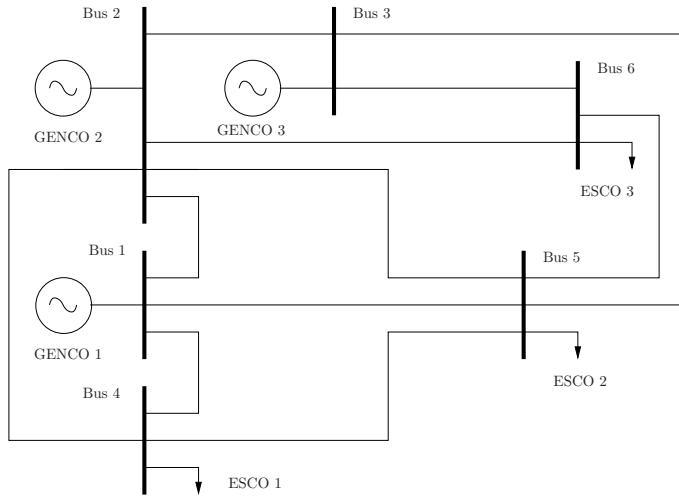


Figure F.2: 6-bus test system.

F.3 9-bus Test System

Figure F.3 depicts the 9-bus test system, which is extracted from [111] and represents three generators (order IV) with AVR (type II). The complete data of this system are as follows:

```

Bus.con = [ ...
  1 16.5 1 0 4 1;
  2 18 1 0 5 1;
  3 13.8 1 0 3 1;
  4 230 1 0 2 1;
  5 230 1 0 2 1;
  6 230 1 0 2 1;
  7 230 1 0 2 1;
  8 230 1 0 2 1;
  9 230 1 0 2 1;
];
Line.con = [ ...
  9 8 100 230 60 0 0      0.0119 0.1008 0.209 0 0 0 0 0 0 1;
  7 8 100 230 60 0 0      0.0085 0.072 0.149 0 0 0 0 0 0 1;
  9 6 100 230 60 0 0      0.039 0.17 0.358 0 0 0 0 0 0 1;
  7 5 100 230 60 0 0      0.032 0.161 0.306 0 0 0 0 0 0 1;
  5 4 100 230 60 0 0      0.01 0.085 0.176 0 0 0 0 0 0 1;
  6 4 100 230 60 0 0      0.017 0.092 0.158 0 0 0 0 0 0 1;
  2 7 100 18 60 0 0.07826087 0      0.0625 0 0 0 0 0 0 0 0 1;
  3 9 100 13.8 60 0 0.06 0      0.0586 0 0 0 0 0 0 0 0 1;
  1 4 100 16.5 60 0 0.07173913 0      0.0576 0 0 0 0 0 0 0 0 1;
];
SW.con = [ ...
  1 100 16.5 1.04 0 99 -99 1.1 0.9 0.8 1 1 1;

```

```

];
PV.con = [ ...
 2 100 18 1.63 1.025 99 -99 1.1 0.9 1 1;
 3 100 13.8 0.85 1.025 99 -99 1.1 0.9 1 1;
];

PQ.con = [ ...
 6 100 230 0.9 0.3 1.2 0.8 0 1;
 8 100 230 1 0.35 1.2 0.8 0 1;
 5 100 230 1.25 0.5 1.2 0.8 0 1;
];

Syn.con = [ ...
 1 100 16.5 60 4 0 0 0.146 0.0608 0 8.96 0 0.0969 0.0969 0 ...
 0.31 0 47.28 0 0 0 1 1 0.002 0 0 1 1;
 2 100 18 60 4 0 0 0.8958 0.1198 0 6 0 0.8645 0.1969 0 ...
 0.535 0 12.8 0 0 0 1 1 0.002 0 0 1 1;
 3 100 13.8 60 4 0 0 1.3125 0.1813 0 5.89 0 1.2578 0.25 0 ...
 0.6 0 6.02 0 0 0 1 1 0.002 0 0 1 1;
];

Exc.con = [ ...
 1 2 5 -5 20 0.2 0.063 0.35 1 0.314 0.001 0.0039 1.555;
 2 2 5 -5 20 0.2 0.063 0.35 1 0.314 0.001 0.0039 1.555;
 3 2 5 -5 20 0.2 0.063 0.35 1 0.314 0.001 0.0039 1.555;
];

Bus.names = {...
 'Bus 1'; 'Bus 2'; 'Bus 3'; 'Bus 4'; 'Bus 5';
 'Bus 6'; 'Bus 7'; 'Bus 8'; 'Bus 9'};

```

A second model of this system is described in [6] and presents a simplified model of generators (order II) without AVR. The complete data of this system are as follows:

```

Bus.con = [ ...
 1 16.5 1 0 4 1;
 2 18 1 0 5 1;
 3 13.8 1 0 3 1;
 4 230 1 0 2 1;
 5 230 1 0 2 1;
 6 230 1 0 2 1;
 7 230 1 0 2 1;
 8 230 1 0 2 1;
 9 230 1 0 2 1;
];

Line.con = [ ...
 9 8 100 230 60 0 0 0.0119 0.1008 0.209 0 0 0 0 0 0 1;
 7 8 100 230 60 0 0 0.0085 0.072 0.149 0 0 0 0 0 0 1;
 9 6 100 230 60 0 0 0.039 0.17 0.358 0 0 0 0 0 0 1;
 7 5 100 230 60 0 0 0.032 0.161 0.306 0 0 0 0 0 0 1;
 5 4 100 230 60 0 0 0.01 0.085 0.176 0 0 0 0 0 0 1;
 6 4 100 230 60 0 0 0.017 0.092 0.158 0 0 0 0 0 0 1;
 2 7 100 18 60 0 0.07826087 0 0.0625 0 0 0 0 0 0 0 1;

```

```

3 9 100 13.8 60 0 0.06 0 0.0586 0 0 0 0 0 0 1;
1 4 100 16.5 60 0 0.07173913 0 0.0576 0 0 0 0 0 0 1;
];

Breaker.con = [ ...
4 7 100 230 60 1 1.083 4 1 0;
];

Fault.con = [ ...
7 100 230 60 1 1.083 0 0.001;
];

SW.con = [ ...
1 100 16.5 1.04 0 99 -99 1.1 0.9 0.8 1 1 1;
];

PV.con = [ ...
2 100 18 1.63 1.025 99 -99 1.1 0.9 1 1;
3 100 13.8 0.85 1.025 99 -99 1.1 0.9 1 1;
];

PQ.con = [ ...
6 100 230 0.9 0.3 1.2 0.8 0 1;
8 100 230 1 0.35 1.2 0.8 0 1;
5 100 230 1.25 0.5 1.2 0.8 0 1;
];

Syn.con = [ ...
2 100 18 60 2 0.0521 0 0.8958 0.1198 0 6 0 0.8645 0.1969 ...
0 0.535 0 12.8 0 0 0 1 1 0.002 0 0 1 1;
3 100 13.8 60 2 0.0742 0 1.3125 0.1813 0 5.89 0 1.2578 0.25 ...
0 0.6 0 6.02 0 0 0 1 1 0.002 0 0 1 1;
1 100 16.5 60 2 0.0336 0 0.146 0.0608 0 8.96 0 0.0969 0.0969 ...
0 0.31 0 47.28 0 0 0 1 1 0.002 0 0 1 1;
];

Bus.names = {...  

'Bus 1'; 'Bus 2'; 'Bus 3'; 'Bus 4'; 'Bus 5';
'Bus 6'; 'Bus 7'; 'Bus 8'; 'Bus 9'};

```

F.4 14-bus Test System

Figure F.4 depicts the IEEE 14-bus test system, which is a benchmark for power system analysis.² The complete data of this system are as follows:

```

Bus.con = [ ...
1 69 1 0 4 1;
2 69 1 0 4 1;
3 69 1 0 4 1;
4 69 1 0 4 1;
5 69 1 0 4 1;
6 13.8 1 0 2 1;

```

² Available at <http://www.ee.washington.edu/research/pstca/>.

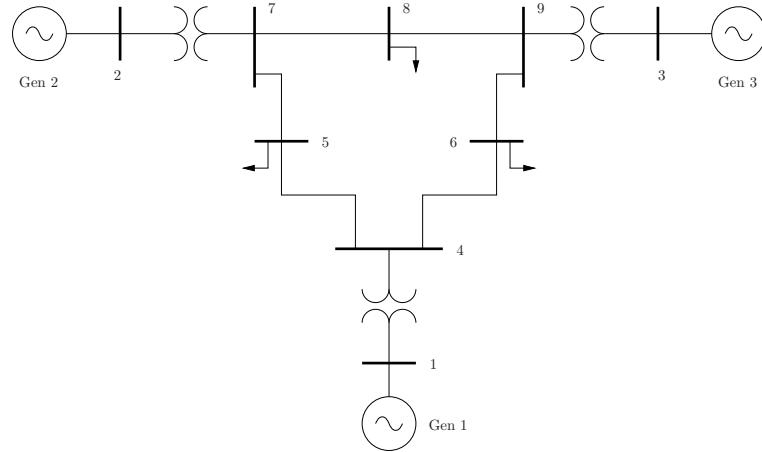


Figure F.3: WSCC 3-generator 9-bus test system.

```

7 13.8 1 0 2 1;
8 18 1 0 3 1;
9 13.8 1 0 2 1;
10 13.8 1 0 2 1;
11 13.8 1 0 2 1;
12 13.8 1 0 2 1;
13 13.8 1 0 2 1;
14 13.8 1 0 2 1;
];
Line.con = [ ...
 2 5 100 69 60 0 0      0.05695 0.17388 0.034 0 0 0 0 0 1;
 6 12 100 13.8 60 0 0    0.12291 0.25581 0 0 0 0 0 0 1;
12 13 100 13.8 60 0 0    0.22092 0.19988 0 0 0 0 0 1;
 6 13 100 13.8 60 0 0    0.06615 0.13027 0 0 0 0 0 1;
 6 11 100 13.8 60 0 0    0.09498 0.1989 0 0 0 0 0 1;
11 10 100 13.8 60 0 0   0.08205 0.19207 0 0 0 0 0 1;
 9 10 100 13.8 60 0 0   0.03181 0.0845 0 0 0 0 0 1;
 9 14 100 13.8 60 0 0   0.12711 0.27038 0 0 0 0 0 1;
14 13 100 13.8 60 0 0   0.17093 0.34802 0 0 0 0 0 1;
 7 9 100 13.8 60 0 0     0 0.11001 0 0 0 0 0 0 1;
 1 2 100 69 60 0 0     0.01938 0.05917 0.0528 0 0 0 0 0 1;
 3 2 100 69 60 0 0     0.04699 0.19797 0.0438 0 0 0 0 0 1;
 3 4 100 69 60 0 0     0.06701 0.17103 0.0346 0 0 0 0 0 1;
 1 5 100 69 60 0 0     0.05403 0.22304 0.0492 0 0 0 0 0 1;
 5 4 100 69 60 0 0     0.01335 0.04211 0.0128 0 0 0 0 0 1;
 2 4 100 69 60 0 0     0.05811 0.17632 0.0374 0 0 0 0 0 1;
 4 9 100 69 60 0 5     0.005 0.55618 0 0.969 0 0 0 0 0 1;
 5 6 100 69 60 0 5     0 0.25202 0 0.932 0 0 0 0 0 1;
 4 7 100 69 60 0 5     0 0.20912 0 0.978 0 0 0 0 0 1;
 8 7 100 18 60 0 1.304348 0 0.17615 0 0 0 0 0 0 1;
];
SW.con = [ ...

```

```
1 100 69 1.06 0 9.9 -9.9 1.2 0.8 2.324 1 1 1;
];

PV.con = [ ...
2 100 69 0.4 1.045 0.5 -0.4 1.2 0.8 1 1;
6 100 13.8 0 1.07 0.24 -0.06 1.2 0.8 1 1;
3 100 69 0 1.01 0.4 0 1.2 0.8 1 1;
8 100 18 0 1.09 0.24 -0.06 1.2 0.8 1 1;
];

PQ.con = [ ...
11 100 13.8 0.049 0.0252 1.2 0.8 0 1;
13 100 13.8 0.189 0.0812 1.2 0.8 0 1;
3 100 69 1.3188 0.266 1.2 0.8 0 1;
5 100 69 0.1064 0.0224 1.2 0.8 0 1;
2 100 69 0.3038 0.1778 1.2 0.8 0 1;
6 100 13.8 0.1568 0.105 1.2 0.8 0 1;
4 100 69 0.6692 0.056 1.2 0.8 0 1;
14 100 13.8 0.2086 0.07 1.2 0.8 0 1;
12 100 13.8 0.0854 0.0224 1.2 0.8 0 1;
10 100 13.8 0.126 0.0812 1.2 0.8 0 1;
9 100 13.8 0.413 0.2324 1.2 0.8 0 1;
];

Bus.names = {...  

'Bus 01'; 'Bus 02'; 'Bus 03'; 'Bus 04'; 'Bus 05';
'Bus 06'; 'Bus 07'; 'Bus 08'; 'Bus 09'; 'Bus 10';
'Bus 11'; 'Bus 12'; 'Bus 13'; 'Bus 14'};
```

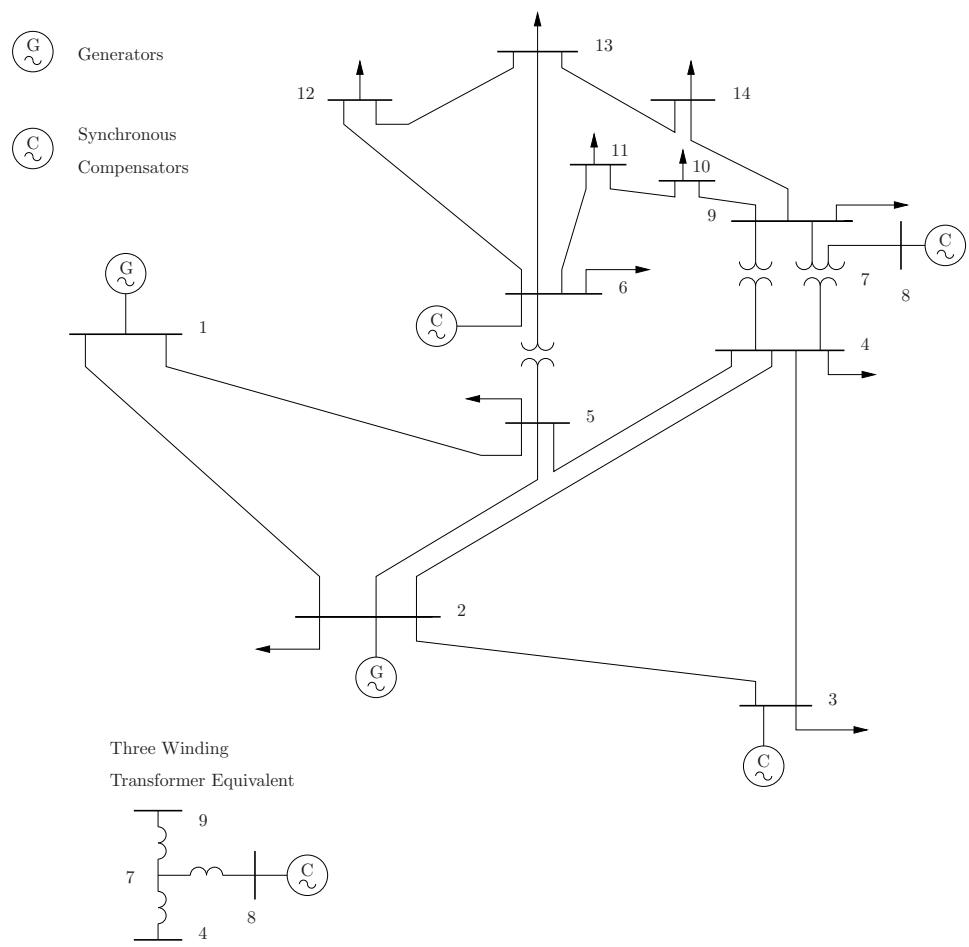


Figure F.4: IEEE 14-bus test system.

Appendix G

FAQs

This appendix presents the most frequent asked questions related to PSAT installation and usage. Following FAQs were mostly raised by users of previous PSAT versions 1.0.x, 1.1.x, and 1.2.x, thus might not apply to the current release.

G.1 Getting Started

When I run PSAT at the Matlab prompt, I got an error messages, as follows:

```
??? Undefined function or variable 'fm_mat'.  
  
Error in ==> C:\psat\fm_main.m  
On line 217 ==>  
  
Error in ==> C:\psat\psat.m  
On line 348 ==> if failed, disp(' '), disp('PSAT is not properly initialized.'),  
else, fm_main, end  
  
??? Error: Missing operator, comma, or semicolon.  
  
??? Error while evaluating figure WindowButtonMotionFcn.  
  
??? Error: Missing operator, comma, or semicolon.  
  
??? Error while evaluating figure WindowButtonMotionFcn.  
  
??? Error: Missing operator, comma, or semicolon.  
  
??? Error while evaluating figure WindowButtonMotionFcn.  
  
??? Error: Missing operator, comma, or semicolon.  
  
....
```

The reason of this error is that the PSAT folder is not set in the MATLAB search path. Some users get confused between the *current* MATLAB path which

is the working folder where MATLAB first looks for custom functions¹ and the MATLAB *search* path which is a list of folders where MATLAB looks for functions if the search in the current folder fails.² The previous PSAT documentation wasn't clear on this regard. Please refer to the new Section 2.3 for a better explanation on how to properly install PSAT on your system.

Which are the differences between PSAT and SimPowerSystems in terms of features, applications and performance?

SimPowerSystems (alias Power System Blockset) is a Simulink-based toolbox for electromechanical transient studies (including detailed models of power electronic components), while PSAT is MATLAB-based and aimed to power flow, optimal power flow, continuation power flow and electromechanical transients.

A very rough comparison of the two software packages is depicted in Table 1.1 of Chapter 1. However, comparing the two software packages is not fair, because they have different goals and use different mathematical models. Maybe it could be interesting comparing power flow results obtained with PSAT and SimPowerSystems, which I think is the only comparable result.

Performances of both toolboxes are typically pretty good for "small" systems, while slow down for "huge" ones. This actually depends on MATLAB features more than on the implemented code. Of course "small" and "huge" depend on the computer.

However, PSAT is free software (well, free but for the MATLAB kernel :)), while SimPowerSystems is a commercial product. Any comments, suggestions and contribution are really welcome and will be taken into account in order to make PSAT a better software and a more reliable and useful tool. I guess this is actually the main advantage of PSAT.

How can I run PSAT from within a function without using GUIs?

Since PSAT version 1.3.0, PSAT includes a set of functions and script files which allow avoiding GUIs. Please refer to Chapter 28 for a detailed documentation about the command line usage of PSAT.

Can I run PSAT on GNU Octave?

As for version 1.3.0, PSAT can run on GNU OCTAVE. Some restrictions and limitations apply. See Chapter 29.

¹The MATLAB current path is returned by the `pwd` function.

²The MATLAB search path is returned by the `path` function.

G.2 Simulink Library

How can I inspect schemes of PSAT-Simulink blocks?

PSAT-SIMULINK blocks are hollow, and works just as data boxes. As a matter of fact running a simulation from the SIMULINK toolbar produces no effects. Static and dynamic models of components are stored in the MATLAB functions provided with the PSAT tarball.

I added a control scheme to a PSAT-Simulink model, but it doesn't work.

PSAT makes use of SIMULINK only as a CAD tool, whereas mathematical models are defined in MATLAB functions.

Why PSAT-Simulink blocks do not work in Simulink models built using PSB (SymsPowerSystems)?

Mixing PSAT blocks with PSB blocks is **not** possible: the two toolboxes work in a completely different way.

Why do I get the following message?

```
Statistics ...
'perl' is not recognized as an internal or external command,
operable program or batch file.
Check of SIMULINK blocks couldn't be performed.
```

That simply means *perl* is not properly installed on your system. The message is just a warning and does not affect simulations.

G.3 Power Flow

I tried to run a *n*-thousands bus test system on PSAT, but it took a long time to get the solution. Is there any hope to get a faster solution?

PSAT is a MATLAB based program, thus it cannot be competitive with commercial C-compiled programs. The power flow can be solved faster by means of a fast decoupled technique; however continuation power flow, optimal power flow and time domain simulation are based on the full system Jacobian matrix and will show poor performances for huge networks.

PST and PSAT produce different power flow results for the IEEE 14-bus test system. Why?

The solution of the IEEE 14-bus test system depends on the power flow settings. PST automatically takes into account generator reactive power limits, whereas PSAT basic power flow routines does not. Since PSAT version 1.2.1, it is possible

to enforce generator reactive power limit control in power flow computations, which allows producing same results as PST. However, for a more accurate power flow analysis which includes security limits, it is recommended running the continuation power flow.

Is there a realistic case (thousands of buses) test system for PSAT?

Although PSAT has been successfully used for solving power flows of big networks (a user told me he solved a 25000-bus system power flow with PSAT), these networks are not available because of copyright reasons.

G.4 Optimal & Continuation Power Flow

Why the OPF routine did not converge?

Typically the Interior Point Method does not converge for the two following reasons:

1. the initial guess is out of the feasibility region;
2. maximum or minimum values of some constrained variables are inconsistent.

The OPF routine performs several checks before running the main loop, however more work has to be done on this issue.

I converted a Matpower test case, but the PSAT optimal power flow routine didn't reach the convergence. Why?

PSAT makes a distinction between base case powers (used for the power flow solution) and power bids (used in the continuation and optimal power flow analysis). When importing a MATPOWER test case into PSAT, one has to disable the “Use base case” option in the OPF settings GUI. MATPOWER and PSAT may give different results since the Interior Point Method implemented in PSAT does not include unit commitment so far.

G.5 Time Domain Simulation

Can you give me an example of perturbation file?

Basic disturbances, such as fault and breaker interventions, are embedded in the program. However, all other perturbations have to be implemented by the user. For instance, a perturbation file for the 14-bus test system is as follows:

```
function dummy = p_test(t)

global PQ

if (t > 1.0)

PQ.con(:,[4 5]) = 1.2*[ ...
0.217    0.127;
```

```

0.942    0.19;
0.478    0.04;
0.076    0.016;
0.112    0.075;
0.295    0.166;
0.09     0.058;
0.035    0.018;
0.061    0.016;
0.135    0.058;
0.149    0.05 ];

else

PQ.con(:,[4 5]) = [ ...
0.217    0.127;
0.942    0.19;
0.478    0.04;
0.076    0.016;
0.112    0.075;
0.295    0.166;
0.09     0.058;
0.035    0.018;
0.061    0.016;
0.135    0.058;
0.149    0.05 ];

end

```

It increase the powers of all PQ loads by 20% at $t = 1s$. A perturbation file should typically contain a declaration of global structures which have to be modified and a *if-then-else* control flow. Although a little bit rusty, this procedure gives the maximum freedom in the definition of the event(s) that disturb(s) the network.

I included a fault/breaker in my network but, when running time domain simulations, nothing happens or the routine stops with an error.

This was due to a bug in the data format of fault/breaker components of the previous version 1.0.1. The bug has been fixed in the current version 1.2.0. See also the advanced settings in Section 27.3.

G.6 Data Conversion

When I try to convert a data file in *xyz* format, a warning window shows up with the message “Filter for *xyz* data format not implemented yet”.

The message literally means that the data format filter has not been implemented and there is no way to convert automatically the source data file into PSAT data format. The creation of data format filters is limited by the availability of a complete documentation of commercial data formats. If you have access to a commercial package for power system analysis and want to create a filter, you can either post me the documentation or write the filter by yourself. In the latter case, I will be glad to include your function in the master program.

I converted a data file in *xyz* format, but when I run the power flow, PSAT results are different from what expected.

The conversion of data files from different data formats can be in some cases tricky, since different programs may have different features or treat data in a different way. Most of the time it is just a matter of properly adjusting the general settings of PSAT. Please report all inconsistencies to me in order to improve the filters.

G.7 Interfaces

I have installed the demo version of GAMS but when I try to run the PSAT-GAMS interface, I get the following error:

```
PSAT-GAMS Interface

Market Clearing Mechanism
Single-Period Auction

"gams" is not recognized like an internal or external command,
program or feasible batch file.

?? Error using ==> fm_gams/psatgams
Too many output arguments.
Error in ==> c:/documents and settings/psat/fm_gams.m
On line 382 ==
?? Error using ==> edit
Neither 'fm_gams/psatgams' nor 'fm_gams/psatgams.m' could be found.
```

The problem it is probably due to the fact that your GAMS folder is not set as an environment variable. How to set GAMS executable files as environment variables depends on the operating system, as follows:

Windows NT and Windows 2000 look for Control Panel → System Properties → Advanced Options → Environment Variables. Then edit the “Path” by adding the full GAMS path.

Windows XP look for Control Panel → Performance and Maintenance → System. A windows with the title “System Properties” will show up. Select the “Advanced” tab and push the “Environment Variables” button. Then edit the “PATH” field by adding the full GAMS path.

Linux edit the `.bash_profile` file (or whatever file where your \$PATH variable is defined) in your home directory and add the full GAMS path in the \$PATH variable.

Following errors are just due to the fact that GAMS didn't run successfully and output files (expected by `fm_gams.m`) were not created.

I have done all steps indicated in Chapter 30, but the PSAT-GAMS interface is still not working.

First, please make sure that you have done **all** the appropriate steps indicated in Chapter 30 referring to the PGI (Psat Gams Interface) installation. A usual problem which is used to show up on Windows XP is that the PSAT folder needs to be the start up folder for MATLAB. Here's what you should do:

1. Go to your desktop in XP and right click on the MATLAB icon.
2. Indicate the full PSAT path in the destination field.

Appendix H

PSAT Forum

A PSAT Forum (see Fig. H.1) is currently available at:

tech.groups.yahoo.com/groups/psatforum

Main functions are as follows:

Function	e-mail
Subscribe	psatforum-subscribe@yahoogroups.com
Post message	psatforum@yahoogroups.com
Unsubscribe	psatforum-unsubscribe@yahoogroups.com
List owner	psatforum-owner@yahoogroups.com

To post a message directly to me, the following e-mail address:

1. Federico.Milano@uclm.es

The latest PSAT distribution archive, as well as latest patches and, when available, data files will be posted on the Forum file repository. However, the web site www.uclm.es/area/gsee/Web/Federico/psat.htm will remain the main source for downloading PSAT and related files.

Forum user statistics are depicted in Fig. H.2.

Yahoo! My Yahoo! Mail More

YAHOO! GROUPS

Make Y! My Homepage New User? Sign Up Sign In Help

Search WEB SEARCH

psatforum · PSAT Forum Search for other groups...

Home **Attachments**

Members Only

- Messages
- Post
- Files
- Photos
- Links
- Database
- Polls
- Members
- Calendar
- Promote
- Groups Labs (Beta)

Info **Settings**

Group Information

Members: 1913 Category: Software Founded: Aug 16, 2003 Language: English

Already a member? [Sign in to Yahoo!](#)

Yahoo! Groups Tips

Did you know... Show off your group to the world. Share a photo of your group with us.

Best of Y! Groups

Check them out and nominate your group.

Stay up to speed on the latest Groups news and updates, visit the Groups blog today!

Join This Group!

Activity within 7 days: 5 New Members - 30 New Messages - 2 New Files - New Questions

Description

Web forum for users of the Power System Analysis Toolbox (PSAT).

PSAT is a Matlab toolbox for electric power system analysis and control. The command line version of PSAT is also GNU Octave compatible. PSAT includes power flow, continuation power flow, optimal power flow, small signal stability analysis and time domain simulation. All operations can be assessed by means of graphical user interfaces (GUIs) and a Simulink-based library provides an user-friendly tool for network design.

Message History

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2009	70	67	56	151	106	70						
2008	52	99	115	101	43	89	135	82	117	163	74	52
2007	163	156	188	178	87	93	139	116	98	131	67	46
2006	119	168	202	251	200	118	175	207	157	206	122	121
2005	80	83	60	104	111	81	123	127	123	129	71	95
2004	46	31	63	76	66	90	70	48	81	46	67	37
2003								7	56	27	43	21

Yahoo! Answers What's this? Hide this tip

What is Yahoo! Answers? Yahoo! Answers, a new Yahoo! community, is a question and answer exchange where the world gathers to share what they know... and make each other's day. People can ask questions on any topic, and help others out by answering their questions.

Questions in Computers & Internet > Software

itunes floating point help please!!!!!! Asked By Object - 0 answers - 2 minutes ago - Answer Now

pdf problem - when ever i try to open a pdf, i just get a blank screen, have adobe reader set up, please help? Asked By Object - 0 answers - 2 minutes ago - Answer Now

Could I please have some help with Sims 3? Asked By cospidie08 - 0 answers - 5 minutes ago - Answer Now

how do i get my quest helper installed with vista? Asked By Object - 0 answers - 6 minutes ago - Answer Now

In my task manager there is something called 'system'? Asked By Moses - 2 answers - 8 minutes ago - Answer Now

Want to help answer other questions? Go to Yahoo! Answers

Group Email Addresses

Related Link: <http://www.power.uwaterloo.ca/~fmilano/>
Post message: psatforum@yahoogroups.com
Subscribe: psatforum-subscribe@yahoogroups.com

1 of 2 18/6/09 12:23

Figure H.1: PSAT Forum main page. Data refer to June 18, 2009.

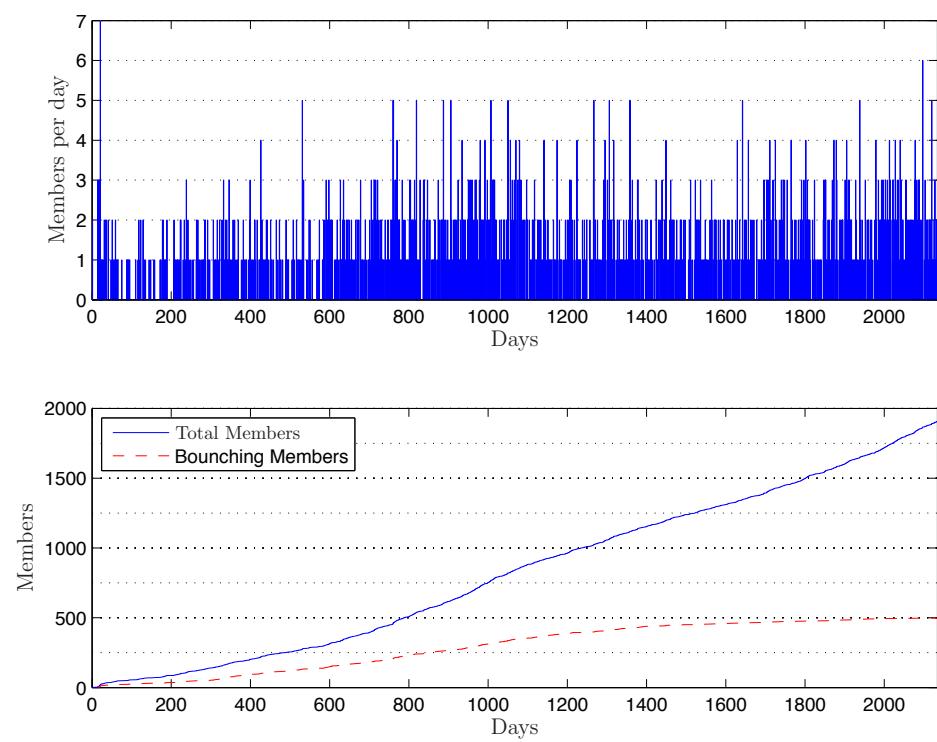


Figure H.2: PSAT Forum statistics. Data refer to June 18, 2009.

Appendix I

Citations & Links

A list of books, PhD dissertations, papers and web-pages that are about, use or cite PSAT follows. The list can be incomplete; please let me know missing references.

I.1 Books

- [1] J. Chow, F. F. Wu, and J. Momoh, *Applied Mathematics for Restructured Electric Power Systems*. Springer-Verlag, 2005, reference in Chapter 8, *Instability Monitoring and Control of Power Systems*, by E. H. Abed, M. A. Hassouneh and M. S. Saad, from page 171.

I.2 Ph.D. Dissertations

- [1] I. Kopcak, “Uma Plataforma Unificada para Análise de Estabilidade de Sistemas Elétricos de Potência,” Ph.D. dissertation, Faculdade de Engenharia Elétrica e de Computação, Unicamp, Barão Geraldo, São Paulo, Brazil, 2007.
- [2] A. Y. Takahata, “Segurança de Sistema de Potência sob Pequenas Perturbações Considerando Incertezas,” Ph.D. dissertation, Universidade Federal do Rio de Janeiro, COPPE, Rio de Janeiro, Brazil, 2008.

I.3 Journals

- [1] M. S. Castro, H. M. Ayres, and L. C. P. da Silva, “Impacts of the SSSC Control Modes on Small-Signal and Transient Stability of a Power System,” *Electric Power System Research*, 2006, in press, available on-line since February 2006.
- [2] M. S. Castro, H. M. Ayres, I. Kopcak, V. F. da Costa, and L. C. P. da Silva, “A Influência do Modo de Operação do SSSC na Estabilidade de Ângulo de Sistemas Elétricos de Potência,” *Revista Controle e Automação*, vol. 13, no. 3, pp. 347–360, July-September 2007, available at www.scielo.br.

- [3] S. El-Kashlan, M. Abdel-Rahman, H. El-Desouki, and M. Mansour, "Voltage Stability of Wind Power Systems using Bifurcation Analysis," *Power and Energy Systems*, vol. 468, 2005.
- [4] S. V. N. Jithin-Sundar and M. Reshma, "Utilization of Controlled Shunt Reactor in a 400 kV Interconnected Network," *International Journal of Emerging Electric Power Systems*, vol. 2, no. 1, 2005.
- [5] M. Larsson, "ObjectStab, An Educational Tool for Power System Stability Studies," *IEEE Transactions on Power Systems*, vol. 19, no. 1, pp. 56–63, Feb. 2004.
- [6] F. Milano, "An Open Source Power System Analysis Toolbox," *IEEE Transactions on Power Systems*, vol. 20, no. 3, pp. 1199–1206, Aug. 2005.
- [7] F. Milano, C. A. Cañizares, and A. J. Conejo, "Sensitivity-based Security-constrained OPF Market Clearing Model," *IEEE Transactions on Power Systems*, vol. 20, no. 4, pp. 2051–2060, Nov. 2005.
- [8] F. Milano, L. Vanfretti, and J. C. Morataya, "An Open Source Power System Virtual Laboratory: The PSAT Case and Experience," *IEEE Transactions on Education*, vol. 51, no. 1, pp. 17–23, Feb. 2008.
- [9] S. N. Pandey, S. Tapaswi, and L. Srivastava, "Nodal Congestion Price Estimation in Spot Power Market Using Artificial Neural Network," *IET Generation, Transmission & Distribution*, vol. 2, no. 2, pp. 280–290, 2008.

I.4 Conference Proceedings

- [1] H. B. Çetinkaya, S. Öztürk, and B. Alboyaci, "Eigenvalues Obtained with Two Simulation Packages (SIMPOW and PSAT) and Effects of Machine Parameters on Eigenvalues," in *Proc. of Melecon 2004*, Dubrovnik, Croatia, May 2004.
- [2] ——, "Machine Parameters and Orders of Machine Impacts on Eigenvalues and Simulations in two Software Packages SIMPOW and PSAT," in *Proc. of IEEE SoutheastCon*, Greensboro, North Carolina, Mar. 2004.
- [3] A. D. Del Rosso and C. A. Negri, "Influencia del Modelado de la Carga en la Evaluación de la Estabilidad Transitoria en Sistemas de Potencia," in *Undécimo Encuentro Regional Iberoamericano del Cigré, XI ERIAC*, Hernandarias, Paraguay, May 2005.
- [4] A. M. Haidar, A. Mohamed, and A. Hussain, "Power System Vulnerability Assessment Considering a New Index Based on Power System Loss," in *International Conference on Energy and Environment*, Bangi, Malaysia, Aug. 2006.

- [5] A. W. N. Izzri, A. Mohamed, and I. Yahya, "A New Method of Transient Stability Assessment in Power Systems Using LS-SVM," in *The 5th Student Conference on Research and Development - SCOReD 2007*, Malaysia, Dec. 2007.
- [6] D. Koesrindartoto, J. Sun, and L. Tesfatsion, "An Agent-Based Computational Laboratory for Testing the Economic Reliability of Wholesale Power Market Designs," in *IEEE PES Conference Proceedings*, San Francisco, California, June 2005.
- [7] I. Kopcak, H. M. Ayres, L. C. P. da Silva, and V. F. da Costa, "Análise Simultânea das Margens de Estabilidade Dinâmica e Estática de Sistemas Elétricos de Potência," in *Anais do VI CLAGTEE 2007 Congresso Latinoamericano de Geração e Transmissão de Energia Elétrica*, Valparaíso, Chile, Nov. 2007, pp. 1–6.
- [8] I. Kopcak, V. F. da Costa, and L. C. P. da Silva, "A Generalized Power Flow Method Including the Steady State Characteristic of Dynamic Devices," in *Proceedings of IEEE 2007 PowerTech Conference*, Lausanne, Switzerland, July 2007, pp. 1–6.
- [9] I. Kopcak, L. C. P. da Silva, and V. F. da Costa, "A Generalized Load Flow Method to Assess Steady-State Equilibrium of Power Systems," in *Proceedings of IEEE PES PowerAfrica 2007 Conference and Exposition*, Johannesburg, South Africa, July 2007, pp. 1–7.
- [10] F. Milano, "A Graphical and Open-Source Matlab-GAMS Interface for Electricity Markets Models," in *Noveno Congreso Hispano-Luso de Ingeniería Eléctrica, CHLIE*, Marbella, Spain, June 2005.
- [11] R. Natesan and G. Radman, "Effects of STATCOM, SSSC and UPFC on Voltage Stability," in *Proceedings of the Thirty-Sixth Southeastern Symposium on System Theory*, Atlanta, Georgia, Mar. 2004, pp. 546–550.
- [12] L. Vanfretti and F. Milano, "Application of the PSAT, an Open Source Software, for Educational and Research Purposes," in *IEEE PES General Meeting*, Tampa, USA, June 2007.
- [13] E. M. Yap, M. Al-Dabbagh, and P. C. Thum, "UPFC Controller in Mitigating Line Congestion for Cost-efficient Power Delivery," in *The 7th International Power Engineering Conference, IPEC 2005*, Singapore, November 29–December 2, 2005.

I.5 Web-pages

☞ IEEE PES PEEC Digital Educational Resources, available at:

www.ece.mtu.edu/faculty/ljbohman/peec/Dig_Rsor.htm

- ☞ Useful Links of the McGill's Electrical and Computer Engineering Research Groups, Canada, available at:

www.power.ece.mcgill.ca/UsefulLinks.htm

- ☞ Web-page of Warren King, University of Waterloo, Canada, available at:

www.power.uwaterloo.ca/~ewking/

- ☞ Web-page on Open-Source Software for Electricity Market Research, Teaching, and Training, by Leigh Tesfation, Iowa State University, USA.

www.econ.iastate.edu/tesfatsi/ElectricOSS.htm

- ☞ PSAT Tips and Tricks Page by Luigi Vanfretti, Rensselaer Polytechnic Institute, New York, USA:

www.rpi.edu/~vanfrl/psat.html/

- ☞ Web-page of Electrical and Computer Engineering, University of Alberta, Canada:

www.ece.ualberta.ca/~ee433/

- ☞ Web-page of Sheng How Goh, University of Queensland, Australia, available at:

www.itee.uq.edu.au/~shgoh/

- ☞ Web-page of *Moisés Roberto Lanner Carvalho*, Instituto Militar de Engenharia, Brasil, available at:

aquarius.ime.eb.br/~mrlc/

- ☞ Web-page of Clodomiro Unsihuay Vila, Universidad Federal de Itajubá, Brasil, available at:

www.clodomiro.unifei.edu.br/

Appendix J

Letters of Reference

The following list depicts the Institutions that sent me a letter of reference for PSAT.

An electronic copy of the reference letters is available at:

www.uclm.es/area/gsee/Web/Federico/psat.htm

If your University, Institution or Company is using PSAT, please send me a letter of reference. These letters are important for me in order to request funds to my University and, in turn, to keep developing PSAT.



University of Waterloo, Ontario, Canada.



Universidad San Carlos de Guatemala, Guatemala.



Universidad Mariano Gálvez de Guatemala, Guatemala.



University of Campinas (Unicamp), Brazil.



Universidad Centroamericana "José Simeón Cañas", El Salvador.



National Institute of Applied Sciences and Technology, Tunisia.



University of Maryland, USA.



University of New South Wales, Australia.



Federal University of Itajubá, Brazil.



Nanjing Automation Research Institute, China.



Asian Institute of Technology, Thailand.



University of Kocaeli, Turkey.



University of Genoa, Italy.



Centro de Investigaciones Eléctricas -Electrónicas, Perú.



Indian Institute of Technology, Kanpur, India.



Federal University of Rio de Janeiro, Brazil.



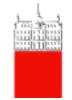
Indian Institute of Technology, Roorkee, India.



University of Kebangsaan, Malaysia.



Indian Institute of Technology, Bombay, India.



University of Ljubljana, Slovenia.



Rensselaer Polytechnic Institute, USA.



Federal University of Pernambuco, Brazil.

Inst. Sup. Politécnico "J. A. Echeverría", La Habana, Cuba.



Open Secure Energy Control Systems, Silver Spring, Maryland.



Politecnico di Bari, Bari, Italy.



Science Applications International Corporation, Huntsville, Alabama.

Appendix K

GNU General Public License

Version 3, 29 June 2007

Copyright © 2007 Free Software Foundation, Inc. <http://fsf.org/>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program—to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

Terms and Conditions

0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without

modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- (a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- (b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to “keep intact all notices”.
- (c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- (d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- (a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- (b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable

physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

- (c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- (d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- (e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- (a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- (b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

- (c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- (d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- (e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- (f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right

to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to

you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR

OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

<one line to give the program’s name and a brief idea of what it does.>

Copyright (C) <textyear> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
```

```
This program comes with ABSOLUTELY NO WARRANTY; for details type 'show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type 'show c' for details.
```

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an “about box”.

You should also get your employer (if you work as a programmer) or school, if any, to sign a “copyright disclaimer” for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see

<http://www.gnu.org/licenses/>

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read

<http://www.gnu.org/philosophy/why-not-lgpl.html>

Bibliography

- [1] S. Acevedo, L. R. Linares, J. R. Martí, and Y. Fujimoto, “Efficient HVDC Converter Model for Real Time Transient Simulation,” *IEEE Transactions on Power Systems*, vol. 14, no. 1, pp. 166–171, Feb. 1999.
- [2] V. Akhmatov, H. Knudsen, and A. H. Nielsen, “Advanced Simulation of Windmills in the Electric Power Supply,” *International Journal of Electric Power and Energy Systems*, vol. 22, no. 6, pp. 421–434, Aug. 2000.
- [3] J. Allemong, L. Radu, and A. Sasson, “A Fast and Reliable Estimation Algorithm for AEP’s New Control Center,” *IEEE Transactions on Power Apparatus and Systems*, vol. 101, no. 4, pp. 933–944, Apr. 1982.
- [4] O. Alsac, J. Bright, M. Prais, and B. Stott, “Further Developments in LP-based Optimal Power Flow,” *IEEE Transactions on Power Systems*, vol. 5, no. 3, pp. 697–711, Aug. 1990.
- [5] P. M. Anderson and A. Bose, “Stability Simulation of Wind Turbine Systems,” *IEEE Transactions on Power Apparatus and Systems*, vol. 102, no. 12, pp. 3791–3795, Dec. 1983.
- [6] P. M. Anderson and A. A. Fouad, *Power System Control and Stability*. Ames, Iowa: The Iowa State University Press, 1977.
- [7] S. Arabi, P. Kundur, and J. H. Sawada, “Appropriate HVDC Transmission Simulation Models for Various Power System Stability Studies,” *IEEE Transactions on Power Systems*, vol. 13, no. 4, pp. 1292–1297, Nov. 1998.
- [8] S. Arabi, G. J. Rogers, D. Y. Wong, P. Kundur, and M. G. Lauby, “Small Signal stability Program Analysis of SVC and HVDC in AC Power Systems,” *IEEE Transactions on Power Systems*, vol. 6, no. 3, pp. 1147–1153, Aug. 1991.
- [9] J. Arillaga and C. P. Arnold, *Computer Analysis Power Systems*. New York: John Wiley & Sons, 1990.
- [10] J. Arillaga, C. P. Arnold, J. R. Camacho, and S. Sankar, “AC-DC Load Flow with Unit-Connected Generator-Converter Infeeds,” *IEEE Transactions on Power Systems*, vol. 8, no. 2, pp. 701–706, May 1993.

- [11] T. Baldwin, L. Mili, M. Boisen, and R. Adapa, "Power System Observability with Minimal Phasor Measurement Placement," *IEEE Transactions on Power Systems*, vol. 8, no. 2, pp. 707–715, May 1993.
- [12] W. R. Barcelo and W. W. Lemmon, "Standardized Sensitivity Coefficients for Power System Networks," *IEEE Transactions on Power Systems*, vol. 3, no. 4, pp. 1591–1599, Nov. 1988.
- [13] G. L. Berg, "Power system load representation," *Proceedings of IEE*, vol. 120, pp. 344–348, 1973.
- [14] S. Bhattacharya and H. W. Dommel, "A New Commutation Margin Control Representation for Digital Simulation of HVDC System Transient," *IEEE Transactions on Power Systems*, vol. 3, no. 3, pp. 1127–1132, Aug. 1988.
- [15] R. Billington, S. Aborehaid, and M. Fotuhi-Firuzabad, "Well-Being Analysis for HVDC Transmission Systems," *IEEE Transactions on Power Systems*, vol. 12, no. 2, pp. 913–918, May 1997.
- [16] K. E. Brenan, S. L. Campbell, and L. Petzold, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*. Philadelphia, PA: SIAM, 1995.
- [17] A. Brooke, D. Kendrick, A. Meeraus, R. Raman, and R. E. Rosenthal, *GAMS, a User's Guide*, GAMS Development Corporation, 1217 Potomac Street, NW, Washington, DC 20007, USA, Dec. 1998, available at www.gams.com.
- [18] C. A. Cañizares, "Applications of Optimization to Voltage Collapse Analysis," in *IEEE-PES Summer Meeting*, San Diego, USA, July 1998.
- [19] ——, "Modeling of TCR and VSI Based FACTS Controllers," ENEC, Milan, Italy, Tech. Rep., Dec. 1999.
- [20] ——, "Voltage Stability Assessment: Concepts, Practices and Tools," IEEE/PES Power System Stability Subcommittee, Final Document, Tech. Rep., Aug. 2002, available at <http://www.power.uwaterloo.ca>.
- [21] C. A. Cañizares and F. L. Alvarado, "Point of Collapse Methods and Continuation Methods for Large AC/DC Systems," *IEEE Transactions on Power Systems*, vol. 8, no. 1, pp. 1–8, Feb. 1993.
- [22] ——, "UWPFLOW Program," 2000, university of Waterloo, available at <http://www.power.uwaterloo.ca>.
- [23] C. A. Cañizares, F. L. Alvarado, C. L. DeMarco, I. Dobson, and W. F. Long, "Point of Collapse Methods applied to AC/DC Power Systems," *IEEE Transactions on Power Systems*, vol. 7, no. 2, pp. 673–683, May 1992.
- [24] C. A. Cañizares, H. Chen, and W. Rosehart, "Pricing System Security in Electricity Markets," in *Proc. Bulk Power systems Dynamics and Control-V*, Onomichi, Japan, Sept. 2001.

- [25] C. A. Cañizares, W. Rosehart, A. Berizzi, and C. Bovo, "Comparison of Voltage Security Constrained Optimal Power flow Techniques," in *Proc. 2001 IEEE-PES Summer Meeting*, Vancouver, BC, Canada, July 2001.
- [26] C. A. Cañizares, W. Rosehart, and V. Quintana, "Costs of Voltage Security in Electricity Markets," in *Proc. 2001 IEEE-PES Summer Meeting*, Seattle, WA, USA, July 2000.
- [27] M. S. Castro, H. M. Ayres, and L. C. P. da Silva, "Impacts of the SSSC Control Modes on Small-Signal and Transient Stability of a Power System," *Electric Power System Research*, 2006, in press, available on-line since February 2006.
- [28] M. S. Castro, H. M. Ayres, I. Kopcak, V. F. da Costa, and L. C. P. da Silva, "A Influência do Modo de Operação do SSSC na Estabilidade de Ângulo de Sistemas Elétricos de Potência," *Revista Controle e Automação*, vol. 13, no. 3, pp. 347–360, July-September 2007, available at www.scielo.br.
- [29] H. B. Çetinkaya, S. Öztürk, and B. Alboyaci, "Eigenvalues Obtained with Two Simulation Packages (SIMPOW and PSAT) and Effects of Machine Parameters on Eigenvalues," in *Proc. of Melecon 2004*, Dubrovnik, Croatia, May 2004.
- [30] ——, "Machine Parameters and Orders of Machine Impacts on Eigenvalues and Simulations in two Software Packages SIMPOW and PSAT," in *Proc. of IEEE SoutheastCon*, Greensboro, North Carolina, Mar. 2004.
- [31] S. J. Chapman, *Electric Machinery and Power System Fundamentals*. New York: McGraw Hill, 2002.
- [32] A. H. L. Chen, C. O. Nwankpa, H. G. Kawatny, and X. ming Yu, "Voltage Stability Toolbox: An Introduction and Implementation," in *Proc. NAPS'96*, MIT, Nov. 1996.
- [33] J. Chow, *Power System Toolbox Version 2.0: Dynamic Tutorial and Functions*, Cherry Tree Scientific Software, RR-5 Colborne, Ontario K0K 1S0, 1991-1997.
- [34] ——, *Power System Toolbox Version 2.0: Load Flow Tutorial and Functions*, Cherry Tree Scientific Software, RR-5 Colborne, Ontario K0K 1S0, 1991-1999.
- [35] J. Chow, F. F. Wu, and J. Momoh, *Applied Mathematics for Restructured Electric Power Systems*. Springer-Verlag, 2005, reference in Chapter 8, *In-stability Monitoring and Control of Power Systems*, by E. H. Abed, M. A. Hassouneh and M. S. Saad, from page 171.
- [36] J. H. Chow and K. W. Cheung, "A Toolbox for Power System Dynamics and Control Engineering Education and Research," *IEEE Transactions on Power Systems*, vol. 7, no. 4, pp. 1559–1564, Nov. 1992.

- [37] L. Chun, J. Qirong, X. Xiaorong, and W. Zhonghong, "Rule-based control for STATCOM to increase power system stability," in *Power System Technology, Proceedings 1998 International Conference on POWERCON*, Aug. 1998, pp. 372–376.
- [38] K. Clements, G. Krumpholz, and P. Davis, "Power System Observability: A Practical Algorithm Using Network Topology," *IEEE Transactions on Power Apparatus and Systems*, vol. 99, no. 4, pp. 1534–1542, July/Aug. 1980.
- [39] ——, "Power System State Estimation Residual Analysis: An Algorithm Using Network Topology," *IEEE Transactions on Power Apparatus and Systems*, vol. 100, no. 4, pp. 1779–1787, Apr. 1981.
- [40] A. J. Conejo and J. M. Arroyo, "Optimal response of a thermal unit to an electricity spot market," *IEEE Transactions on Power Systems*, vol. 15, pp. 1098–1104, Aug. 2000.
- [41] ——, "Multiperiod Auction for a Pool-based electricity market," *IEEE Transactions on Power Systems*, vol. 17, no. 4, pp. 1225–1231, Nov. 2002.
- [42] A. D. Del Rosso and C. A. Negri, "Influencia del Modelado de la Carga en la Evaluación de la Estabilidad Transitoria en Sistemas de Potencia," in *Undécimo Encuentro Regional Iberoamericano del Cigré, XI ERIAC*, Hernandarias, Paraguay, May 2005.
- [43] G. B. Denegri, M. Invernizzi, and F. Milano, "A Security Oriented Approach to PMU Positioning for Advanced Monitoring of a Transmission Grid," in *Proc. of PowerCon 2002*, Kunming, China, Oct. 2002.
- [44] A. S. Drud, *GAMS/CONOPT*, ARKI Consulting and Development, Bagsvaerdvej 246A, DK-2880 Bagsvaerd, Denmark, 1996, the documentation is available at <http://www.gams.com/>.
- [45] S. El-Kashlan, M. Abdel-Rahman, H. El-Desouki, and M. Mansour, "Voltage Stability of Wind Power Systems using Bifurcation Analysis," *Power and Energy Systems*, vol. 468, 2005.
- [46] European Wind Energy Association, "Wind Force 12-A Blueprint to Achieve 12% of the World's Electricity from Wind Power by 2020," EWEA, Tech. Rep., 2001, 56 pages.
- [47] M. C. Ferris, *MATLAB and GAMS: Interfacing Optimization and Visualization Software*, Computer Sciences Department, University of Wisconsin-Madison, Aug. 1999, available at <http://www.cs.wisc.edu/math-prog/matlab.html>.
- [48] I. S. R. T. Force, "First Benchmark Model for Computer Simulation of Sub-synchronous Resonance," *IEEE Transactions on Power Apparatus and Systems*, vol. 96, no. 5, pp. 1565–1572, Sept./Oct. 1977.

- [49] B. S. Gisin, M. V. Obessis, and J. V. Mitsche, "Practical Methods for Transfer Limit Analysis in the Power Industry Deregulated Environment," in *Proc. PICA IEEE International Conference*, 1999, pp. 261–266.
- [50] C. A. Gross, *Power System Analysis*. Second Edition, John Wiley & Sons, 1986.
- [51] I. E. Grossmann, J. Viswanathan, A. Vecchietti, R. Raman, and E. Kalvelagen, *GAMS/DICOPT: A Discrete Continuous Optimization Package*, Engineering Research Design Center, Carnegie Mellon University, Pittsburg, PA, 2002, the documentation is available at <http://www.gams.com/>.
- [52] A. M. Haidar, A. Mohamed, and A. Hussain, "Power System Vulnerability Assessment Considering a New Index Based on Power System Loss," in *International Conference on Energy and Environment*, Bangi, Malaysia, Aug. 2006.
- [53] M. H. Haque, "Improvement of first swing stability limit by utilizing full benefit of shunt FACTS devices," *IEEE Transactions on Power Systems*, vol. 19, no. 4, pp. 1894–1902, 2004.
- [54] C. J. Hatziadoniu, A. A. Lobo, F. Pourboghrat, and M. Daneshdoost, "A Simplified Dynamic Model of Grid-Connected Fuel-Cell Generators," *IEEE Transactions on Power Systems*, vol. 17, no. 2, pp. 467–473, Apr. 2002.
- [55] D. J. Hill, "Nonlinear Dynamic Load Models with Recovery for Voltage Stability Studies," *IEEE Transactions on Power Systems*, vol. 8, no. 1, pp. 166–176, Feb. 1993.
- [56] G. Hingorani and L. Gyugyi, *Understanding FACTS: Concepts and Technology of Flexible AC Transmission Systems*. IEEE Press, 1999.
- [57] P. Hirsch, *Extended Transient-Midterm Stability Program (ETMSP) Ver. 3.1: User's Manual*, EPRI, TR-102004-V2R1, May 1994.
- [58] M. Huneault and F. D. Galiana, "A Survey of the Optimal Power Flow Literature," *IEEE Transactions on Power Systems*, vol. 6, no. 2, pp. 762–770, May 1991.
- [59] IEC 61400-1, "Wind Turbines - Part 1: Design Requirements," International Electrotechnical Commission, Geneva, Switzerland, Tech. Rep., Aug. 2005.
- [60] M. Ilić and J. Zaborszky, *Dynamic and Control of Large Electric Power Systems*. New York: Wiley-Interscience Publication, 2000.
- [61] G. D. Irisarri, X. Wang, J. Tong, and S. Mokhtari, "Maximum Loadability of Power Systems using Interior Point Nonlinear Optimization Method," *IEEE Transactions on Power Systems*, vol. 12, no. 1, pp. 162–172, Feb. 1997.

- [62] S. Iwamoto and Y. Tamura, "A Load Flow Calculation Method for Ill-Conditioned Power Systems," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-100, no. 4, pp. 1736–1743, Apr. 1981.
- [63] A. W. N. Izzri, A. Mohamed, and I. Yahya, "A New Method of Transient Stability Assessment in Power Systems Using LS-SVM," in *The 5th Student Conference on Research and Development - SCOReD 2007*, Malaysia, Dec. 2007.
- [64] K. Jimma, A. Tomac, C. C. Liu, and K. T. Vu, "A Study of Dynamic Load Models for Voltage Collapse Analysis," in *Proc. Bulk Power Syst. Voltage Phenomena II - Voltage Stability and Security*, Aug. 1991, pp. 423–429.
- [65] S. V. N. Jithin-Sundar and M. Reshma, "Utilization of Controlled Shunt Reactor in a 400 kV Interconnected Network," *International Journal of Emerging Electric Power Systems*, vol. 2, no. 1, 2005.
- [66] D. Karlsson and D. J. Hill, "Modelling and Identification of Nonlinear Dynamic Loads in Power Systems," *IEEE Transactions on Power Systems*, vol. 9, no. 1, pp. 157–166, Feb. 1994.
- [67] V. Knyazkin, L. Söder, and C. Cañizares, "Control Challenges of Fuel Cell-Driven Distributed Generation," in *Proc. of the IEEE/PES General Meeting*, Toronto, July 2003.
- [68] D. Koesrindartoto, J. Sun, and L. Tesfatsion, "An Agent-Based Computational Laboratory for Testing the Economic Reliability of Wholesale Power Market Designs," in *IEEE PES Conference Proceedings*, San Francisco, California, June 2005.
- [69] I. Kopcak, "Uma Plataforma Unificada para Análise de Estabilidade de Sistemas Elétricos de Potência," Ph.D. dissertation, Faculdade de Engenharia Elétrica e de Computação, Unicamp, Barão Geraldo, São Paulo, Brazil, 2007.
- [70] I. Kopcak, H. M. Ayres, L. C. P. da Silva, and V. F. da Costa, "Análise Simultânea das Margens de Estabilidade Dinâmica e Estática de Sistemas Elétricos de Potência," in *Anais do VI CLAGTEE 2007 Congresso Latinoamericano de Geração e Transmissão de Energia Elétrica*, Valparaíso, Chile, Nov. 2007, pp. 1–6.
- [71] I. Kopcak, V. F. da Costa, and L. C. P. da Silva, "A Generalized Power Flow Method Including the Steady State Characteristic of Dynamic Devices," in *Proceedings of IEEE 2007 PowerTech Conference*, Lausanne, Switzerland, July 2007, pp. 1–6.
- [72] I. Kopcak, L. C. P. da Silva, and V. F. da Costa, "A Generalized Load Flow Method to Assess Steady-State Equilibrium of Power Systems," in *Proceedings of IEEE PES PowerAfrica 2007 Conference and Exposition*, Johannesburg, South Africa, July 2007, pp. 1–7.

- [73] P. Kumkratug and M. Haque, "Versatile model of a unified power flow controller a simple power system," *IEE Proceedings on Generation, Transmission and Distribution*, vol. 150, pp. 155–161, Mar. 2003.
- [74] P. Kumkratug and M. H. Haque, "Improvement of stability region and damping of a power system by using SSSC," in *IEEE Power Engineering Society General Meeting*, vol. 3, Denver, 2003.
- [75] P. Kundur, *Power System Stability and Control*. New York: McGraw Hill, 1994.
- [76] M. Larsson, "ObjectStab, An Educational Tool for Power System Stability Studies," *IEEE Transactions on Power Systems*, vol. 19, no. 1, pp. 56–63, Feb. 2004.
- [77] M. Madrigal, "Optimization Model and Techniques for Implementation and Pricing of Electricity Markets," Ph.D. dissertation, University of Waterloo, Waterloo, ON, Canada, 2000.
- [78] M. Madrigal and V. H. Quintana, "Optimal Day-ahead Network-constrained Power System's Market Operations Planning using an Interior Point Method," in *IEEE Canadian Conference on Electrical and Computer Eng.*, vol. 1, May 1998, pp. 388–401.
- [79] J. Mahseredjian and F. Alvarado, "Creating an Electromagnetic Transient Program in MATLAB: MatEMTP," *IEEE Transactions on Power Delivery*, vol. 12, no. 1, pp. 380–388, Jan. 1997.
- [80] Z. J. Meng and P. L. So, "A current injection UPFC model for enhancing power system," in *IEEE Power Engineering Society Winter Meeting*, vol. 2, 2000, pp. 1544–1549.
- [81] R. Mihalic and U. Gabrijel, "A structure-preserving energy function for a static series synchronous compensator," *IEEE Transactions on Power Systems*, vol. 19, no. 3, pp. 1501–1507, 2004.
- [82] F. Milano, "Continuous Newton's Method for Power Flow Analysis," accepted for publication on the IEEE Transactions on Power systems, August 2008.
- [83] ——, "Pricing System Security in Electricity Market Models with Inclusion of Voltage Stability Constraints," Ph.D. dissertation, University of Genova, Genova, Italy, 2003, available at www.uclm.es/area/gsee/Web/Federico.
- [84] ——, "A Graphical and Open-Source Matlab-GAMS Interface for Electricity Markets Models," in *Noveno Congreso Hispano-Luso de Ingeniería Eléctrica, CHLIE*, Marbella, Spain, June 2005.
- [85] ——, "An Open Source Power System Analysis Toolbox," *IEEE Transactions on Power Systems*, vol. 20, no. 3, pp. 1199–1206, Aug. 2005.

- [86] F. Milano, C. A. Cañizares, and A. J. Conejo, "Sensitivity-based Security-constrained OPF Market Clearing Model," *IEEE Transactions on Power Systems*, vol. 20, no. 4, pp. 2051–2060, Nov. 2005.
- [87] F. Milano, C. A. Cañizares, and M. Invernizzi, "Multi-objective Optimization for Pricing System Security in Electricity Markets," *IEEE Transactions on Power Systems*, vol. 18, no. 2, May 2003.
- [88] F. Milano, L. Vanfretti, and J. C. Morataya, "An Open Source Power System Virtual Laboratory: The PSAT Case and Experience," *IEEE Transactions on Education*, vol. 51, no. 1, pp. 17–23, Feb. 2008.
- [89] J. Milias-Argitis, T. Zacharias, C. Hatzadoniu, and G. D. Galanos, "Transient Simulation of Integrated AC/DC Systems, Part I: Converter Modeling and Simulation," *IEEE Transactions on Power Systems*, vol. 3, no. 1, pp. 166–172, Feb. 1988.
- [90] ———, "Transient Simulation of Integrated AC/DC Systems, Part II: System Modeling and Simulation," *IEEE Transactions on Power Systems*, vol. 3, no. 1, pp. 173–179, Feb. 1988.
- [91] Y. Mitani and K. Tsuji, "Bifurcations Associated with Sub-Synchronous Resonance," *IEEE Transactions on Power Systems*, vol. 10, no. 4, pp. 1471–1478, Nov. 1995.
- [92] Y. Mitani, K. Tsuji, M. Varghese, F. Wu, and P. Varaiya, "Bifurcation Associated with Sub-Synchronous Resonance," *IEEE Transactions on Power Systems*, vol. 13, no. 1, pp. 139–144, Feb. 1998.
- [93] J. A. Monmoh, S. X. Guo, E. C. Ogbuobiri, and R. Adapa, "The Quadratic Interior Point Method solving Power System Optimization Problems," *IEEE Transactions on Power Systems*, vol. 9, no. 3, pp. 1327–1336, Aug. 1994.
- [94] A. Monticelli and A. Garcia, "Fast Decoupled State Estimators," *IEEE Transactions on Power Systems*, vol. 5, no. 2, pp. 556–564, May 1990.
- [95] A. L. Motto, F. D. Galiana, A. J. Conejo, and J. M. Arroyo, "Network-constrained multiperiod auction for a pool-based electricity market," *IEEE Transactions on Power Systems*, vol. 17, no. 3, pp. 646–653, Aug. 2002.
- [96] B. A. Murtagh, M. A. Saunders, W. Murray, P. E. Gill, R. Raman, and E. Kalvelagen, *GAMS/MINOS: A Solver for Large-Scale Nonlinear Optimization Problems*, 2002, available at www.gams.com.
- [97] R. Natesan and G. Radman, "Effects of STATCOM, SSSC and UPFC on Voltage Stability," in *Proceedings of the Thirty-Sixth Southeastern Symposium on System Theory*, Atlanta, Georgia, Mar. 2004, pp. 546–550.
- [98] M. Noroozian, L. Angquist, M. Ghandhari, and G. Andersson, "Improving power system dynamics by series-connected FACTS devices," *IEEE Transactions on Power Delivery*, vol. 12, pp. 1635–1641, Oct. 1997.

- [99] C. Nwankpa, *Voltage Stability Toolbox, version 2*, Center for Electric Power Engineering, Drexel University, 2002, available at power.ece.drexel.edu/research/VST/vst.htm.
- [100] J. Padullés, G. W. Ault, and J. R. McDonald, "An Integrated SOFC Plant Dynamic Model for Power Systems Simulation," *International Journal of Power Sources*, vol. 86, pp. 495–500, 2000.
- [101] S. N. Pandey, S. Tapaswi, and L. Srivastava, "Nodal Congestion Price Estimation in Spot Power Market Using Artificial Neural Network," *IET Generation, Transmission & Distribution*, vol. 2, no. 2, pp. 280–290, 2008.
- [102] H. A. Panofsky and J. A. Dutton, *Atmospheric Turbulence; Models and Methods for Engineering Applications*. New York: John Wiley & Sons, 1984.
- [103] L. A. S. Pilotto, M. Roitman, and J. E. R. Alves, "Digital Control HVDC Converters," *IEEE Transactions on Power Systems*, vol. 4, no. 2, pp. 704–711, May 1989.
- [104] V. H. Quintana and G. L. Torres, "Nonlinear Optimal Power Flow in Rectangular Form via Primal-Dual Logarithmic Barrier Interior Point Method," University of Waterloo, Tech. Rep., 1996, technical Report 96-08.
- [105] A. H. M. A. Rahim, S. A. Al-Baiyat, and H. M. Al-Maghribi, "Robust damping controller design for a static compensator," *IEE Proceedings on Generation, Transmission and Distribution*, vol. 149, pp. 491–496, July 2002.
- [106] I. C. Report, "Reader's Guide to SSR," *IEEE Transactions on Power Apparatus and Systems*, vol. 7, no. 2, pp. 150–157, Feb. 1992.
- [107] I. P. S. E. C. Report, "Terms, Definitions & Symbols for Subsynchronous Oscillations," *IEEE Transactions on Power Apparatus and Systems*, vol. 104, no. 6, pp. 1326–1334, June 1985.
- [108] W. Rosehart, C. A. Cañizares, and V. H. Quintana, "Optimal Power Flow Incorporating Voltage Collapse Constraints," in *Proc. 1999 IEEE-PES Summer Meeting*, Edmonton, Alberta, July 1999.
- [109] N. Rostamkolai, C. A. Wegner, R. J. Piwko, H. Elahi, M. A. Eitzmann, G. Garzi, and P. Tietz, "Control Design of Santo Tomé Bak-to-Back HVDC Link," *IEEE Transactions on Power Systems*, vol. 8, no. 3, pp. 1250–1256, Aug. 1993.
- [110] M. Sato, K. Yamaji, and M. Sekita, "Development of a Hybrid Margin Angle Controller for HVDC Continuous Operation," *IEEE Transactions on Power Systems*, vol. 11, no. 4, pp. 1792–1798, Nov. 1996.
- [111] P. W. Sauer and M. A. Pai, *Power System Dynamics and Stability*. Upper Saddle River, New Jersey: Prentice Hall, 1998.

- [112] F. Scheppele, J. Wildes, and D. Rom, "Power System Static-State Estimation, Part I,II, III," *IEEE Transactions on Power Apparatus and Systems*, vol. 89, no. 1, pp. 120–135, Jan. 1970.
- [113] K. Schoder, A. Feliachi, and A. Hasanović, "PAT: A Power Analysis Toolbox for MATLAB/Simulink," *IEEE Transactions on Power Systems*, vol. 18, no. 1, pp. 42–47, Feb. 2003.
- [114] R. Seydel, *Practical Bifurcation and Stability Analysis: From Equilibrium to Chaos*. New York: Second Edition, Springer-Verlag, 1994.
- [115] G. B. Sheblé, *Computational Auction Mechanism for Restructured Power Industry Operation*. Boston: Kluwer Academic Publishers, 1998.
- [116] T. Shome, A. M. Gole, D. P. Brandt, and R. J. Hamlin, "Adjusting Converter Control for Parallel DC Converters Using a Digital Transient Simulation Program," *IEEE Transactions on Power Systems*, vol. 5, no. 1, pp. 12–19, Feb. 1990.
- [117] E. Simiu and R. H. Scanlan, *Wind Effects on Structures; an Introduction to Wind Engineering*. New York: John Wiley & Sons, 1986.
- [118] J. G. Slootweg, "Wind Power: Modelling and Impact on Power System Dynamics," Ph.D. dissertation, Delft University of Technology, Delft, Netherlands, 2003.
- [119] Y. H. Song and A. T. Johns, *Flexible AC Transmission System (FACTS)*. London: The Institute of Electrical Engineers, 1999.
- [120] B. Stott, "Review of load-Flow Calculation Methods," *Proceedings of the IEEE*, vol. 62, no. 7, pp. 916–929, July 1974.
- [121] B. Stott and O. Alsac, "Fast Decoupled Load Flow," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-93, no. 3, pp. 859–869, June 1974.
- [122] B. Stott, J. L. Marino, and O. Alsac, "Review of Linear Programming Applied to Power System Rescheduling," in *Proc. PICA IEEE International Conference*, 1979, pp. 142–154.
- [123] G. Sybille, *SimPowerSystems User's Guide, Version 4*, published under sub-license from Hydro-Québec, and The MathWorks, Inc., Oct. 2004, available at <http://www.mathworks.com>.
- [124] A. Y. Takahata, "Segurança de Sistema de Potência sob Pequenas Perturbações Considerando Incertezas," Ph.D. dissertation, Universidade Federal do Rio de Janeiro, COPPE, Rio de Janeiro, Brazil, 2008.
- [125] C. W. Taylor and S. Lefebvre, "HVDC Controls for System Dynamic Performance," *IEEE Transactions on Power Systems*, vol. 6, no. 2, pp. 743–752, May 1991.

- [126] J. Thorp, A. Phadke, and K. Karimi, "Real Time Voltage Phasor Measurement for Static State Estimation," *IEEE Transactions on Power Apparatus and Systems*, vol. 104, no. 11, pp. 3908–3103, Nov. 1985.
- [127] W. F. Tinney and C. E. Hart, "Power Flow Solution by Newton's Method," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-86, pp. 1449–1460, Nov. 1967.
- [128] G. L. Torres and V. H. Quintana, "Introduction to Interior-Point Methods," in *IEEE PICA*, Santa Clara, CA, May 1999.
- [129] R. van Amerongen, "A General-Purpose Version of the Fast Decoupled Load-flow," *IEEE Transactions on Power Systems*, vol. 4, no. 2, pp. 760–770, May 1989.
- [130] L. Vanfretti and F. Milano, "Application of the PSAT, an Open Source Software, for Educational and Research Purposes," in *IEEE PES General Meeting*, Tampa, USA, June 2007.
- [131] S. Venkataraman, M. H. Khammash, and V. Vittal, "Analysis and Synthesis of HVDC Controls for Robust Stability of Power Systems," *IEEE Transactions on Power Systems*, vol. 10, no. 4, pp. 1933–1938, Nov. 1995.
- [132] C. D. Vournas, E. G. Potamianakis, C. Moors, and T. V. Cutsem, "An Educational Simulation Tool for Power System Control and Stability," *IEEE Transactions on Power Systems*, vol. 19, no. 1, pp. 48–55, Feb. 2004.
- [133] K. T. Vu, C.-C. Liu, C. W. Taylor, and K. M. Jimma, "Voltage instability: Mechanisms and Control Strategy," *Proceedings of the IEEE*, vol. 83, no. 11, pp. 1442–1455, Nov. 1995.
- [134] C. Wang and S. M. Shahidehpour, "Ramp-rate limits in unit commitment and economic dispatch incorporating roto fatigue effect," *IEEE Transactions on Power Systems*, vol. 9, pp. 1539–1545, Aug. 1994.
- [135] O. Wasynczuk, D. T. Man, and J. P. Sullivan, "Dynamic Behavior of a Class of Wind Turbine Generators during Random Wind Fluctuations," *IEEE Transactions on Power Apparatus and Systems*, vol. 100, no. 6, pp. 2837–2845, June 1981.
- [136] Working Group H-7 of the Relaying Channels Subcommittee of the IEEE Power System Relaying Committee, "Synchronized Sampling and Phasor Measurements for Relaying and Control," *IEEE Transactions on Power Delivery*, vol. 9, no. 1, pp. 442–452, Jan. 1994.
- [137] Working Group on a Common Format for Exchange of Solved Load Flow Data, "Common Format for the Exchange of Solved Load Flow Data," *IEEE Transactions on Power Apparatus and Systems*, vol. 92, no. 6, pp. 1916–1925, Nov./Dec. 1973.

- [138] K. Xie, Y.-H. Song, J. Stonham, E. Yu, and G. Liu, "Decomposition Model and Interior Point Methods for Optimal Spot Pricing of Electricity in Deregulation Environments," *IEEE Transactions on Power Systems*, vol. 15, no. 1, pp. 39–50, Feb. 2000.
- [139] W. Xu and Y. Mansour, "Voltage Stability Analysis Using Generic Dynamic Load Models," *IEEE Transactions on Power Systems*, vol. 9, no. 1, Feb. 1994.
- [140] Yao-Nan-Yu, *Electric Power System Dynamic*. New York: Academic Press, 1983.
- [141] E. M. Yap, M. Al-Dabbagh, and P. C. Thum, "UPFC Controller in Mitigating Line Congestion for Cost-efficient Power Delivery," in *The 7th International Power Engineering Conference, IPEC 2005*, Singapore, November 29-December 2, 2005.
- [142] M. Zhou, "An ODM Reference Implementation," available at <http://sites.google.com/a/interpss.org/interpss/Home/ieee-pes-oss/reference-implementation>.
- [143] ——, "ODM Data Structure and Relationship," available at <http://sites.google.com/a/interpss.org/interpss/Home/ieee-pes-oss/data-structure-relationship>.
- [144] ——, "ODM Schema Specification," available at <http://sites.google.com/a/interpss.org/interpss/Home/ieee-pes-oss/xml-schema-specification>.
- [145] W. Zhu, R. Mohler, R. Spee, W. Mittelstadt, and D. Maratukulam, "Hopf Bifurcations in a SMIB Power System with SSR," *IEEE Transactions on Power Systems*, vol. 11, no. 3, pp. 1579–1584, Aug. 1996.
- [146] Y. Zhu and K. Tomsovic, "Development of Models for Analyzing the Load-Following Performance of Microturbines and Fuel Cells," *Electric Power System Research*, vol. 62, pp. 1–11, 2002.
- [147] R. D. Zimmerman and D. Gan, *Matpower, Documentation for version 2*, Power System Engineering Research Center, Cornell University, 1997, available at <http://www.pserc.cornell.edu/matpower/matpower.html>.

Index

Symbols

LATEX, 15, 26, 37, 66, 310, 366, 371, 388, 390, 393, 394

A

Area, 113, **124–125**, 381

Asynchronous machine, *see* Induction machine

Automatic voltage regulator, *see* AVR
AVR, 4, 22, 25, 93, 161, 167, 179, **184–189**, 285, 287, 290, 404

type I, 185

type II, 186

type III, 187

B

Bus, **113–114**, 279, 280, 282

Bus frequency, 25, **141–142**

C

CAC, 4, 179, 197, **197**, 287

CC, 4, 179, 197, **199**, 287

Center of inertia, *see* COI

Central Area Controller, *see* CAC

CEPEL, 301, 389, 394, 399

CESI, 303, 389, 394

Chapman's format, 301, 389, 394

Cluster Controller, *see* CC

COI, 86, 166, **171–172**, 365, 382

Constant power generator, *see* PQ generator

Constant power load, *see* PQ load

Constant Speed Wind Turbines, *see* CSWT

Continuation Power Flow, *see* CPF

Continuous Newton's method, 34

CPF, 3, 6, 15, 16, 22–26, 44, 47, 48, **49–58**, 115, 127, 133, 267, 268,

270, 305, 317–319, 345, 346, 348, 351, 352, 374, 381, 388

C SWT, 4, **238–241**

CygWin, 345, 346

CYME, 301, 389, 394, 399

CYMFLOW, *see* CYME

D

DDSG, 4, **248–250**

Demand, 47, 54, 128, 131–132, 283, 348, 381

Demand profile, 132–134, 381

DFIG, 4, **243–246**

DIgSILENT, 301, 389, 394, 399

Direct Drive Synchronous Generator, *see* DDSG

Doubly Fed Induction Generator, *see* DFIG

Dynamic shaft, **253–254**, 288

E

EPRI, 301, 303, 389, 394

Euler's method, 83, **84**, 368

Eurostag, 303, 389, 394

Excel, 15, 26, 37, 66, 310, 366, 390

Excitation, *see* AVR

Exponential recovery load, 4, 147, **153–154**, 283

F

FACTS, 4, 16, 209, 267, 345, 346, 382

Fast Decoupled Power Flow, *see* FDPF

FDPF, 31, **32–33**, 368

Flexible ac transmission system, *see* FACTS

FlowDemo.net, 303, 389, 394, 399

Frequency dependent load, 4, 147, **150**, 283

- Frequency regulation, *see* TG
 Fuel cell, *see* Solid oxide fuel cell
- G**
 GAMS, 4, 7, 16, 26, 128, 132, 283, 317, 319, **329–338**, 371, 384, 391, 394, 395, 416
 GE, 303, 389, 399
 GNU GPL, v, 323, 391, 395, **431–443**
 GNU Linux, *see* Linux
 GNU Octave, 3, 9, 22, 25, 87, **323–325**, 326, 367, 368, 389, 412
- H**
 High voltage dc transmission system, *see* HVDC
 HTML, 15, 366, 390
 HVDC, 4, 16, 23, 209, **225–228**, 345, 346, 382
- I**
 IEEE, 16, 54, 56, 78, 92, 184, 277, 301, 303, 348, 389, 394, 400, 406, 409, 413
 Induction machine, 161, **174–178**, 382
 double cage, 177
 mechanical model, 176
 order I, 176
 order III, 176
 order V, 177
 single cage, 176
 Infinite bus, *see* Slack generator
 INPTC1, 303, 389, 394
 Interior Point Method, *see* IPM
 InterPSS, 303, 389, 399
 IPM, 61, **61–62**
 Iwamoto's method, 34, 368
- J**
 Jimma's load, 4, 24, 147, **156–157**, 283
- L**
 Line, *see* Transmission line
 Linux, 23, 310, 323, 330, 345, 416
 LMP, 44, 69, 305, 338, 343, 378
 Load tap changer, *see* Tap changer
 Locational Marginal Price, *see* LMP
- LTC, *see* Tap Changer
- M**
 Matlab, v, 3, 6, 9–12, 14, 22, 24–26, 32, 81, 92, 93, 109, 232, 291, 296, 301, 305, 315, 316, 318, 319, 323, 324, 329, 330, 332, 346, 348, 355, 361, 367, 372, 387, 393–395, 397, 411–413, 417
 Matpower, 303, 389, 399
 Merhotra's predictor-corrector, 61, 378
 Mexican hat wavelet, 22, 231, 233, **237**
 Mixed load, 4, 24, 147, **157–158**, 283
- N**
 NCP, 44, 69, 305
 NEPLAN, 24, 303, 389, 400
 Newton's direction, 61, 378
 Newton-Raphson's method, 31, **31–32**, 33, 36, 48, 53, 83, 84, 86, 95, 368
 Nodal Congestion Price, *see* NCP
- O**
 Octave, *see* GNU Octave
 ODM, 16, 23, 301, 303, 389, 394
 OLTC, *see* Tap Changer
 OPF, 3, 6, 15, 16, 23, 25, 26, 47, **61–70**, 115, **127–136**, 267, 268, 270, 283, 305, 317–320, 330, 331, 335, 338, 374, 376–378, 381, 384, 385, 388, 391, 394, 414
 Optimal Power Flow, *see* OPF
 Overexcitation limiter, *see* OXL
 OXL, 4, 179, **193–195**, 285
- P**
 Pareto's set, 70, 319, 331, 378, 385, 388
 PCFLO, 303, 389, 394
 Phase shifter, *see* Phase shifting transformer
 Phase shifting transformer, 4, 203, **205–206**
 Phasor Measurement Unit, *see* PMU
 PHS, *see* Phase shifting transformer
 PMU, 3, 4, 14, 24, 26, **95–106**, 142–144, 317–319, 372, 379, 382, 391

- POD, 179, **199–201**, 209, 220, 245, 382
 Power system stabilizer, *see* PSS
 Power System Toolbox, *see* PST
 PowerWorld, 303, 389, 395, 400
 PQ generator, **123–124**
 PQ load, 3, 33, 48, 49, 54, **121–123**,
 127, 132, 147, 149, 150, 204,
 269, 279, 280, 283, 320, 346,
 348, 381, 415
 Primary freq. regulation, *see* TG
 Primary voltage regulation, *see* AVR
 PSAP, 303, 389, 394
 PSS, 4, 25, 26, 179, **189–193**, 285
 type I, 190
 type II, 192
 type III, 192
 type IV, 193
 type V, 193
 PSS/E, 24, 25, 303, 389, 395, 400
 PST, 303, 389, 400
 PV generator, 37
 PV bus, *see* PV generator
 PV generator, 3, 24, 33, 36, 48, 49, **121**,
 127, 161, 204, 209, 210, 231,
 253, 258, 260, 269, 279, 280,
 283, 285, 288, 320, 346, 348,
 381
- R**
 Rayleigh's distribution, 234
 Region, 113, **124–125**, 381
- S**
 Secondary voltage control, 4, 179, **197–199**, 287, 345, 346
 Shaft, *see* Subsynchronous resonance,
 Dynamic shaft
 Shunt, 3, 114, **124**, 269, 346, 381
 SIMPOW, 303, 390, 395
 Simulink, 3–5, 7, 9, 12, 14–16, 23, 24, 87,
 111, 159, 237, **267–275**, **279–297**, 301, 303, 305, 316, 320,
 323, 337, 338, 368, 370, 372,
 383, 389, 401, 412, 413
 Slack bus, *see* Slack generator
- Slack generator, 3, 24, 33, 114, **118–120**, 170, 171, 253, 258, 260,
 269, 280, 283, 285, 320, 346,
 381
 Solid oxide fuel cell, **258–263**, 288
 SSCL, 4
 SSSC, 4, 23, 209, **216–219**, 382
 STATCOM, 288
 Statcom, 4, 209, **215–216**, 382
 Static Compensator, *see* Statcom
 Static Synchronous Series Compensator,
 see SSSC
 Static VAr compensator, *see* SVC
 Sub-synchronous resonance, **255–258**
 Supplementary Stabilizing Control, *see*
 POD
 Supply, 47, 54, 127–129, 283, 348, 381
 SVC, 4, 209, **209–212**, 287, 288, 382
 Swing bus, *see* Slack generator
 Synchronous machine, 25
 Synchronous machine, 4, 24, 31, 137, 161,
 161–172, 179, 184, 189, 195,
 253, 285, 382
 electromechanical model, 167
 order II, 167
 order III, 167
 order IV, 168
 order V, type 1, 168
 order V, type 2, 169
 order V, type 3, 169
 order VI, 170
 order VIII, 170
- T**
 Tap changer, 4, 26, 287
 dynamic model, **203–205**
 with embedded load, *see* Voltage de-
 pendent load
 TCSC, 4, 23, 209, **212–215**, 383
 TCUL, *see* Tap Changer
 TG, 4, 161, 179, **179–183**, 285
 type I, 182
 type II, 183
 Thermostatically controlled load, 4, 147,
 154–156, 283

- Thyristor Controlled Series Compensator, *see* TCSC
 Transformer, **116–118**
 Transmission line, **114–116**
 Tsing Hua University, 303, 390, 395
 Turbine governor, *see* TG

U

- UCTE, 303, 390, 395
 ULTC, 151, *see* Tap Changer
 Unified PF controller, *see* UPFC
 Unix, 10, 310, 345
 UPFC, 4, 23, 209, **219–221**, 383
 UWPFLOW, 4, 7, 16, 317, 319, **345–348**, 372, 385, 391, 400

V

- Voltage dependent load, 4, 147, 283
 static model, **147–148**
 with dynamic tap changer, **151–152**
 Voltage regulation, *see* AVR
 Voltage sourced inverter, *see* VSI
 VSI, 209
 VST, 303, 390, 395, 400

W

- WebFlow, 303, 390, 395, 400
 Weibull's distribution, 25, 231, **232–234**, 237
 Wind, 4, 25, **231–250**, 267, 383
 composite model, **234–236**
 constant speed wind turbine, *see* CSWT
 direct drive synchronous generator,
see DDSG
 doubly-fed induction generator, *see* DFIG
 gust, 236
 measurement data, 237
 Mexican hat, *see* Mexican hat wavelet
 ramp, 235
 turbulence, 236
 Weibull's distribution, *see* Weibull's distribution
 Windows, 10, 310, 329, 330, 345, 346, 416, 417

Z

- ZIP load, 4, 147, **148–149**, 283, 286