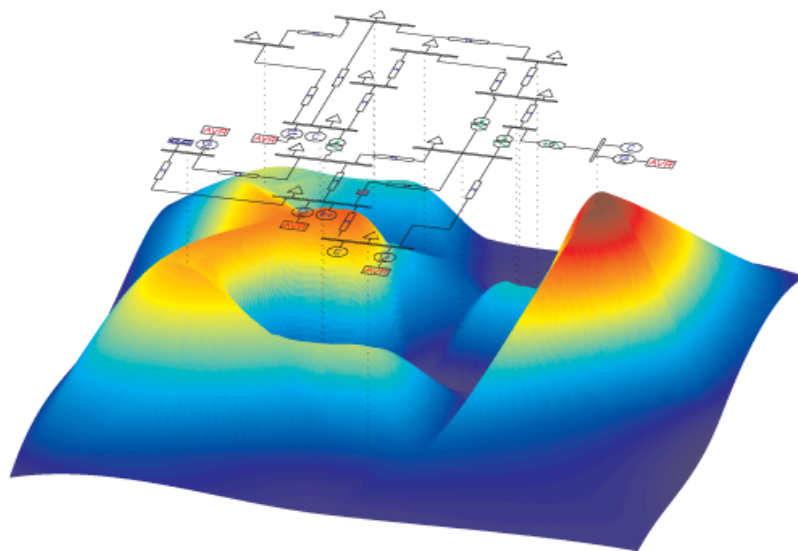


PSAT

Power System Analysis Toolbox
Documentation for PSAT version 2.0.0, February 14, 2008



Federico Milano

Copyright © 2003 - 2008 Federico Milano

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Sections being all sections, with no Front-Cover Texts, and with no Back-Cover Texts. A copy of the license is included in Appendix L entitled “GNU Free Documentation License”.

Ai miei genitori

Note

PSAT is a MATLAB toolbox for static and dynamic analysis and control of electric power systems. I began writing PSAT in September 2001, while I was studying as Ph.D. student at the Università degli Studi di Genova, Italy, and completed the first public version in 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 assistant professor at the Universidad de Castilla-La Mancha, Ciudad Real, 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; for details refer to Appendix K of this documentation or type `gnulicense` at the MATLAB prompt.

PSAT is currently in a early stage of development and its features, structures and data formats may be partially or completely changed in future versions. Be sure to visit often my webpage in order to get the last version:

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

If you find bugs or have any suggestions, please send me an e-mail at:

Federico.Milano@uclm.es

or you can subscribe to the PSAT Forum, which is available at:

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

Acknowledgements

I wish to thank very much Professor C. A. Cañizares for his priceless help, teachings and advises. Thanks also for providing me a webpage and a link to my software in the main webpage of the E&CE Department, University of Waterloo, Canada.

Many thanks to the moderators of the PSAT Forum for spending their time on answering tons of messages: Luigi Vanfretti, Juan Carlos Morataya, Raul Rabinovici, Ivo Šmon, and Zhen Wang.

Thanks to Hugo M. Ayres, Marcelo S. Castro, Alberto Del Rosso, Jasmine, Igor Kopcak, Liu Lin, Lars Lindgren, Marcos Miranda, Juan Carlos Morataya, Difahoui Rachid, Santiago Torres, 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 Manual	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	14
2.7	Displaying Results	14
2.8	Saving Results	15
2.9	Settings	15
2.10	Network Design	16
2.11	Tools	16
2.12	Interfaces	17
3	News	19
3.1	News in version 2.0.0 beta	19
3.2	News in version 1.3.4	20
3.3	News in version 1.3.3	21
3.4	News in version 1.3.2	21
3.5	News in version 1.3.1	21
3.6	News in version 1.3.0	22
3.7	News in version 1.2.2	22
3.8	News in version 1.2.1	23
3.9	News in version 1.2.0	23
3.10	News in version 1.1.0	23
3.11	News in version 1.0.1	23

II	Routines	25
4	Power Flow	27
4.1	Power Flow Solvers	27
4.1.1	Newton-Raphson Method	27
4.1.2	Fast Decoupled Power Flow	28
4.1.3	Distributed Slack Bus Model	29
4.1.4	Initialization of State Variables	30
4.2	Settings	30
4.3	<i>Example</i>	31
5	Bifurcation Analysis	39
5.1	Direct Methods	40
5.1.1	Saddle-Node Bifurcation	40
5.1.2	Limit Induced Bifurcation	40
5.2	Continuation Power Flow	41
5.2.1	Predictor Step	43
5.2.2	Corrector Step	43
5.2.3	N-1 Contingency Analysis	44
5.2.4	Graphical User Interface and Settings	45
5.3	<i>Examples</i>	46
6	Optimal Power Flow	53
6.1	Interior Point Method	53
6.2	OPF Routines	54
6.2.1	Maximization of the Social Benefit	54
6.2.2	Maximization of the Distance to Collapse	54
6.2.3	Multi-Objective Optimization	56
6.2.4	Lagrangian Function	57
6.3	OPF Settings	57
6.4	<i>Example</i>	58
7	Small Signal Stability Analysis	63
7.1	Small Signal Stability Analysis	63
7.1.1	<i>Example</i>	66
7.2	Power Flow Sensitivity Analysis	69
7.2.1	<i>Example</i>	70
7.3	Graphical User Interface	72
8	Time Domain Simulation	75
8.1	Integration Methods	75
8.1.1	Forward Euler Method	76
8.1.2	Trapezoidal Method	76
8.2	Settings	76
8.3	Output Variable Selection	79
8.4	Snapshots	82

8.5	Disturbances	82
8.6	<i>Examples</i>	84
9	PMU Placement	89
9.1	Linear Static State Estimation	89
9.2	PMU Placement Rules	90
9.3	Algorithms	90
9.3.1	Depth First	90
9.3.2	Graph Theoretic Procedure	91
9.3.3	Bisecting Search Method	91
9.3.4	Recursive Security N Algorithm	91
9.3.5	Single Shot Security N Algorithm	92
9.3.6	Recursive and Single-Shot Security $N-1$ Algorithms	92
9.4	PMU Placement GUI and Settings	97
9.4.1	<i>Example</i>	97
III	Models	101
10	Power Flow Data	103
10.1	Bus	103
10.2	Transmission Line	104
10.3	Transformers	105
10.3.1	Two-Winding Transformers	107
10.3.2	Three-Winding Transformers	107
10.4	$V\theta$ and Slack Generator	108
10.5	PV Generator	110
10.6	PQ Load	111
10.7	PQ Generator	113
10.8	Shunt	113
10.9	Area & Regions	114
11	CPF and OPF Data	117
11.1	Generator Supply	118
11.2	Generator Reserve	119
11.3	Generator Power Ramping	120
11.4	Load Demand	121
11.5	Demand Profile	122
11.6	Load Ramping	124
12	Faults & Breakers	127
12.1	Fault	127
12.2	Breaker	127
13	Measurements	131
13.1	Bus Frequency Measurement	131

13.2 Phasor Measurement Unit	132
14 Loads	135
14.1 Voltage Dependent Load	135
14.2 ZIP Load	136
14.3 Frequency Dependent Load	137
14.4 Exponential Recovery Load	138
14.5 Thermostatically Controlled Load	140
14.6 Jimma's Load	142
14.7 Mixed Load	143
14.8 Note on the Use of Non-conventional Loads	144
15 Machines	147
15.1 Synchronous Machine	147
15.1.1 Order II	154
15.1.2 Order III	154
15.1.3 Order IV	154
15.1.4 Order V, Type 1	155
15.1.5 Order V, Type 2	155
15.1.6 Order V, Type 3	156
15.1.7 Order VI	157
15.1.8 Order VIII	157
15.1.9 Center of Inertia	158
15.2 Induction Motor	158
15.2.1 Order I	159
15.2.2 Order III (single cage)	161
15.2.3 Order V (double cage)	162
16 Controls	165
16.1 Turbine Governor	165
16.1.1 TG Type I	166
16.1.2 TG Type II	168
16.2 Automatic Voltage Regulator	169
16.2.1 AVR Type I	169
16.2.2 AVR Type II	170
16.2.3 AVR Type III	171
16.3 Power System Stabilizer	174
16.3.1 Type I	176
16.3.2 Type II	176
16.3.3 Type III	177
16.3.4 Type IV and V	177
16.4 Over Excitation Limiter	177
16.5 Secondary Voltage Control	180
16.6 Power Oscillation Damper	183
17 Regulating Transformers	185

17.1 Under Load Tap Changer	185
17.2 Load Tap Changer With Embedded Load	186
17.3 Phase Shifting Transformer	189
18 FACTS	193
18.1 SVC	194
18.2 TCSC	196
18.3 STATCOM	198
18.4 SSSC	201
18.5 UPFC	204
18.6 HVDC	209
19 Wind Turbines	213
19.1 Wind Models	213
19.1.1 Weibull Distribution	214
19.1.2 Composite Wind Model	216
19.1.3 Measurement Data	217
19.2 Wind Turbines	217
19.2.1 Constant Speed Wind Turbine	219
19.2.2 Doubly Fed Induction Generator	221
19.2.3 Direct Drive Synchronous Generator	227
20 Other Models	231
20.1 Dynamic Shaft	231
20.2 Sub-synchronous Resonance Model	233
20.3 Solid Oxide Fuel Cell	236
IV CAD	243
21 Network Design	245
21.1 Simulink Library	245
21.2 Extracting Data from Simulink Models	245
21.3 Displaying Results in Simulink Models	253
21.4 <i>Examples</i>	253
22 Block Usage	257
22.1 Block Connections	257
22.2 Standard Blocks	258
22.3 Nonstandard Blocks	260
22.3.1 Buses	260
22.3.2 Goto and From Blocks	260
22.3.3 Links	260
22.3.4 Breakers	261
22.3.5 Power Supplies and Demands	262
22.3.6 Generator Ramping	262

22.3.7	Generator Reserves	262
22.3.8	Non-conventional Loads	262
22.3.9	Synchronous Machines	264
22.3.10	Primary Regulators	265
22.3.11	Secondary Voltage Regulation	265
22.3.12	Under Load Tap Changers	266
22.3.13	SVCs & STATCOMs	266
22.3.14	Solid Oxide Fuel Cells	268
22.3.15	Dynamic Shafts	268
23	Block Masks	271
23.1	Blocks vs. Global Structures	271
23.2	Editing Block Masks	272
23.2.1	Mask Initialization	272
23.2.2	Mask Icon	274
23.2.3	Mask Documentation	276
23.3	Syntax of Mask Parameter Names	276
23.4	Remarks on Creating Custom Blocks	277
V	Tools	281
24	Data Format Conversion	283
25	User Defined Models	287
25.1	Installing and Removing Models	287
25.2	Creating a User Defined Model	289
25.2.1	Component Settings	289
25.2.2	State Variable Settings	293
25.2.3	Parameter Settings	294
25.3	Limitations	294
26	Utilities	295
26.1	Command History	295
26.2	Sparse Matrix Visualization	295
26.3	Themes	295
26.4	Text Viewer	295
26.5	Building p-code Archive	296
27	Command Line Usage	301
27.1	Basics	301
27.2	Advanced Usage	304
27.3	Command Line Options	305
27.4	<i>Example</i>	306
28	Running PSAT on GNU Octave	309
28.1	Basic Commands	310

28.2 Plot Variables	310
28.3 ToDos	312
VI Interfaces	313
29 GAMS Interface	315
29.1 Getting Started	315
29.2 GAMS Solvers	316
29.3 PSAT-GAMS Interface	316
29.4 PSAT-GAMS Models	317
29.5 Multiperiod Market Clearing Model	320
29.5.1 Notation	320
29.5.2 Model Equations and Constraints	321
29.6 <i>Example</i>	323
30 UWPFLOW Interface	331
30.1 Getting Started	331
30.2 Graphical User Interface	332
30.3 Limitations and ToDos	332
30.4 <i>Example</i>	334
VII Libraries	339
31 Numeric Linear Analysis	341
31.1 Description	341
31.2 Test cases	342
31.2.1 Comparison of state matrices	343
31.2.2 Results for a change of an exciter reference voltage	343
31.2.3 Results for a change of governor reference speeds	344
31.2.4 Results for a change of a SVC reference voltage	347
VIII Appendices	351
A Global Structures & Classes	353
A.1 General Settings	353
A.2 Other Settings	357
A.3 System Properties and Settings	360
A.4 Outputs and Variable Names	366
A.5 User Defined Models	366
A.6 Models	368
A.7 Command Line Usage	370
A.8 Interfaces	371
A.9 Classes	372

B Matlab Functions	375
C Other Files and Folders	381
D Third Party Matlab Code	385
E Power System Softwares	387
F Test System Data	389
F.1 3-bus Test System	389
F.2 6-bus Test System	390
F.3 9-bus Test System	391
F.4 14-bus Test System	394
G FAQs	397
G.1 Getting Started	397
G.2 Simulink Library	399
G.3 Power Flow	400
G.4 Optimal & Continuation Power Flow	401
G.5 Time Domain Simulation	401
G.6 Data Conversion	402
G.7 Interfaces	403
H PSAT Forum	405
I Citations & Links	409
I.1 Books	409
I.2 Journals	409
I.3 Conference Proceedings	410
I.4 Webpages	410
J Letters of Reference	413
K The GNU General Public License	417
L GNU Free Documentation License	425
Bibliography	443

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
4.1	GUI for general settings.	32
4.2	GUI for displaying power flow results.	33
4.3	2D visualization of power flow results.	37
4.4	3D visualization of power flow results.	38
5.1	GUI for saddle-node bifurcation settings.	41
5.2	GUI for limit-induced bifurcation settings.	42
5.3	Continuation Power Flow: tangent vector	43
5.4	Continuation Power Flow: perpendicular intersection	44
5.5	Continuation Power Flow: local parametrization	45
5.6	GUI for the continuation power flow settings.	47
5.7	GUI for plotting CPF results.	48
5.8	Nose curves for the 6-bus test system (LIB)	49
5.9	Nose curves for the 6-bus test system (SNB)	50
6.1	GUI for the optimal power flow.	58
6.2	GUI for displaying OPF results.	59
6.3	GUI for plotting OPF Pareto sets.	62
7.1	Eigenvalue Analysis: S -domain.	65
7.2	Eigenvalue Analysis: Z -domain.	65
7.3	Eigenvalue Analysis: QV sensitivity.	70
7.4	GUI for the small signal stability analysis.	73
8.1	Time domain integration block diagram.	77
8.2	GUI for general settings.	79
8.3	GUI for plot variable selection.	81
8.4	Snapshot GUI.	82
8.5	GUI for plotting time domain simulations.	85
8.6	Generator speeds for the 9-bus test system.	86
8.7	Generator rotor angles for the 9-bus test system.	87

8.8	Bus voltages for the 9-bus test system.	88
9.1	PMU placement rules.	91
9.2	Flowchart of the Graph Theoretic Procedure.	92
9.3	Flowchart of the Bisecting Search.	93
9.4	Pseudo-code of the simulated Annealing Algorithm.	94
9.5	Recursive N Security Method.	95
9.6	Search of alternative placement sets.	95
9.7	Pure transit node filtering.	95
9.8	Single-Shot N Security Method.	96
9.9	Recursive N-1 Security Method.	97
9.10	Single Shot N-1 Security Method.	98
9.11	GUI for the PMU placement methods.	99
10.1	Transmission line π circuit.	105
10.2	Three-winding transformer equivalent circuit.	108
11.1	Example of daily demand profile.	125
13.1	Bus frequency measurement filter.	131
13.2	Phasors from sample data.	133
14.1	Measure of frequency deviation.	138
14.2	Thermostatically controlled load.	141
14.3	Jimma's load.	142
15.1	Synchronous machine scheme.	150
15.2	Synchronous machine: block diagram of stator fluxes.	151
15.3	Field saturation characteristic of synchronous machines.	152
15.4	Order I induction motor: electrical circuit.	161
15.5	Order III induction motor: electrical circuit.	162
15.6	Order V induction motor: electrical circuit.	163
16.1	Turbine governor type I.	167
16.2	Turbine governor type II.	168
16.3	Exciter Type I.	170
16.4	Exciter Type II.	172
16.5	Exciter Type III.	173
16.6	Power system stabilizer Type I.	176
16.7	Power system stabilizer Type II.	176
16.8	Power system stabilizer Type III.	177
16.9	Power system stabilizer Type IV.	178
16.10	Power system stabilizer Type V.	178
16.11	Over excitation limiter.	180
16.12	Secondary voltage control scheme.	182
17.1	Under Load Tap Changer: equivalent π circuit.	187

17.2	Under Load Tap Changer: voltage and reactive power controls. . .	187
17.3	Load Tap Changer with embedded load.	189
17.4	Phase shifting transformer circuit.	191
17.5	Phase shifting transformer control scheme.	191
18.1	SVC Type 1 Regulator.	194
18.2	SVC Type 2 Regulator.	195
18.3	TCSC Regulator.	197
18.4	STATCOM circuit and control block diagram.	200
18.5	SSSC circuit.	202
18.6	SSSC control block diagram.	203
18.7	UPFC circuit.	206
18.8	UPFC phasor diagram.	206
18.9	UPFC control block diagrams.	207
18.10	HVDC scheme.	210
18.11	HVDC current control.	211
19.1	Low-pass filter to smooth wind speed variations.	214
19.2	Wind turbine types	218
19.3	Rotor speed control scheme.	224
19.4	Voltage control scheme.	225
19.5	Power-speed characteristic.	225
19.6	Pitch angle control scheme.	225
20.1	Synchronous machine mass-spring shaft model.	232
20.2	Generator with dynamic shaft and compensated line.	234
20.3	Solid Oxide Fuel Cell scheme.	239
20.4	Solid Oxide Fuel Cell connection with the AC grid.	241
20.5	AC voltage control for the Solid Oxide Fuel Cell.	241
21.1	Simulink library: Main Window.	246
21.2	Simulink library: Connections.	246
21.3	Simulink library: Power Flow data.	247
21.4	Simulink library: OPF & CPF data.	248
21.5	Simulink library: Faults & Breakers.	248
21.6	Simulink library: Measurements.	248
21.7	Simulink library: Loads.	249
21.8	Simulink library: Machines.	249
21.9	Simulink library: Regulators.	250
21.10	Simulink library: Regulating Transformers.	250
21.11	Simulink library: FACTS controllers.	251
21.12	Simulink library: Wind Turbines.	252
21.13	Simulink library: Other models.	252
21.14	GUI for Simulink model settings.	253
21.15	Simulink model of the WSCC 3-generator 9-bus test system. . . .	254
21.16	Simulink model of the IEEE 14-bus test system.	255

21.17	Simulink model of the 6-bus test system.	256
22.1	Examples of standard blocks of the PSAT SIMULINK Library. . . .	258
22.2	Examples of allowed connections.	259
22.3	Examples of not allowed connections.	259
22.4	Examples of infeasible connections.	259
22.5	Bus block usage.	260
22.6	Goto and From block usage.	261
22.7	Breaker block usage.	261
22.8	Supply and Demand block usage.	262
22.9	Generator Ramping block usage.	263
22.10	Generator Reserve block usage.	263
22.11	Non-conventional Load block usage.	264
22.12	Synchronous Machine block usage.	265
22.13	Primary Regulator block usage.	266
22.14	Secondary Voltage Regulation block usage.	267
22.15	Under Load Tap Changer block usage.	267
22.16	SVC block usage.	268
22.17	Solid Oxide Fuel Cell block usage.	268
22.18	Dynamic Shaft block usage.	269
23.1	SIMULINK blocks vs. PSAT global structures	272
23.2	Mask GUI of a PSAT-SIMULINK block.	273
23.3	Mask initialization GUI for a PSAT-SIMULINK block.	274
23.4	Mask icon GUI of a PSAT-SIMULINK block.	275
23.5	Mask documentation GUI of a PSAT-SIMULINK block.	276
23.6	SIMULINK model underneath a mask of a PSAT block.	279
24.1	GUI for data format conversion.	285
25.1	Browser of user defined models.	288
25.2	GUI for creating user defined models.	290
25.3	GUI for setting component properties.	291
25.4	GUI for setting state variable properties.	292
25.5	GUI for setting parameters properties.	293
26.1	Command history GUI.	296
26.2	GUI for sparse matrix visualization.	297
26.3	GUI for PSAT theme selection.	298
26.4	GUI for text viewer selection.	299
26.5	GUI for p-code archive builder.	299
27.1	Master-slave architecture.	304
28.1	Example of graph obtained using GNU/Octave and gplot.	311
29.1	Structure of the PSAT-GAMS interface.	318

29.2	GUI of the PSAT-GAMS interface.	319
29.3	PSAT-Simulink model of the three-bus test system.	324
29.4	Demand profile for the multiperiod auction.	328
29.5	Multiperiod auction without P_{mn}^{\max} limits.	329
29.6	Multiperiod auction with P_{mn}^{\max} limits.	330
30.1	GUI of the PSAT-UWPFLOW interface.	333
30.2	UWPFLOW nose curves for the 6-bus test systems.	338
31.1	Comparison of voltages at buses 6 and 7.	345
31.2	Comparison of reactive powers flows in lines 2-7 and 6-4.	345
31.3	Comparison of active powers flows in line 2-7.	346
31.4	Comparison of rotor speeds.	348
31.5	Detail of the comparison of rotor speeds.	348
31.6	Comparison of active powers flows in line 2-7.	349
31.7	Comparison of SVC state variables.	350
31.8	Comparison of voltages at bus 8.	350
F.1	3-bus test system.	390
F.2	6-bus test system.	392
F.3	WSCC 3-generator 9-bus test system.	394
F.4	IEEE 14-bus test system.	396
H.1	PSAT Forum main page	406
H.2	PSAT Forum statistics	407

List of Tables

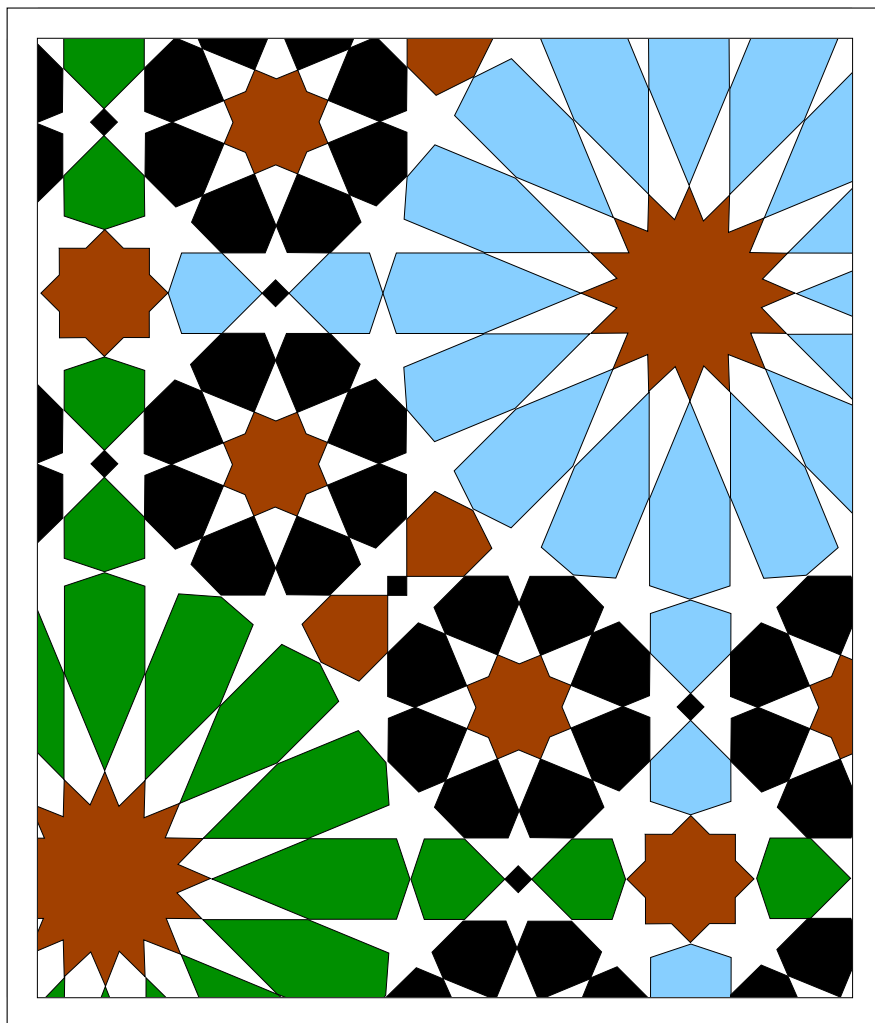
1.1	Matlab-based packages for power system analysis	6
5.1	N-1 Contingency Analysis Report	51
10.1	Bus Data Format	103
10.2	Line Data Format	106
10.3	Alternative Line Data Format	106
10.4	Transformer Data Format	107
10.5	Three-Winding Transformer Data Format	109
10.6	Slack Generator Data Format	110
10.7	PV Generator Data Format	112
10.8	PQ Load Data Format	113
10.9	PQ Generator Data Format	114
10.10	Shunt Admittance Data Format	114
10.11	Area & Regions Data Format	115
11.1	Power Supply Data Format	119
11.2	Power Reserve Data Format	120
11.3	Generator Power Ramping Data Format	121
11.4	Power Demand Data Format	123
11.5	Demand Profile Data Format	125
11.6	Load Ramping Data Format	126
12.1	Fault Data Format	128
12.2	Breaker Data Format	129
13.1	Bus Frequency Measurement Data Format	132
13.2	Phasor Measurement Unit Data Format	134
14.1	Voltage Dependent Load Data Format	136
14.2	ZIP Load Data Format	137
14.3	Frequency Dependent Load Data Format	139
14.4	Typical load coefficients	139
14.5	Exponential Recovery Load Data Format	140
14.6	Thermostatically Controlled Load Data Format	142
14.7	Jimma's Load Data Format	143

14.8	Mixed Load Data Format	145
15.1	Synchronous Machine Data Format	149
15.2	Reference table for synchronous machine parameters.	150
15.3	Induction Motor Data Format	160
16.1	Turbine Governor Type I Data Format	167
16.2	Turbine Governor Type II Data Format	168
16.3	Exciter Type I Data Format	171
16.4	Exciter Type II Data Format	172
16.5	Exciter Type III Data Format	173
16.6	Power System Stabilizer Data Format	175
16.7	Over Excitation Limiter Data Format	180
16.8	Central Area Controller Data Format	182
16.9	Cluster Controller Data Format	183
16.10	Power Oscillation Damper Data Format	184
17.1	Load Tap Changer Data Format	188
17.2	Tap Changer with Embedded Load Data Format	190
17.3	Phase Shifting Transformer Data Format	192
18.1	SVC Type 1 Data Format	195
18.2	SVC Type 2 Data Format	196
18.3	TCSC Data Format	199
18.4	STATCOM Data Format	200
18.5	SSSC Data Format	203
18.6	UPFC Data Format	208
18.7	HVDC Data Format	212
19.1	Wind Speed Data Format	215
19.2	Roughness length for various ground surfaces	217
19.3	Recent wind turbines	219
19.4	Constant Speed Wind Turbine Data Format	222
19.5	Doubly Fed Induction Generator Data Format	226
19.6	Direct Drive Synchronous Generator Data Format	229
20.1	Dynamic Shaft Data Format	233
20.2	SSR Data Format	237
20.3	Solid Oxide Fuel Cell Data Format	240
23.1	Mask parameter symbols	277
23.2	Example of well formed mask variable names	277
23.3	Mask parameter constants	278
25.1	Functions and files to be modified for installing a UDM	287
27.1	Routine Conventional Names for Command Line Usage.	303

27.2	General Options for Command Line Usage.	304
27.3	Structures to be modified to change default behavior.	305
29.1	PSAT IPM-based OPF report for the three-bus test system.	325
29.2	PSAT-GAMS OPF report for the three-bus test system.	326
29.3	Input file <code>psatglobs.gms</code> for the three-bus test system.	326
29.4	Input file <code>psatdata.gms</code> for the three-bus test system.	327
29.5	Output file <code>psatsol.m</code> for the three-bus test system.	327
30.1	IEEE CDF file to be used within UWPFLOW	335
30.2	UWPFLOW power flow results	336
30.3	Input file which defines power directions in UWPFLOW	337
30.4	UWPFLOW output file with CPF results	337
31.1	State matrix eigenvalues for the 9-bus test system	343

Part I

Outlines



Chapter 1

Introduction

This chapter provides 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 reported.

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.

PSAT core is the power flow routine, which also takes care of state variable initialization. Once the power flow has been solved, further static and/or dynamic analysis can be performed. 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, generator ramping data, and power demand bids and limits.

- ◊ *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, 63], thermostatically controlled loads [57], Jimma's loads [61], 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 model, and Solid Oxide Fuel Cell.

Besides mathematical routines and models, PSAT includes a variety of utilities, as follows:

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

Finally, PSAT includes 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.

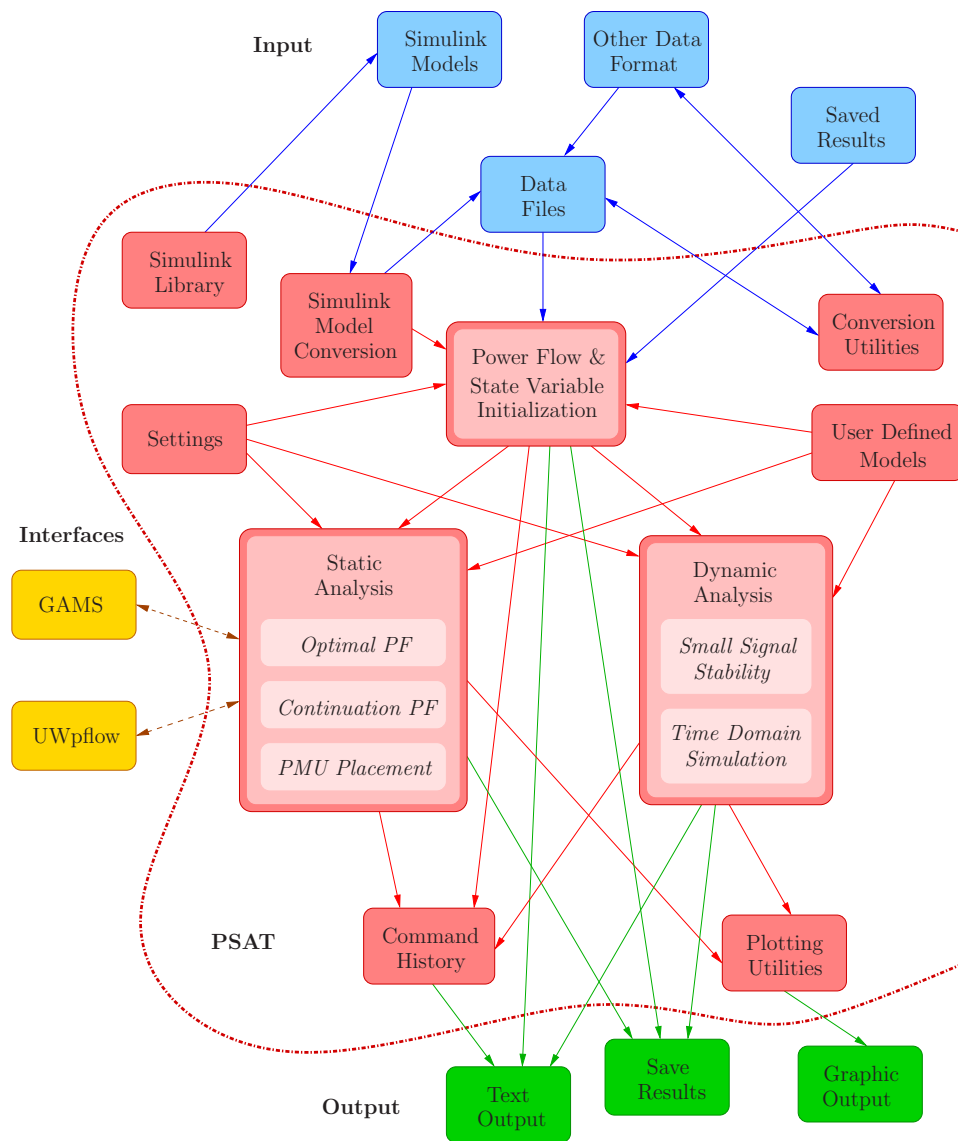


Figure 1.1: PSAT at a glance.

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) [121];
2. MatEMTP [72];
3. MATPOWER [132];
4. Power System Toolbox (PST) [35, 33, 32]
5. Power Analysis Toolbox (PAT) [103];
6. SimPowerSystems (SPS) [113];¹
7. Voltage Stability Toolbox (VST) [31, 90].

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 Manual

This documentation is divided in seven parts, as follows.

Part I provides an introduction to PSAT features and a quick tutorial.

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

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

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

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, Belgium, Brazil, 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, and Vietnam. Figure 1.2 depicts PSAT users around the world.

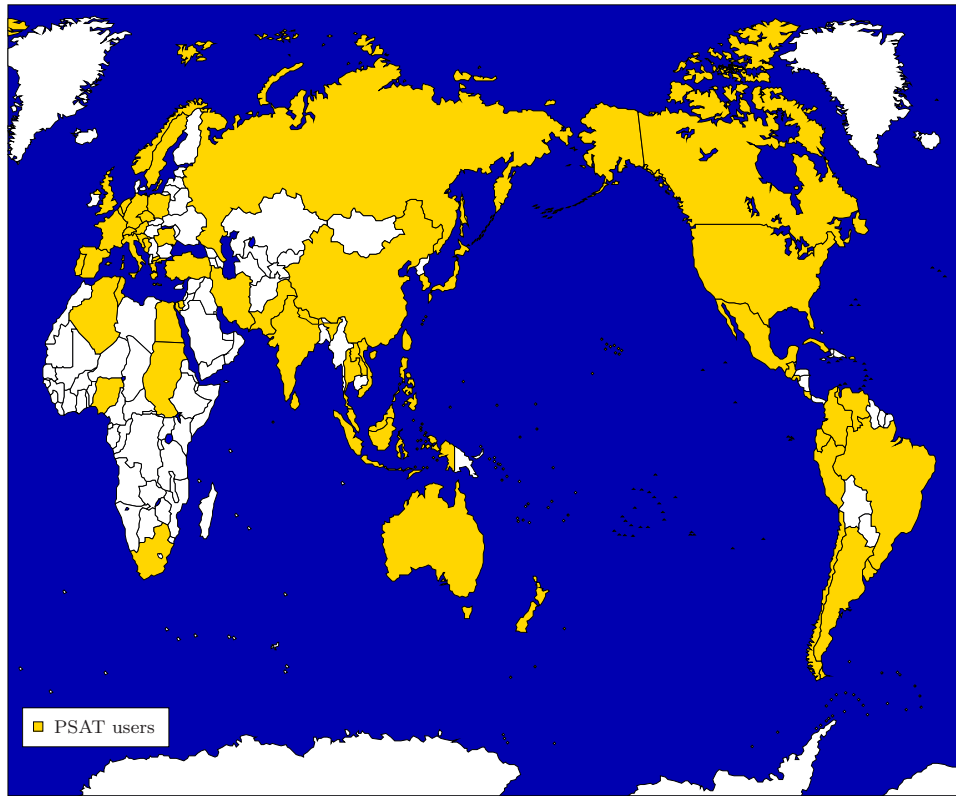


Figure 1.2: PSAT around the world.

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 presented.

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 and is kindly provided by Prof. Claudio A. Cañizares, who has been my supervisor for 16 months (September 2001-December 2002), when I was a Visiting Scholar at the E&CE of the University of Waterloo, Canada.

2.2 Requirements

PSAT 2.0.0 has been developed using MATLAB 7.0.4 (R14) on Fedora Linux Core 4 for i686. It has also been tested on a Sun workstation (Solaris 2.9), Irix 6.5, Mac OS X 10.4 and Windows XP platforms. The new PSAT 2.0.0 is not compatible with older version of Matlab. PSAT 2.0.0 makes use of the latest features of the current MATLAB R14, such as physical components for the Simulink library. Furthermore, PSAT 2.0.0 makes use of classes, thus it is not compatible with GNU Octave.

User that needs compatibility with older Matlab versions back to 5.3 (R11) and/or with GNU Octave should use PSAT 1.3.4. Observe that in PSAT 1.3.4 some of the latest MATLAB features are disabled. This is the case of some built-in functions (e.g. `uigetdir`) and Perl modules.¹

¹Perl filters for data file conversion can be used only with MATLAB 6.5. Older MATLAB files such as `fm.cdf.m` are still included in the PSAT distribution but will be no longer maintained.

In order to run PSAT, only the basic MATLAB and SIMULINK packages are needed, except for compiling user defined models, which requires the Symbolic Toolbox.

The command line version of PSAT 1.3.4 can work on GNU Octave as well. In particular, the main PSAT 1.3.4 routines and component models have been tested using version 2.1.72 and the version 2005.06.13 of the octave-forge package on Fedora Linux Core 4 for i686.²

2.3 Installation

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-pcode-1.x.y.tar.gz
$ tar xvf psat-pcode-1.x.y.tar
```

or:

```
$ tar zxvf psat-pcode-1.x.y.tar
```

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

```
$ unzip psat-pcode-1.x.y.zip
```

where *x* and *y* are the current PSAT patch numbers. This will create in the working directory a **psat** folder which contains all p-code files and all necessary directories. 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 it and extract the PSAT package.

Then open 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 operation but at the MATLAB prompt. For example:

²Refer to Chapter 28 for further information on the usage of PSAT on GNU Octave.


```
>> 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 in different folders. Just be sure that in your `pathdef.m` file there is no PSAT folder. You should also update the MATLAB path or restart MATLAB anytime you want to work with a different PSAT version.
4. If you have an older version of PSAT on your computer and this version is working fine, just expand the PSAT tarball on top of it. Then launch PSAT as usual.

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

Note 2: PSAT makes use of four internal folders (`images`, `build`, `themes`, and `filters`). It is recommended not to change the position and the names of these folders. Observe that PSAT can work properly only if the current MATLAB folder and the data file folders are writable. Furthermore, also the PSAT folder should be writable if you want to build and install user defined components.

2.4 Launching PSAT

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

```
>> psat
```

This will create all the structures required by the toolbox, as follows:³

³By 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`.

```
>> 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
Bus	Exload	Lines	Param	SNB	Supply	Vltn
Buses	Fault	Ltc	Path	SSR	Svc	Wind
Busfreq	Fig	Mass	Phs	SSSA	Syn	Ypdp
CPF	File	Mixed	Pl	SW	Tap	ans
Cac	Fl	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 window⁴ 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 <Ctrl-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 <Ctrl-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 8 for details). In order to use the program, it is not necessary to load a perturbation file,

⁴This 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.

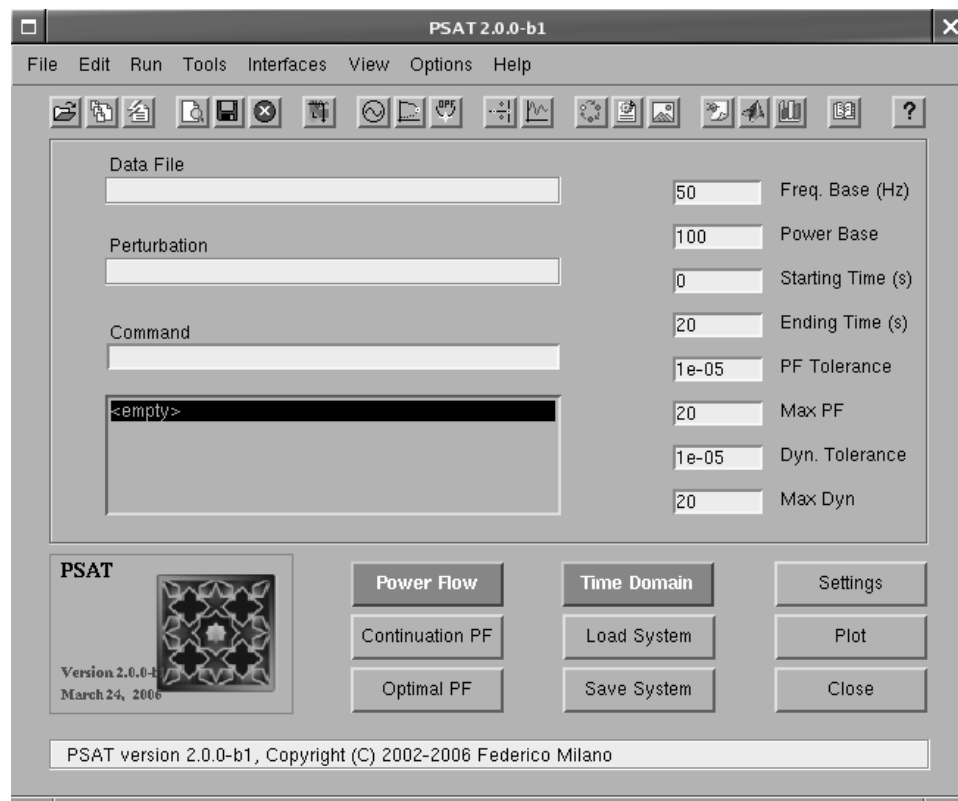


Figure 2.1: Main graphical user interface of PSAT.

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 <Ctrl-p>). Refer to Chapter 4 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 5), Optimal Power Flow (Chapter 6), Small Signal Stability Analysis (Chapter 7), Time Domain Simulation (Chapter 8), PMU placement (Chapter 9), 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 <Ctrl-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 <Ctrl-w>). Refer to the chapters where these topics are discussed for details and examples.

Some computations and several user actions result also in messages stored in the **History** structure. These messages/results are displayed one at the time in the static text banner at the bottom of the main window. By double clicking on this banner or using the menu *Options/History* 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.⁵

⁵All 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 me whenever you encounter one.

2.8 Saving Results

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

Furthermore, all static computations allow to create a report in a text file that can be stored and used later. The extensions for these files are as follows:

- .txt* for reports in plain text;
- .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.⁶ 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 and 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.

⁶Well, after writing the 99th file, the file with the number 01 is actually overwritten without asking for any confirmation.

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 26.

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 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 21.

2.11 Tools

Several tools are provided with PSAT, e.g. data format conversion functions and user defined model routines.

The data format conversion routines (see Chapter 24) allow importing data files from other power system software packages. However, observe 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 Common Data Format.

User defined model routines (see Chapter 25) provide a simple way for extending the capabilities of PSAT and, hopefully, facilitating contributions. The construction of a user defined model can be yielded in few steps, as follows:

1. Define parameters and differential and algebraic equations by means of a GUI;
2. Create the MATLAB function of the model;⁷
3. Save the model in a *.m* file;
4. Install the model in the program, by means of an automatic procedure.

If the component is not needed any longer it can also be “uninstalled” in a similar way. Thus, user defined models can be shared easily by simply providing the component function and the component structure stored in a MATLAB script file. However, the routine which compiles model functions is not complete so far, and it is intended only for creating a first draft of the component function.

⁷The Symbolic Toolbox is required for building the new component function.

Other PSAT tools and utilities, such as the command history, the sparse matrix visualization GUI, the theme selector, and the text viewer selector are described in Chapter 26.

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 29 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 30 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 30.3 for limitations and Todos.

Chapter 3

News

This chapter lists new features of the current release of PSAT with respect of older versions.

3.1 News in version 2.0.0 beta

1. This is a development version and is only compatible with MATLAB 7.0 or newer. Note that this version is NOT compatible with Octave.
2. First version of PSAT which uses classes.
3. Added the *status* field of several components.
4. New more reliable versions of FACTS and Power Oscillations damper model for FACTS by H. Ayres, M. S. Castro and A. Del Rosso.
5. New SIMULINK library with physical components.
6. Several new filters for data format conversion by J. C. Morataya.
7. Improved PF, CPF, OPF, SSSA and TD algorithms.
8. This version has been tested using a 15000 bus test network.
9. Added the possibility of monitoring voltage evolution on SIMULINK models during time domain simulations.
10. Added a structure for Interchange Area definition. currently this structure is idle but will be used in future version of PSAT.
11. The check of component bases has been improved to take into account voltage rates (see function `fm.base`) of several components.
12. Corrected the way shunts are included in the admittance matrix, so that more than shunt is allowed per bus.

13. Corrected the Jacobian and hessian matrix of apparent power flows in transmission line for OPF routine.
14. Changed the logo of PSAT.

3.2 News in version 1.3.4

1. Added unit commitment and multiperiod market clearing models for the PSAT-GAMS interface (see Section 29.5).
2. Added Phasor Measurement Unit (PMU) model (see Section 13.2).
3. Added Jimma's load model (see Section 14.6).
4. Added mixed load model (see Section 14.7).
5. Added a filter to convert data file in NEPLAN format (see Chapter 24).
6. Added the possibility of exporting plots as MTV plot files and as Matlab scripts. These new features are available from within the GUI for plotting results.
7. 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.
8. Added the option of stopping time domain simulations when the machine angle degree is greater than a given $\Delta\delta_{\max}$ (default value 180°).
9. Added the function `fm_connectivity` to detect separation in areas following a breaker operation during time domain simulation (by courtesy of Laurent Lenoir).
10. 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.
11. Patched the `fm_sim` function. It is now allowed using bus names with carriage return characters.
12. Many minor function patches. These are: `fm_breaker`, `fm_cdf`, `fm_int`, `fm_m2cdf`, `fm_m2wscc`, `fm_ncomp`, `fm_opfm`, `fm_opfsdr`, `fm_pss`, `fm_snb`.

3.3 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 [63]. The corresponding component structure has been renamed `Exload`.

3.4 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 octave 2.1.57 & octave-forge 2004-07-07 for MAC OS X 10.3.5 (by Randall Smith).

3.5 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), AVRs and PSSs.
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 Octave compatibility issues.

3.6 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 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, \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.

3.7 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, \LaTeX or plain text formats.
3. Added filters to convert data files into BPA and Tshingua 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.

3.8 News in version 1.2.1

Minor bug-fixing release. Main improvements are in functions `psat.m`, `fm.base.m` and `fm.sim.m`.

3.9 News in version 1.2.0

1. First PSAT release which 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.

3.10 News in version 1.1.0

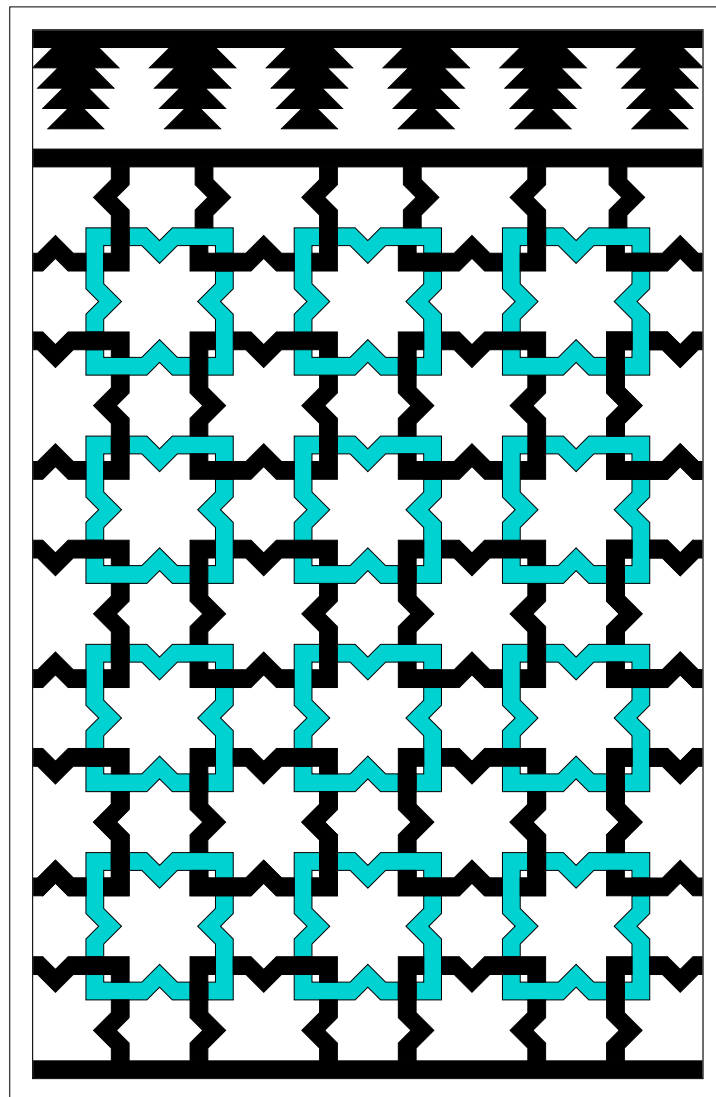
1. Created the PSAT Forum (<http://groups.yahoo.com/group/psatforum>).
2. Added PSAT/GAMS interface.
3. Added PSAT/UWPFLOW interface.
4. Added phase shifting transformer model.
5. Added filter for CYMFLOW data format.
6. Corrected some bugs in the filter for MatPower data format.

3.11 News in version 1.0.1

Minor bug-fixing release. Main improvements are in functions `fm.fault.m` and in the documentation.

Part II

Routines



Chapter 4

Power Flow

This chapter describes routines, settings and graphical user interfaces for power flow computations. The standard Newton-Raphson method [116] and the Fast Decoupled Power Flow (XB and BX variations [111, 110, 118]) are implemented. A power flow with a distributed slack bus model is also included.

4.1 Power Flow Solvers

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

$$\begin{aligned}\dot{x} &= 0 = f(x, y) \\ 0 &= g(x, y)\end{aligned}\tag{4.1}$$

where y ($y \in \mathbb{R}^m$) are the algebraic variables, i.e. voltage amplitudes V and phases θ at the network buses and all other algebraic variables such as generator field voltages, AVR reference voltages, etc., x ($x \in \mathbb{R}^n$) are the state variables, g ($g \in \mathbb{R}^m$) are the algebraic equations and f ($f \in \mathbb{R}^n$) 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 (4.1) 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 4.1.4 for the complete list of components that are included in or initialized after the power flow solution.

4.1.1 Newton-Raphson Method

Newton-Raphson algorithms for solving the power flow problem are described in many books and papers (e.g. [116]). At each iteration, the Jacobian matrix of (4.1)

is updated and the following linear problem is solved:

$$\begin{aligned} \begin{bmatrix} \Delta x^i \\ \Delta y^i \end{bmatrix} &= - \begin{bmatrix} F_x^i & -F_y^i \\ G_x^i & G_y^i \end{bmatrix}^{-1} \begin{bmatrix} f^i \\ g^i \end{bmatrix} \\ \begin{bmatrix} x^{i+1} \\ y^{i+1} \end{bmatrix} &= \begin{bmatrix} x^i \\ y^i \end{bmatrix} + \begin{bmatrix} \Delta x^i \\ \Delta y^i \end{bmatrix} \end{aligned} \quad (4.2)$$

where $F_x = \nabla_x f$, $F_y = \nabla_y f$, $G_x = \nabla_x g$ and $J_{LFV} = \nabla_y g$. If the variable increments Δx and Δy 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 J_{LFV} is a submatrix of 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 $g_{P_{slack}}$ is set to zero;
- The rows of the derivatives of generator reactive power balances g_Q 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 g associated with the generator reactive powers and the slack bus active power are set to zero.

These assumptions are equivalent to the equations:

$$\begin{aligned} \theta_{slack} &= \theta_{slack_0} \\ V_G &= V_{G_0} \end{aligned} \quad (4.3)$$

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

4.1.2 Fast Decoupled Power Flow

The Fast Decoupled Power Flow (FDPF) was originally proposed in [111] and has been further developed and generalized in several variations. PSAT uses the XB and BX methods presented in [118].

This method can be used if algebraic variable are only voltage magnitudes and phases. In this case, $G_y = J_{LFV}$. The power flow Jacobian matrix J_{LFV} can be decomposed in four sub-matrices:

$$J_{LFV} = \begin{bmatrix} J_{P\theta} & J_{PV} \\ J_{Q\theta} & J_{QV} \end{bmatrix} \quad (4.4)$$

where $J_{P\theta} = \nabla_{\theta} g_P$, $J_{PV} = \nabla_V g_P$, $J_{Q\theta} = \nabla_{\theta} g_Q$, and $J_{QV} = \nabla_V g_Q$. The basic assumptions of FDPF methods are:

$$\begin{aligned} J_{PV} &= 0 \\ J_{Q\theta} &= 0 \\ J_{P\theta} &\approx B' \\ J_{QV} &\approx B'' \end{aligned} \quad (4.5)$$

where B' and B'' can be thought as admittance matrices with the following simplifications:

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

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

XB: line resistances are neglected when computing B' ;

BX: line resistances are neglected when computing B'' .

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

$$\begin{aligned} \Delta g_P^{i'}/V^{i'} &= B' \Delta \theta^{i'} \\ \Delta g_Q^{i''}/V^{i''} &= B'' \Delta V^{i''} \end{aligned} \quad (4.6)$$

where Δg_P and Δg_Q are the active and reactive power flow equation mismatches. The solution of the active equations is used as input to the reactive ones, as this reduces the number of iterations.

PSAT allows using FDPF methods for system which contain only PV generators, PQ loads and one slack bus. If other components are present in the network, the standard Newton-Raphson routine is used.

4.1.3 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 (4.1) a variable k_G and rewriting the system active power balance as follows:

$$\sum_i^{n_G} (1 + k_G \gamma_i) P_{G_i} - \sum_i^{n_P} P_{L_i} - P_{losses} = 0 \quad (4.7)$$

Equations (4.2) are modified by adding to the Jacobian matrix J_{LFV} 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 . The additional parameter γ is also included in order to allow tuning the weight of the participation of each generator to the losses. (In the single slack bus model, $\gamma = 0$ for all generators but one.) When the distributed slack bus flag is active, FDPF methods are automatically disabled.

4.1.4 Initialization of State Variables

Dynamic components and non-conventional loads can be included in or initialized after the power flow solution. The following components are included in the power flow equation set:

Hvdc	Lines	Ltc	Mn	Mot	PQ	PV
Phs	P1	SAE1	SAE2	SAE3	SW	Tap

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

Busfreq	Cac	Cluster	Cswt	Ddsg	Dfig	Exc
F1	Exload	Mass	Mn	Oxl	P1	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 the specific descriptions of each component for details.

4.2 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 the Newton-Raphson/FDPF techniques 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. 4.1. The following options are available for power flow analysis:

Power Flow Solver: these are the Newton-Raphson (NR) method, the fast decoupled XB and BX methods. Observe that only the NR method is available if the distributed slack bus option is enabled or if there are dynamic component included in the power flow analysis.

Use Distributed Slack Bus: 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: 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 Comp.: 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: 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.

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

Show Power Flow Results: 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 fully described in Appendix A.

4.3 Example

Figure 4.2 depicts the GUI for power flow results. Data refer to a 9-bus test system presented and discussed in [101]. 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, \LaTeX , Excel) and displayed with the selected viewer (see Section 26.4 for details). For example, the plain text power flow solution for the WSCC 9-bus test system is as follows:

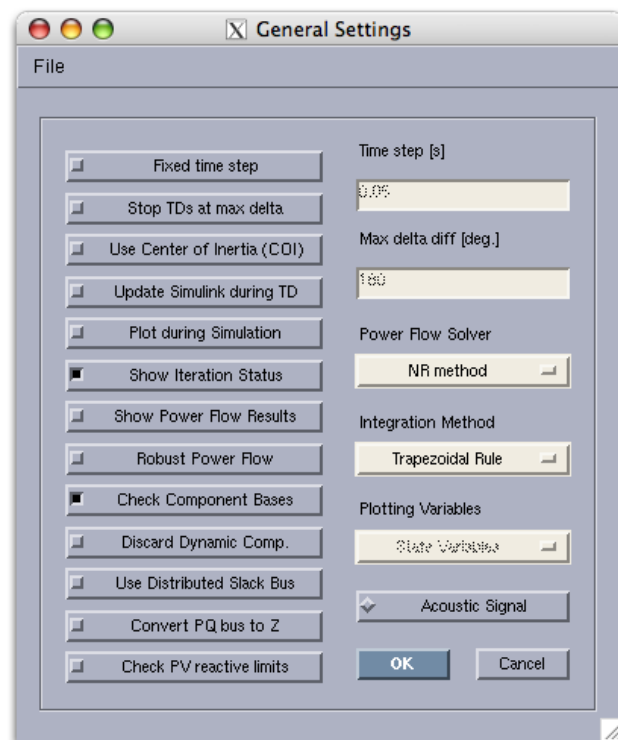


Figure 4.1: GUI for general settings.

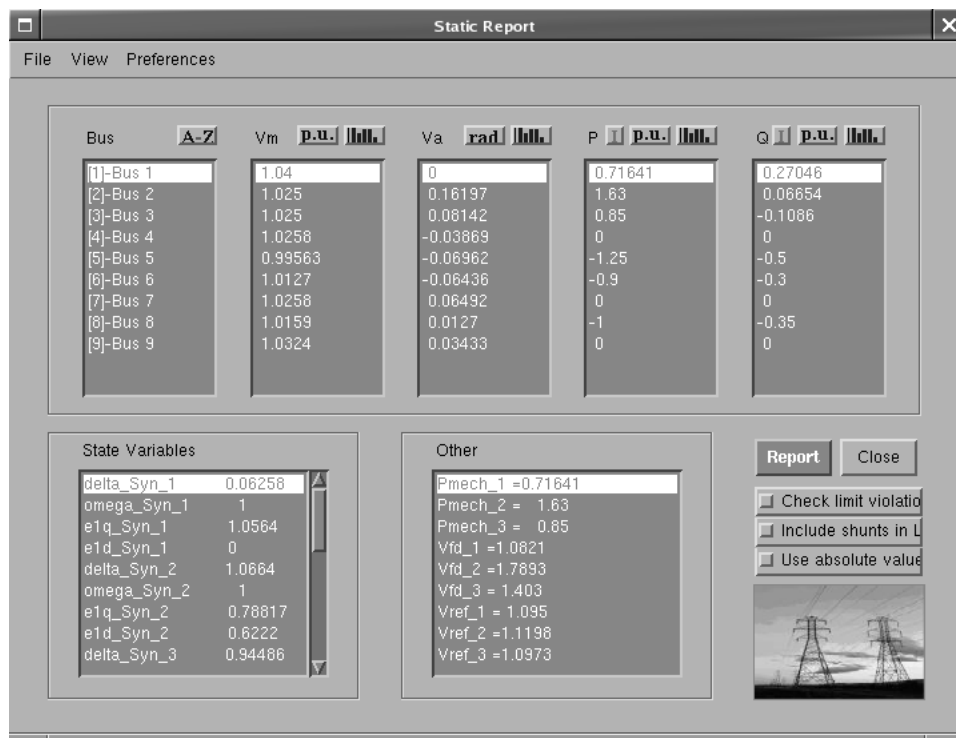


Figure 4.2: GUI for displaying power flow results.

POWER FLOW REPORT

P S A T 1.3.4

Author: Federico Milano, (c) 2002-2005
 e-mail: fmilano@thunderbox.uwaterloo.ca
 website: <http://thunderbox.uwaterloo.ca/~fmilano>

File: ~/psat/tests/d_009.mdl

Date: 26-Oct-2003 12:53:43

NETWORK STATISTICS

Bus: 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 VECTOR

delta_Syn_1	1.0664
omega_Syn_1	1
e1q_Syn_1	0.78817
e1d_Syn_1	0.6222
delta_Syn_2	0.94486
omega_Syn_2	1
e1q_Syn_2	0.76786
e1d_Syn_2	0.62424
delta_Syn_3	0.06258
omega_Syn_3	1
e1q_Syn_3	1.0564
e1d_Syn_3	0
vm_Exc_1	1.025

vr1_Exc_1	1.446
vr2_Exc_1	-0.25254
efd_Exc_1	1.403
vm_Exc_2	1.025
vr1_Exc_2	1.8951
vr2_Exc_2	-0.32208
efd_Exc_2	1.7893
vm_Exc_3	1.04
vr1_Exc_3	1.1006
vr2_Exc_3	-0.19479
efd_Exc_3	1.0822

MECHANICAL POWERS & FIELD VOLTAGES

Pmech_1	1.63
Pmech_2	0.85
Pmech_3	0.71641
Vfd_1	1.7893
Vfd_2	1.403
Vfd_3	1.0822

EXCITER REFERENCE VOLTAGES

Vref_1	1.0973
Vref_2	1.1198
Vref_3	1.095

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 SHUNT

REAL POWER [p.u.]	0
REACTIVE POWER (IND) [p.u.]	0
REACTIVE POWER (CAP) [p.u.]	0

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. 4.3 and 4.4). 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, and generator rotor angles and speeds. The same GUI can be used to create movies for CPF analysis and time domain simulations.

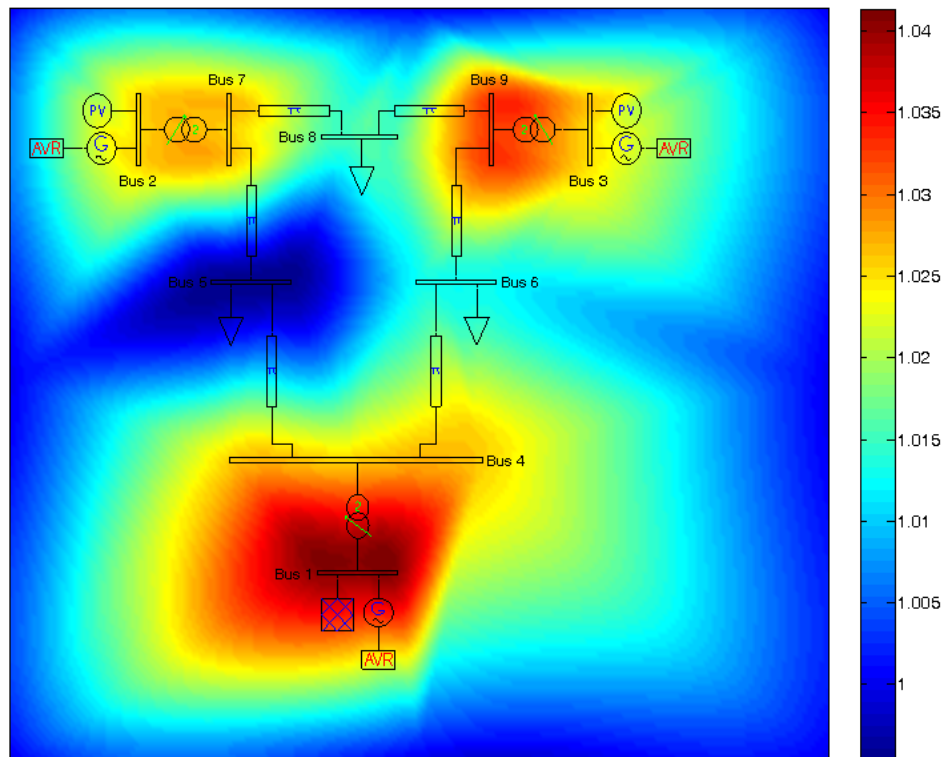


Figure 4.3: 2D visualization of power flow results. Voltage magnitudes for the 9-bus test system.

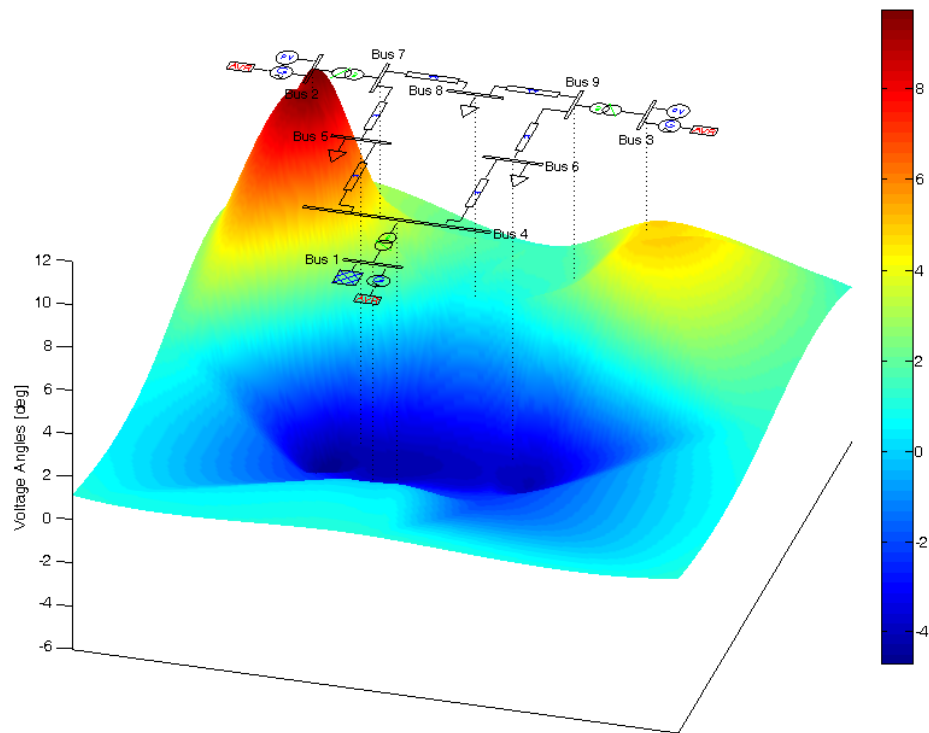


Figure 4.4: 3D visualization of power flow results. Voltage angles for the 9-bus test system.

Chapter 5

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:

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

where x are the state variables, 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:

$$\begin{aligned}P_G &= P_{G_0} + (\lambda + \gamma k_G)P_{S_0} \\ P_L &= P_{L_0} + \lambda P_{D_0} \\ Q_L &= Q_{L_0} + \lambda Q_{D_0}\end{aligned}\tag{5.2}$$

In (5.2), P_{G_0} , P_{L_0} and Q_{L_0} are the “base case” generator and load powers, whereas P_{S_0} , P_{D_0} and Q_{D_0} are the generator and load power directions. Power directions are defined in the structures **Supply** and **Demand** (see also Sections 11.1 and 11.4 for more details). If these data are not defined, the base case powers are used as load directions and (5.2) become:

$$\begin{aligned}P_G &= (\lambda + \gamma k_G)P_{G_0} \\ P_L &= \lambda P_{L_0} \\ Q_L &= \lambda Q_{L_0}\end{aligned}\tag{5.3}$$

Observe that power directions (5.2) and (5.3) used in the bifurcation analysis differ from (6.3), i.e. the power directions used in the voltage stability constrained OPF described in Chapter 6. The distributed slack bus variable k_G and the generator participation coefficients γ are optional.

5.1 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 10). Thus, (5.1) reduce to the algebraic set g . Before running any direct method routine, the power flow analysis has to be run first to initialize the algebraic variables.

5.1.1 Saddle-Node Bifurcation

The conditions for a SNB point are as follows:

$$\begin{aligned} g(y, \lambda) &= 0 \\ \nabla_y g(y, \lambda)v &= 0 \\ |v| &= 1 \end{aligned} \tag{5.4}$$

or

$$\begin{aligned} g(y, \lambda) &= 0 \\ \nabla_y g(y, \lambda)^T w &= 0 \\ |w| &= 1 \end{aligned} \tag{5.5}$$

where v and 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 (5.4) and (5.5) are obtained by means of a Newton-Raphson technique, and the complete Jacobian matrix is computed explicitly:

$$\begin{bmatrix} \nabla_y g & 0 & \nabla_\lambda g \\ \nabla_{yy} g v & \nabla_y g & 0 \\ 0 & \nabla_v(|v|) & 0 \end{bmatrix} \tag{5.6}$$

Since the Hessian matrix $\nabla_{yy} g$ 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 v and w . However the best way to initialize the SNB routine is to run first a CPF analysis.

Figure 5.1 depicts the GUI for SNB settings. A complete description of SNB settings is reported in Appendix A.

5.1.2 Limit Induced Bifurcation

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

$$\begin{aligned} 0 &= g(y, \lambda) \\ 0 &= \rho(y) \end{aligned} \tag{5.7}$$

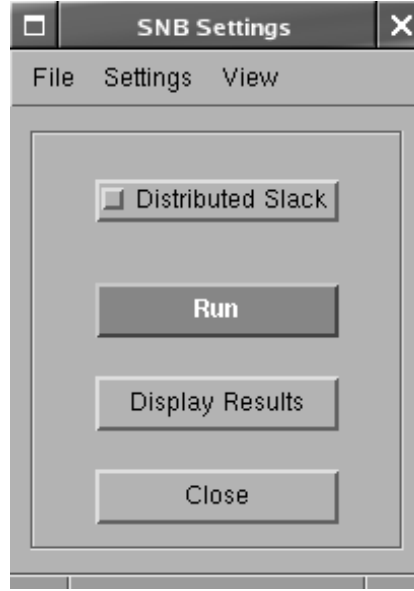


Figure 5.1: GUI for saddle-node bifurcation settings.

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

$$Q_G = Q_{G_{lim}} \quad (5.8)$$

for slack or PV generator buses , or

$$V_L = V_{L_{lim}} \quad (5.9)$$

for PQ load buses . Observe that 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 5.2 depicts the GUI for LIB settings. A complete description of LIB structure is reported in Appendix A.

5.2 Continuation Power Flow

The Continuation Power Flow 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.

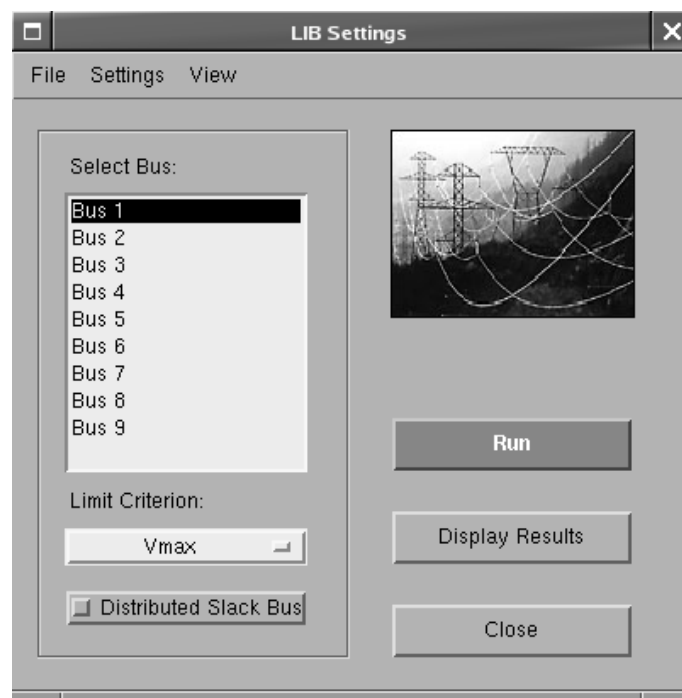


Figure 5.2: GUI for limit-induced bifurcation settings.

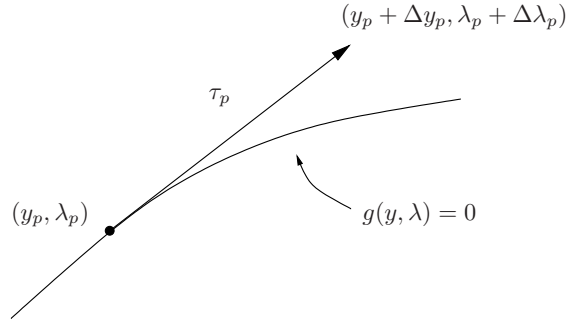


Figure 5.3: Continuation Power flow: predictor step obtained by means of tangent vector.

5.2.1 Predictor Step

At a generic equilibrium point, the following relation applies:

$$g(y_p, \lambda_p) = 0 \quad \Rightarrow \quad \left. \frac{dg}{d\lambda} \right|_p = 0 = \nabla_y g|_p \left. \frac{dy}{d\lambda} \right|_p + \left. \frac{\partial g}{\partial \lambda} \right|_p \quad (5.10)$$

and the tangent vector can be approximated by:

$$\tau_p = \left. \frac{dy}{d\lambda} \right|_p \approx \frac{\Delta y_p}{\Delta \lambda_p} \quad (5.11)$$

From (5.10) and (5.11), one has:

$$\begin{aligned} \tau_p &= -\nabla_y g|_p^{-1} \left. \frac{\partial g}{\partial \lambda} \right|_p \\ \Delta y_p &= \tau_p \Delta \lambda_p \end{aligned} \quad (5.12)$$

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

$$\Delta \lambda_p \triangleq \frac{k}{|\tau_p|} \quad \Delta y_p \triangleq \frac{k \tau_p}{|\tau_p|} \quad (5.13)$$

where $k = \pm 1$, and its sign determines the increase or the decrease of λ . Figure 5.3 presents a pictorial representation of the predictor step.

5.2.2 Corrector Step

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

$$\begin{aligned} g(y, \lambda) &= 0 \\ \rho(y, \lambda) &= 0 \end{aligned} \quad (5.14)$$

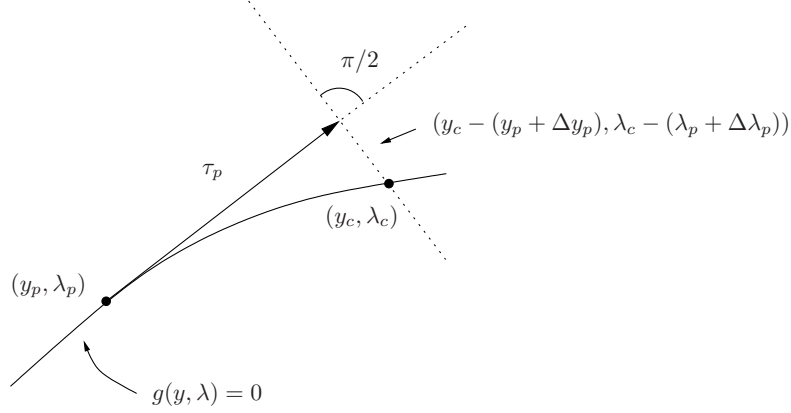


Figure 5.4: Continuation Power flow: corrector step obtained by means of perpendicular intersection.

where the solution of g 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*.

In case of perpendicular intersection, whose pictorial representation is reported in Fig. 5.4, the expression of ρ becomes:

$$\rho(y, \lambda) = \begin{bmatrix} \Delta y_p \\ \Delta \lambda_p \end{bmatrix}^T \begin{bmatrix} y_c - (y_p + \Delta y_p) \\ \lambda_c - (\lambda_p + \Delta \lambda_p) \end{bmatrix} = 0 \quad (5.15)$$

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

$$\rho(y, \lambda) = \lambda_c - \lambda_p - \Delta \lambda_p \quad (5.16)$$

or

$$\rho(y, \lambda) = y_{c_i} - y_{p_i} - \Delta y_{p_i} \quad (5.17)$$

The choice of the variable to be fixed depends on the bifurcation manifold of g , as depicted in Fig. 5.5.

5.2.3 N-1 Contingency Analysis

PSAT is provided with a N-1 contingency analysis which 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.

At this aim, it is performed a continuation power flow analysis for each line outage. Note that 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

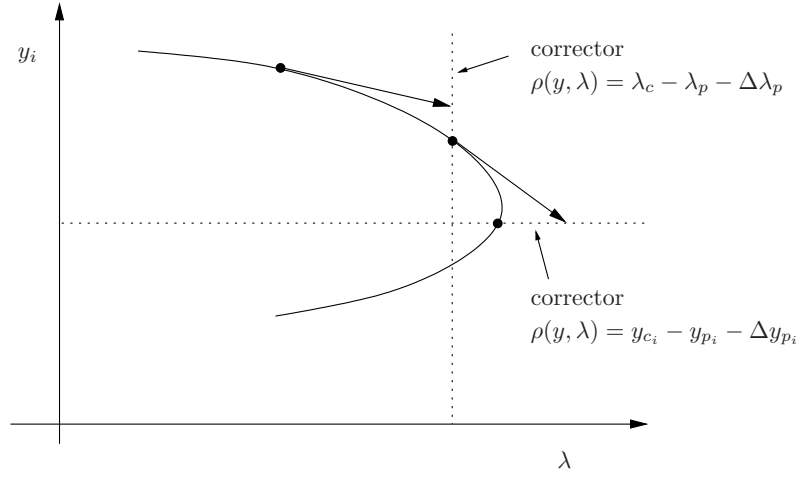


Figure 5.5: Continuation Power flow: corrector step obtained by means of local parametrization.

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.

The N-1 contingency analysis can be run for all networks for which the continuation power flow routine will work. Thus, in general, dynamic components are not supported so far.

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.

5.2.4 Graphical User Interface and Settings

Figure 5.6 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 technique 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. This feature is still experimental, as the support of dynamic components in the CPF analysis is not fully supported yet. 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 (5.3), 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 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 (5.3), while, by default, they would be excluded.

This option 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.

5.3 Examples

Figure 5.7 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 5.8 and 5.9 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 5.8 and 5.9 are obtained with and without generator reactive power limits and show a limit-induced bifurcation and a saddle-node bifurcation, respectively.

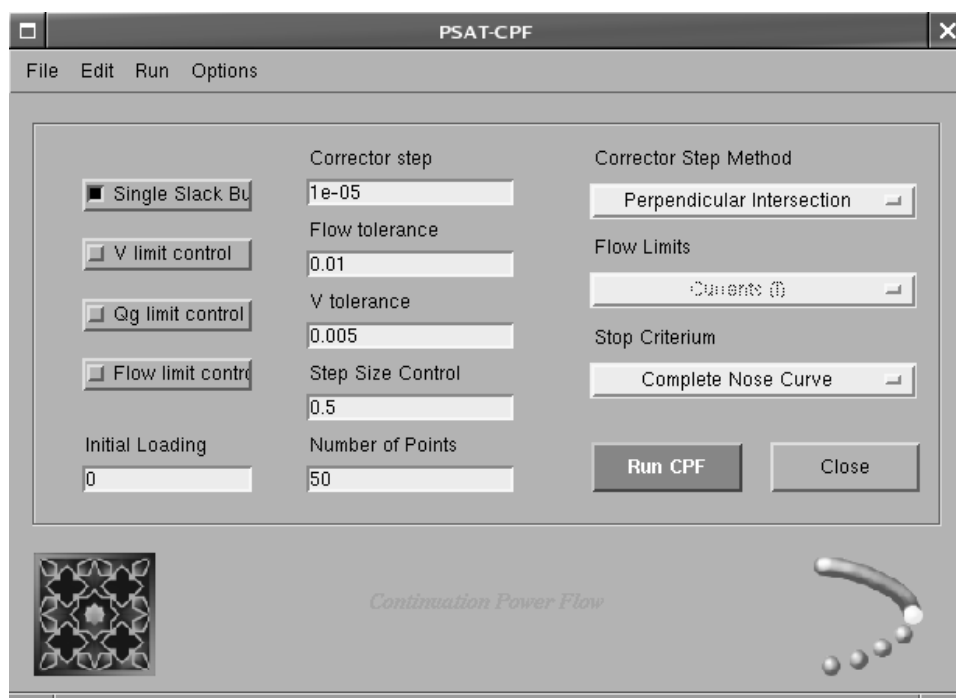


Figure 5.6: GUI for the continuation power flow settings.

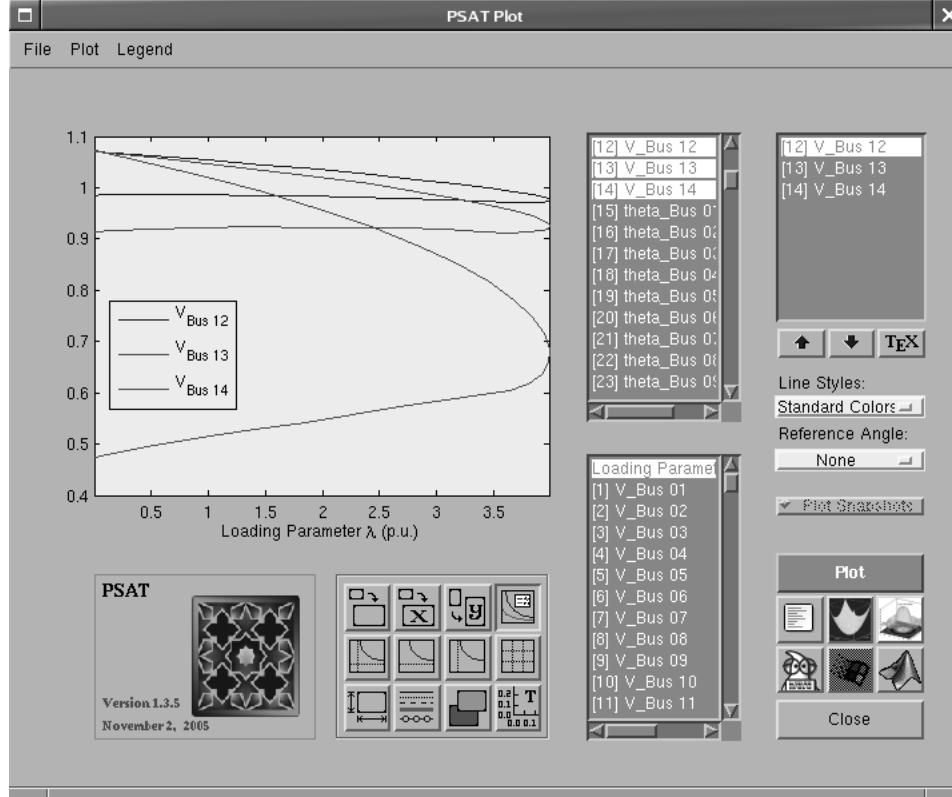


Figure 5.7: GUI for plotting CPF results. The nose curves refers to three load voltages of the IEEE 14-bus test system.

Table 5.1 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.

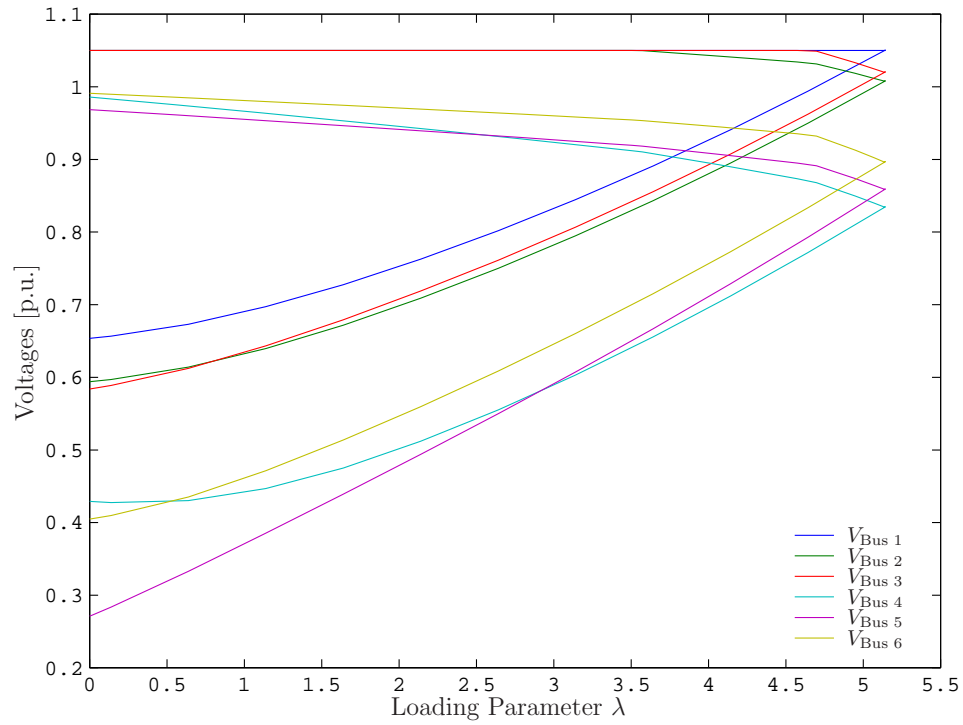


Figure 5.8: Nose curves for the 6-bus test system with generator reactive power limits. The maximum loading condition is due to a saddle limit-induced bifurcation.

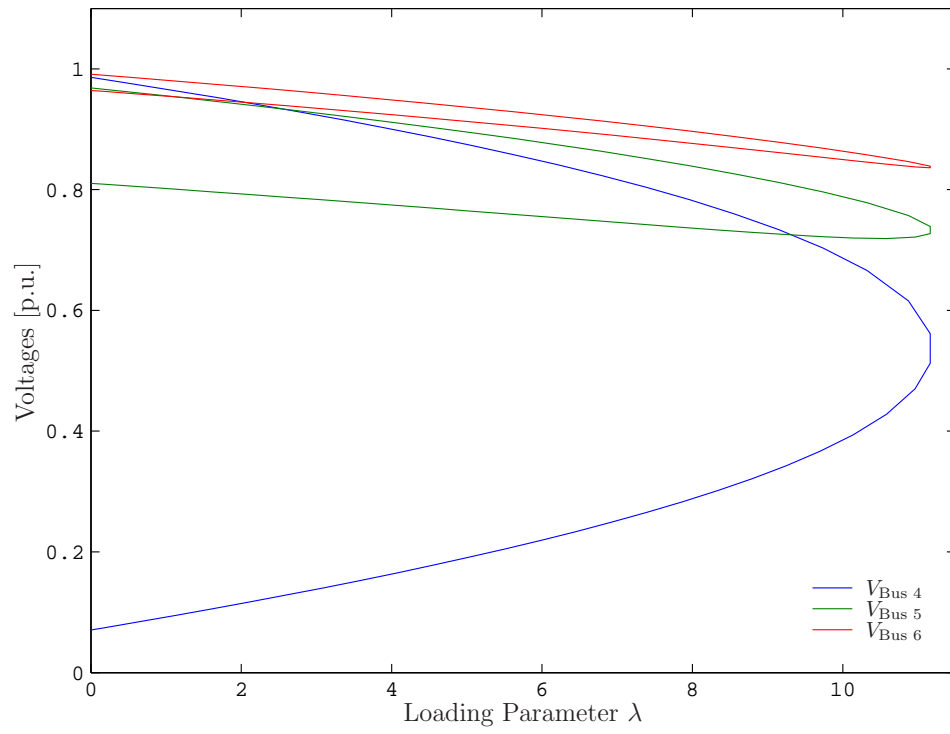


Figure 5.9: Nose curves for the 6-bus test system without generator reactive power limits. The maximum loading condition is due to a saddle-node bifurcation.

Table 5.1: N-1 Contingency Analysis Report for the 6-bus test system

N-1 CONTINGENCY ANALYSIS			
P S A T 1.3.2			
Author: Federico Milano, (c) 2002-2004			
e-mail: fmilano@thunderbox.uwaterloo.ca			
website: http://thunderbox.uwaterloo.ca/~fmilano			
File: ~/psatd/tests/d_006.mdl			
Date: 25-Nov-2004 17:28:02			
POWER FLOW LIMITS			
Line	Line Outage	Pij [p.u.]	Pij max [p.u.]
2-3	1-5	0.15013	0.22452
3-6	2-5	0.50254	0.62959
4-5	3-5	0.07867	0.11511
3-5	1-5	0.24653	0.31823
5-6	1-2	0.0199	0.02523
2-4	3-5	0.60904	0.72198
1-2	2-4	0.11245	0.1899
1-4	2-5	0.40302	0.47836
1-5	2-5	0.38453	0.50879
2-6	2-4	0.44108	0.51417
2-5	1-2	0.30954	0.36198

Chapter 6

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 [117]. 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.

6.1 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 [112, 84, 4].

In [94] and [117], the authors present a comprehensive investigation of the use of IPM for non-linear problems, and describe the application of Newton 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 [117] 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 [60], [18], [26] and [25]. In [71] and [70], 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 [79], 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.

6.2 OPF Routines

In the program 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 6.2.4 presents the Lagrangian function which is minimized by means of the IPM-NLP method.

6.2.1 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. [127]):

$$\begin{array}{llll}
 \text{Min.} & -(\Sigma C_{Di}(P_{Di}) - \Sigma C_{Si}(P_{Si})) & \rightarrow & \text{Social benefit} & (6.1) \\
 \text{s.t.} & g(\delta, V, Q_G, P_S, P_D) = 0 & \rightarrow & \text{PF equations} \\
 & 0 \leq P_S \leq P_{S_{\max}} & \rightarrow & \text{Sup. bid blocks} \\
 & 0 \leq P_D \leq P_{D_{\max}} & \rightarrow & \text{Dem. bid blocks} \\
 & |P_{ij}(\delta, V)| \leq P_{ij_{\max}} & \rightarrow & \text{Power transfer lim.} \\
 & |P_{ji}(\delta, V)| \leq P_{ji_{\max}} & \rightarrow & \\
 & Q_{G_{\min}} \leq Q_G \leq Q_{G_{\max}} & \rightarrow & \text{Gen. } Q \text{ lim.} \\
 & V_{\min} \leq V \leq V_{\max} & \rightarrow & V \text{ "security" lim.}
 \end{array}$$

where C_S and C_D are vectors of supply and demand bids in \$/MWh, respectively; Q_G stand for the generator reactive powers; V and δ represent the bus phasor voltages; P_{ij} and P_{ji} represent the powers flowing through the lines in both directions, and model system security by limiting the transmission line power flows, together with line current I_{ij} and I_{ji} thermal limits and bus voltage limits; and P_S and P_D represent bounded supply and demand power bids in MW. In this model, which is typically referred to as a security constrained OPF, P_{ij} and P_{ji} limits are 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 continuation power flow routines described in Chapter 5.

6.2.2 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]:

$$\begin{aligned}
\text{Min. } & G = -\lambda_c & (6.2) \\
\text{s.t. } & g(\delta, V, Q_G, P_S, P_D) = 0 & \rightarrow \text{PF equations} \\
& g(\delta_c, V_c, Q_{G_c}, \lambda_c, P_S, P_D) = 0 & \rightarrow \text{Max load PF eqs.} \\
& \lambda_{c_{\min}} \leq \lambda_c \leq \lambda_{c_{\max}} & \rightarrow \text{loading margin} \\
& 0 \leq P_S \leq P_{S_{\max}} & \rightarrow \text{Sup. bid blocks} \\
& 0 \leq P_D \leq P_{D_{\max}} & \rightarrow \text{Dem. bid blocks} \\
& I_{ij}(\delta, V) \leq I_{ij_{\max}} & \rightarrow \text{Thermal limits} \\
& I_{ji}(\delta, V) \leq I_{ji_{\max}} \\
& I_{ij}(\delta_c, V_c) \leq I_{ij_{\max}} \\
& I_{ji}(\delta_c, V_c) \leq I_{ji_{\max}} \\
& Q_{G_{\min}} \leq Q_G \leq Q_{G_{\max}} & \rightarrow \text{Gen. } Q \text{ limits} \\
& Q_{G_{\min}} \leq Q_{G_c} \leq Q_{G_{\max}} \\
& V_{\min} \leq V \leq V_{\max} & \rightarrow V \text{ "security" lim.} \\
& V_{\min} \leq V_c \leq V_{\max}
\end{aligned}$$

In this case, a second set of power flow equations and constraints with a subscript c is introduced to represent the system at the limit or "critical" conditions associated with the maximum loading margin λ_c in p.u., where λ is the parameter that drives the system to its maximum loading condition. 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) [20,98]. Thus, for the current and maximum loading conditions, the generator and load powers are defined as follows:

$$\begin{aligned}
P_G &= P_{G_0} + P_S \\
P_L &= P_{L_0} + P_D \\
P_{G_c} &= (1 + \lambda_c + k_{G_c})P_G \\
P_{L_c} &= (1 + \lambda_c)P_L
\end{aligned} \tag{6.3}$$

where P_{G_0} and 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_c in (6.2) are distributed among all generators. Observe that power directions (6.3) used in the voltage stability constrained OPF differ from (5.2), i.e. the power directions used in the bifurcation analysis presented in Chapter 5.

6.2.3 Multi-Objective Optimization

A multi-objective optimization is also implemented, based on what was proposed in [79], so that system security which is modeled through the use of voltage stability conditions is combined with the electricity market:

$$\begin{aligned}
 \text{Min. } & G = -\omega_1(\Sigma C_{Di}(P_{Di}) - \Sigma C_{Si}(P_{Si})) - \omega_2 \lambda_c & (6.4) \\
 \text{s.t. } & g(\delta, V, Q_G, P_S, P_D) = 0 & \rightarrow \text{PF equations} \\
 & g(\delta_c, V_c, Q_{G_c}, \lambda_c, P_S, P_D) = 0 & \rightarrow \text{Max load PF eqs.} \\
 & \lambda_{c_{\min}} \leq \lambda_c \leq \lambda_{c_{\max}} & \rightarrow \text{loading margin} \\
 & 0 \leq P_S \leq P_{S_{\max}} & \rightarrow \text{Sup. bid blocks} \\
 & 0 \leq P_D \leq P_{D_{\max}} & \rightarrow \text{Dem. bid blocks} \\
 & I_{ij}(\delta, V) \leq I_{ij_{\max}} & \rightarrow \text{Thermal limits} \\
 & I_{ji}(\delta, V) \leq I_{ji_{\max}} & \\
 & I_{ij}(\delta_c, V_c) \leq I_{ij_{\max}} & \\
 & I_{ji}(\delta_c, V_c) \leq I_{ji_{\max}} & \\
 & Q_{G_{\min}} \leq Q_G \leq Q_{G_{\max}} & \rightarrow \text{Gen. } Q \text{ limits} \\
 & Q_{G_{\min}} \leq Q_{G_c} \leq Q_{G_{\max}} & \\
 & V_{\min} \leq V \leq V_{\max} & \rightarrow V \text{ "security" lim.} \\
 & V_{\min} \leq V_c \leq V_{\max} &
 \end{aligned}$$

In the multi-objective function G , 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 (6.4) is the same as in (6.3).

6.2.4 Lagrangian Function

Internally, the program represents the previous problems (6.1), (6.2) and (6.4) in the same way, ignoring the constraints that are not used or assuming proper values for the parameters that are not defined. The following Lagrangian function is minimized:

$$\begin{aligned}
 \text{Min. } \mathcal{L} = & G - \rho^T f(\delta, V, Q_G, P_S, P_D) \\
 & - \rho_c^T f(\delta_c, V_c, Q_{G_c}, \lambda_c, P_S, P_D) \\
 & - \mu_{\lambda_c \max} (\lambda_{c \max} - \lambda_c - s_{\lambda_c \max}) \\
 & - \mu_{\lambda_c \min} (\lambda_c - s_{\lambda_c \min}) \\
 & - \mu_{P_S \max}^T (P_{S \max} - P_S - s_{P_S \max}) \\
 & - \mu_{P_S \min}^T (P_S - s_{P_S \min}) \\
 & - \mu_{P_D \max}^T (P_{D \max} - P_D - s_{P_D \max}) \\
 & - \mu_{P_D \min}^T (P_D - s_{P_D \min}) \\
 & - \mu_{I_{ij \max}}^T (I_{\max} - I_{ij} - s_{I_{ij \max}}) \\
 & - \mu_{I_{ji \max}}^T (I_{\max} - I_{ji} - s_{I_{ji \max}}) \\
 & - \mu_{I_{ijc \max}}^T (I_{\max} - I_{ijc} - s_{I_{ijc \max}}) \\
 & - \mu_{I_{jic \max}}^T (I_{\max} - I_{jic} - s_{I_{jic \max}}) \\
 & - \mu_{Q_G \max}^T (Q_{G \max} - Q_G - s_{Q_G \max}) \\
 & - \mu_{Q_G \min}^T (Q_G - Q_{G \min} - s_{Q_G \min}) \\
 & - \mu_{Q_{Gc \max}}^T (Q_{G \max} - Q_{Gc} - s_{Q_{Gc \max}}) \\
 & - \mu_{Q_{Gc \min}}^T (Q_{Gc} - Q_{G \min} - s_{Q_{Gc \min}}) \\
 & - \mu_{V_{\max}}^T (V_{\max} - V - s_{V_{\max}}) \\
 & - \mu_{V_{\min}}^T (V - V_{\min} - s_{V_{\min}}) \\
 & - \mu_{V_c \max}^T (V_{\max} - V_c - s_{V_c \max}) \\
 & - \mu_{V_c \min}^T (V_c - V_{\min} - s_{V_c \min}) - \mu_s \left(\sum_i \ln s_i \right)
 \end{aligned} \tag{6.5}$$

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

6.3 OPF Settings

Figure 6.1 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 [117]. The parameters and the results of OPF

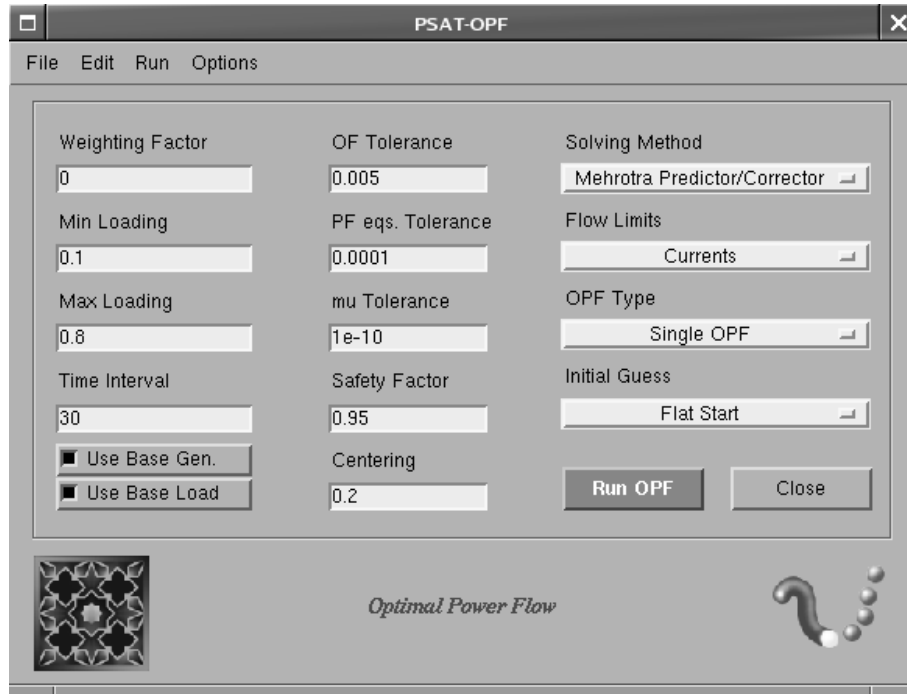


Figure 6.1: GUI for the optimal power flow.

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

6.4 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, \LaTeX , Excel) and displayed with the selected viewer (see Section 26.4 for details). For example, the plain text power flow solution for the 6-bus test system with $\omega = 0$ (standard OPF) is as follows:

```
OPTIMAL POWER FLOW REPORT
(Standard OPF)
```

```
P S A T  1.3.3
```

```
Author:  Federico Milano, (c) 2002-2005
```

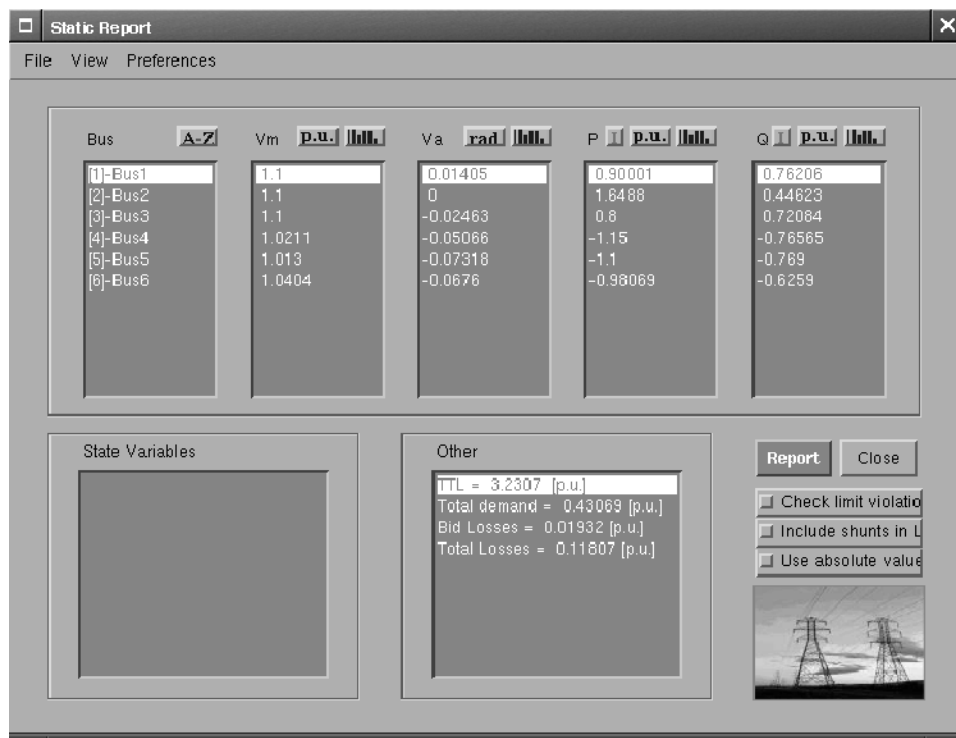



Figure 6.2: GUI for displaying OPF results.

e-mail: fmilano@thunderbox.uwaterloo.ca
 website: <http://thunderbox.uwaterloo.ca/~fmilano>

File: ~/psatd/tests/d_006_md1
 Date: 10-Mar-2005 19:41:11

NETWORK STATISTICS

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

SOLUTION STATISTICS

Objective Function [\$ /h]:	-121.6493
Active Limits:	8
Number of Iterations:	13
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	0.65773	0.001	0.001	20	0
Bus2	0	0.001	25	25	0.17662
Bus3	0	0.001	20	20	2.0968

POWER DEMANDS

Bus	mu min	Pd min [MW]	Pd [MW]	Pd max [MW]	mu max
Bus4	0	0.001	25	25	2.304
Bus5	0	0.001	10	10	0.42491
Bus6	0	0.001	8.0694	20	0

REACTIVE POWERS

Bus	mu min	Qg min [MVar]	Qg [MVar]	Qg max [MVar]	mu max
Bus2	0	-150	76.206	150	0
Bus1	0	-150	44.6233	150	0
Bus3	0	-150	72.0844	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	1.36	0.01405
Bus2	0	0.9	1.1	1.1	0.69913	0
Bus3	0	0.9	1.1	1.1	0.29865	-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	44.6233	9.0204	0	-0.04872	-812
Bus2	164.8754	76.206	8.9805	0	0	-1481
Bus3	80	72.0844	9.1455	0	0.07648	-732
Bus4	-115	-76.665	9.563	0.39306	0.20737	1100
Bus5	-110	-77	9.6535	0.40762	0.29043	1062
Bus6	-98.0693	-62.6898	9.4284	0.21472	0.23945	925

FLOWS IN TRANSMISSION LINES

From bus	To bus	I _{ij} [p.u.]	I _{ij} max [p.u.]	mu I _{ij}
Bus2	Bus3	0.11693	0.3082	0
Bus3	Bus6	0.731	1.3973	0
Bus4	Bus5	0.07148	0.1796	0
Bus3	Bus5	0.33729	0.6585	0
Bus5	Bus6	0.11578	0.2	0
Bus2	Bus4	0.84775	1.374	0
Bus1	Bus2	0.08127	0.2591	0
Bus1	Bus4	0.49408	0.9193	0
Bus1	Bus5	0.39214	0.8478	0
Bus2	Bus6	0.4327	0.9147	0
Bus2	Bus5	0.35683	0.7114	0

FLOWS IN TRANSMISSION LINES

From bus	To bus	I _{ji} [p.u.]	I _{ji} max [p.u.]	mu I _{ji}
Bus3	Bus2	0.10451	0.3082	0
Bus6	Bus3	0.74506	1.3973	0
Bus5	Bus4	0.06342	0.1796	0
Bus5	Bus3	0.36729	0.6585	0
Bus6	Bus5	0.0635	0.2	0
Bus4	Bus2	0.8581	1.374	0
Bus2	Bus1	0.06232	0.2591	0
Bus4	Bus1	0.51836	0.9193	0
Bus5	Bus1	0.42224	0.8478	0
Bus6	Bus2	0.45115	0.9147	0
Bus5	Bus2	0.3779	0.7114	0

TOTALS

TOTAL LOSSES [MW]: 11.807

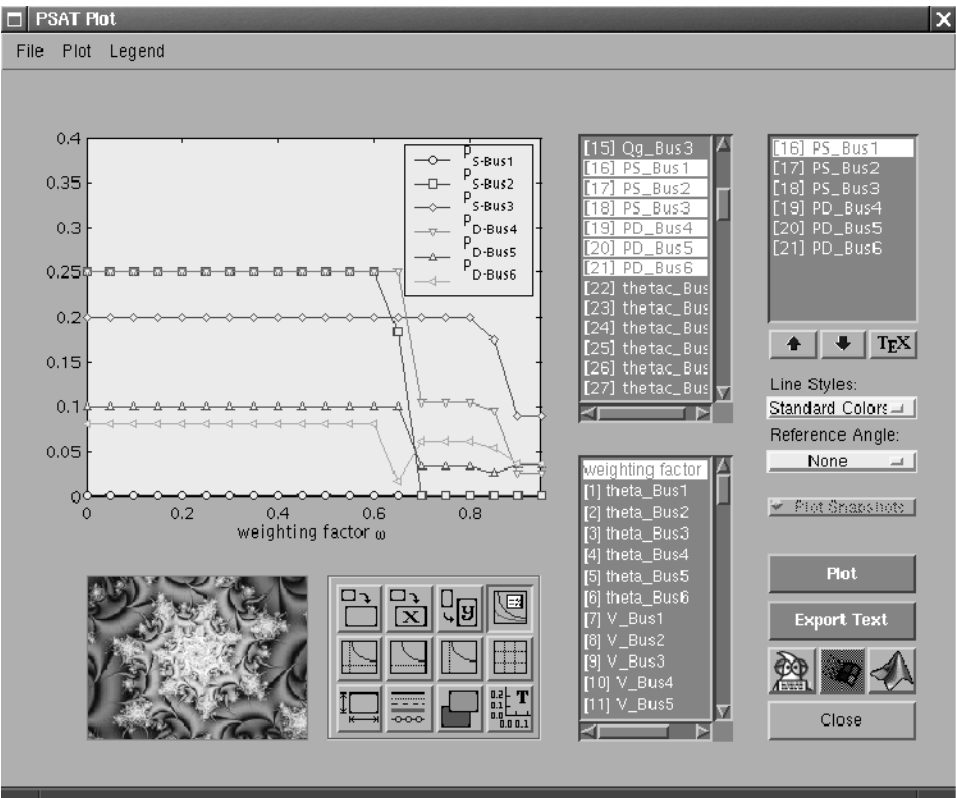


Figure 6.3: GUI for plotting OPF Pareto sets.

BID LOSSES [MW]	1.932
TOTAL DEMAND [MW] :	43.0694
TTL [MW] :	323.0694
IMO PAY [\$ /h] :	62.1219

Figure 6.3 depicts the graphical user interface for plotting the Pareto set, which can be obtained by setting a vector of values for the weighting factor ω . The GUI permits 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 7

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) [101, 59], and for three different types of power flow Jacobian matrices (QV sensitivity analysis) [128].

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

7.1 Small Signal Stability Analysis

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

$$\begin{aligned} \dot{x} &= f(x, y) \\ 0 &= g(x, y) \end{aligned} \tag{7.1}$$

where x is the vector of the state variables and y the vector of the algebraic variables, which in PSAT are only the voltages amplitudes V and phases θ .

The state matrix A_S is thus computed by manipulating the complete Jacobian matrix A_C , that is defined by the linearization of the DAE system equations (7.1):

$$\begin{bmatrix} \Delta \dot{x} \\ 0 \end{bmatrix} = \begin{bmatrix} \nabla_x f & \nabla_y f \\ \nabla_x g & \nabla_y g \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = [A_C] \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \tag{7.2}$$

Hereinafter, the Jacobian matrices which form the A_C matrix, will be referred to

as follows:

$$\begin{aligned} F_x &\triangleq \nabla_x f \\ F_y &\triangleq \nabla_y f \\ G_x &\triangleq \nabla_x g \\ G_y &\triangleq \nabla_y g \end{aligned} \quad (7.3)$$

where 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 J_{LF} and J_{LFD} , which are described in the next section.

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

$$A_S = F_x - F_y G_y^{-1} G_x \quad (7.4)$$

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 V and W be the right and the left eigenvector matrices respectively, such that $\Lambda = W A_S V$ and $W = V^{-1}$, then the participation factor p_{ij} of the i^{th} state variable to the j^{th} eigenvalue can be defined as:

$$p_{ij} = \frac{w_{ij} v_{ji}}{w_j^t v_j} \quad (7.5)$$

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

$$p_{ij} = \frac{|w_{ij}| |v_{ji}|}{\sum_{k=1}^n |w_{jk}| |v_{kj}|} \quad (7.6)$$

No normalization of the participation factors is performed.

The state matrix in (7.4) 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 bilinear transformation is performed:

$$A_Z = (A_S + \rho I_n)(A_S - \rho I_n)^{-1} \quad (7.7)$$

where ρ is a weighting factor, that in the program is set to 8. Even though more expensive, 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.

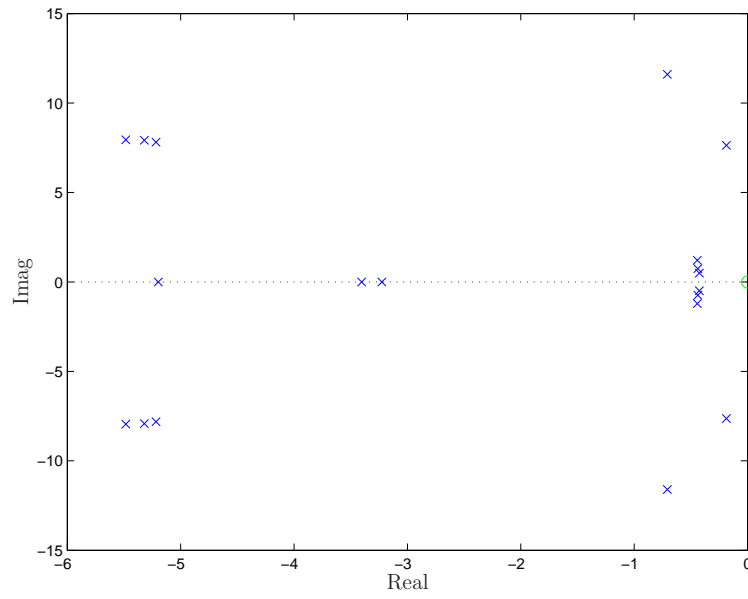


Figure 7.1: Eigenvalue Analysis: S -domain.

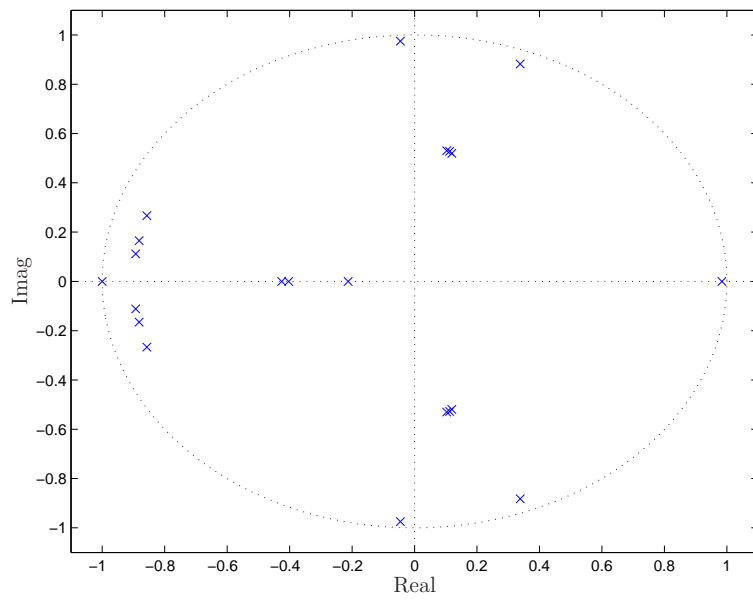


Figure 7.2: Eigenvalue Analysis: Z -domain.

7.1.1 Example

The following Figs. 7.1 and 7.2, and the small signal stability report 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. For the static and dynamic data of the network, refer to Subsection F.3.

EIGENVALUE REPORT

P S A T 1.3.4

Author: Federico Milano, (c) 2002-2005
e-mail: fmilano@thunderbox.uwaterloo.ca
website: http://thunderbox.uwaterloo.ca/~fmilano

File: ~/psatd/tests/d_009.mdl
Date: 20-Sep-2005 18:52:53

STATE MATRIX EIGENVALUES

Eigvalue	Most Associated States	Real part	Imag. Part	Frequency
Eig As 1	vm_Exc_1	-1000	0	0
Eig As 2	vm_Exc_1	-1000	0	0
Eig As 3	vm_Exc_3	-1000	0	0
Eig As 4	delta_Syn_3, omega_Syn_3	-0.72015	12.7454	2.0285
Eig As 5	delta_Syn_3, omega_Syn_3	-0.72015	-12.7454	2.0285
Eig As 6	omega_Syn_2, delta_Syn_2	-0.19077	8.3657	1.3314
Eig As 7	omega_Syn_2, delta_Syn_2	-0.19077	-8.3657	1.3314
Eig As 8	vr1_Exc_2, vf_Exc_2	-5.4818	7.9465	1.2647
Eig As 9	vr1_Exc_2, vf_Exc_2	-5.4818	-7.9465	1.2647
Eig As 10	vr1_Exc_1, vf_Exc_1	-5.2171	7.8127	1.2434
Eig As 11	vr1_Exc_1, vf_Exc_1	-5.2171	-7.8127	1.2434
Eig As 12	vr1_Exc_3, vf_Exc_3	-5.3181	7.9197	1.2605
Eig As 13	vr1_Exc_3, vf_Exc_3	-5.3181	-7.9197	1.2605
Eig As 14	e1d_Syn_2	-5.178	0	0
Eig As 15	e1d_Syn_3	-3.3996	0	0
Eig As 16	e1q_Syn_1, vr2_Exc_1	-0.44307	1.212	0.19289
Eig As 17	e1q_Syn_1, vr2_Exc_1	-0.44307	-1.212	0.19289
Eig As 18	e1q_Syn_1, e1q_Syn_2	-0.43845	0.74006	0.11778
Eig As 19	e1q_Syn_1, e1q_Syn_2	-0.43845	-0.74006	0.11778
Eig As 20	e1q_Syn_3, vr2_Exc_3	-0.42503	0.49669	0.07905
Eig As 21	e1q_Syn_3, vr2_Exc_3	-0.42503	-0.49669	0.07905
Eig As 22	delta_Syn_1	0	0	0
Eig As 23	omega_Syn_1	0	0	0
Eig As 24	e1d_Syn_1	-3.2258	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	0.01789	0	0.00014
Eig As 11	0.00015	0.00015	0.01789	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	0.23861	0	0.00068
Eig As 17	0.00021	0.00021	0.23861	0	0.00068
Eig As 18	0.00025	0.00025	0.24352	0	0.0007
Eig As 19	0.00025	0.00025	0.24352	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	0.36233	0.36233	0	0	0.09334
Eig As 23	0.36233	0.36233	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	0.01326	0.00609	0.04918	0.04918
Eig As 7	0.30889	0.01326	0.00609	0.04918	0.04918
Eig As 8	0.00102	0.01367	0.00171	0.0003	0.0003
Eig As 9	0.00102	0.01367	0.00171	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	0.16014	0.01106	0.00041	0.00041
Eig As 17	0.00068	0.16014	0.01106	0.00041	0.00041
Eig As 18	0.0007	0.18935	0.00948	0.00018	0.00018
Eig As 19	0.0007	0.18935	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	0.00384	0.00046	0.00017	0.34841	0.10666
Eig As 11	0.00384	0.00046	0.00017	0.34841	0.10666
Eig As 12	0.01082	0.00274	1e-05	0.06428	0.01876
Eig As 13	0.01082	0.00274	1e-05	0.06428	0.01876
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	0.09174	0.00622	0.00031	0.02477	0.17649
Eig As 17	0.09174	0.00622	0.00031	0.02477	0.17649
Eig As 18	0.05439	0.00511	0.00019	0.02221	0.18919
Eig As 19	0.05439	0.00511	0.00019	0.02221	0.18919
Eig As 20	0.32212	0.0365	0	0.00023	0.0021
Eig As 21	0.32212	0.0365	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	0.38763	0.11196	0.38115
Eig As 9	0.00665	0.00013	0.38763	0.11196	0.38115
Eig As 10	0.33901	2e-05	0.02092	0.0065	0.02065
Eig As 11	0.33901	2e-05	0.02092	0.0065	0.02065
Eig As 12	0.06244	2e-05	0.02233	0.00661	0.02202
Eig As 13	0.06244	2e-05	0.02233	0.00661	0.02202
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	0.03708	0.00021	0.01904	0.11574	0.02431
Eig As 17	0.03708	0.00021	0.01904	0.11574	0.02431
Eig As 18	0.03415	0.00016	0.0204	0.14627	0.0264
Eig As 19	0.03415	0.00016	0.0204	0.14627	0.0264
Eig As 20	0.00036	8e-05	0.01429	0.10722	0.01856
Eig As 21	0.00036	8e-05	0.01429	0.10722	0.01856
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	0.03708	0.0106	0.03607
Eig As 9	1e-05	0.03708	0.0106	0.03607
Eig As 10	4e-05	0.05721	0.0176	0.05593
Eig As 11	4e-05	0.05721	0.0176	0.05593
Eig As 12	0.0001	0.3454	0.10129	0.33712
Eig As 13	0.0001	0.3454	0.10129	0.33712
Eig As 14	2e-05	0.00028	0.00508	3e-05
Eig As 15	1e-05	0	0.01006	3e-05
Eig As 16	0.00012	0.00994	0.06744	0.01417
Eig As 17	0.00012	0.00994	0.06744	0.01417
Eig As 18	4e-05	0.0054	0.04361	0.00787
Eig As 19	4e-05	0.0054	0.04361	0.00787
Eig As 20	0.00018	0.03164	0.2681	0.04642
Eig As 21	0.00018	0.03164	0.2681	0.04642
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

7.2 Power Flow Sensitivity Analysis

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

1. J_{LF} , which is obtained from the static equations (10.1) of power flows in transmission lines and transformers, and is generally defined as the *standard* power flow Jacobian matrix.
2. J_{LFV} , which is the complete Jacobian matrix of the power flow equations of the system.
3. J_{LFD} , which is computed from the complete matrix A_C :

$$J_{LFD} = J_{LFV} - G_x F_x^{-1} F_y \quad (7.8)$$

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. $J_{LFV} = G_y$. If there are other algebraic variables, these are removed from the Jacobian matrix as follows:

$$J_{LFV} = G_y^{(m_1, m_1)} - G_y^{(m_1, m_2)} [G_y^{(m_2, m_2)}]^{-1} G_y^{(m_2, m_1)} \quad (7.9)$$

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 G_y corresponds to the active and reactive power equation gradients. Thus G_y is as follows:

$$G_y = \begin{bmatrix} G_y^{(m_1, m_1)} & G_y^{(m_1, m_2)} \\ G_y^{(m_2, m_1)} & G_y^{(m_2, m_2)} \end{bmatrix} \quad (7.10)$$

The matrix J_{LFD} can be defined in a similar way as J_{LFV} in (7.9).

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

$$J_{LF} = \begin{bmatrix} J_{P\theta} & J_{PV} \\ J_{Q\theta} & J_{QV} \end{bmatrix} \quad (7.11)$$

In case of the *standard* Jacobian matrix 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:

$$\begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} = \begin{bmatrix} J_{P\theta} & J_{PV} \\ J_{Q\theta} & J_{QV} \end{bmatrix} \begin{bmatrix} \Delta \theta \\ \Delta V \end{bmatrix} \quad (7.12)$$

Then, the reduced matrix is defined as follows:

$$J_{LFr} = J_{QV} - J_{Q\theta} J_{P\theta}^{-1} J_{PV} \quad (7.13)$$

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

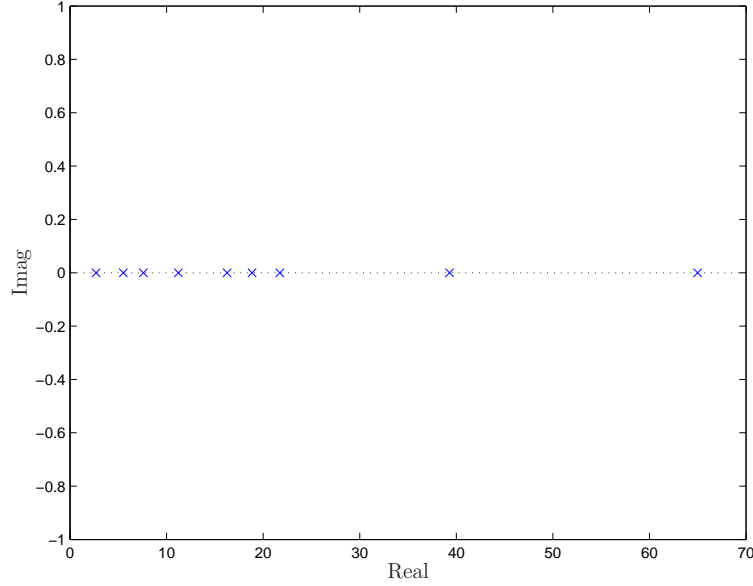


Figure 7.3: Eigenvalue Analysis: QV sensitivity.

That can thus be used for a QV sensitivity analysis, if one assumes that $\Delta P = 0$ and that the sub-matrix $J_{P\theta}$ is non-singular:

$$\Delta Q = J_{LFr} \Delta V \quad (7.14)$$

For J_{LFV} and J_{LFD} , the reduced matrix is defined as in (7.13), even though it lacks the rigorous physical meaning of (7.14).

7.2.1 Example

The following Fig. 7.3 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 J_{LF} matrix, but since the system has only constant power loads and generators and there are no dynamic components, one has $J_{LF} = J_{LFV} = J_{LFD}$. The report shows five high eigenvalues ($\mu = 999$), which represent the constant voltage buses of the five generators.

EIGENVALUE REPORT

P S A T 1.3.0

Author: Federico Milano, (c) 2002-2004
e-mail: fmilano@thunderbox.uwaterloo.ca
website: <http://thunderbox.uwaterloo.ca/~fmilano>

File: ~/psatd/tests/d_014.mdl
 Date: 16-Mar-2004 14:47:53

EIGENVALUES OF THE COMPLETE POWER JACOBIAN MATRIX

Eigevalue	Real part		Imaginary Part
Eig Jlfv1	64.9803	0	
Eig Jlfv2	39.2929	0	
Eig Jlfv3	21.7272	0	
Eig Jlfv4	18.8536	0	
Eig Jlfv5	16.2706	0	
Eig Jlfv6	2.6984	0	
Eig Jlfv7	5.5274	0	
Eig Jlfv8	7.6017	0	
Eig Jlfv9	11.2207	0	
Eig Jlfv10	999	0	
Eig Jlfv11	999	0	
Eig Jlfv12	999	0	
Eig Jlfv13	999	0	
Eig Jlfv14	999	0	

PARTICIPATION FACTORS (Euclidean norm)

	Bus 01	Bus 02	Bus 03	Bus 04	Bus 05
Eig Jlfv1	0	0	0	0.53957	0.45377
Eig Jlfv2	0	0	0	3e-05	0.00065
Eig Jlfv3	0	0	0	0.07934	0.15554
Eig Jlfv4	0	0	0	0.00038	0.00048
Eig Jlfv5	0	0	0	0.28196	0.31696
Eig Jlfv6	0	0	0	0.00823	0.00397
Eig Jlfv7	0	0	0	0.00246	0.00136
Eig Jlfv8	0	0	0	1e-05	0
Eig Jlfv9	0	0	0	0.08802	0.06726
Eig Jlfv10	1	0	0	0	0
Eig Jlfv11	0	1	0	0	0
Eig Jlfv12	0	0	1	0	0
Eig Jlfv13	0	0	0	0	0
Eig Jlfv14	0	0	0	0	0

PARTICIPATION FACTORS (Euclidean norm)

	Bus 06	Bus 07	Bus 08	Bus 09	Bus 10
Eig Jlfv1	0	0.00653	0	0.00012	1e-05
Eig Jlfv2	0	0.1549	0	0.61494	0.21327
Eig Jlfv3	0	0.49063	0	0.00357	0.21787
Eig Jlfv4	0	0.00013	0	0.00015	0.00443
Eig Jlfv5	0	0.02216	0	0.04693	0.16526
Eig Jlfv6	0	0.0691	0	0.19882	0.2394
Eig Jlfv7	0	0.01672	0	0.03173	0.11939
Eig Jlfv8	0	3e-05	0	4e-05	0.0364
Eig Jlfv9	0	0.23981	0	0.10371	0.00396
Eig Jlfv10	0	0	0	0	0
Eig Jlfv11	0	0	0	0	0
Eig Jlfv12	0	0	0	0	0
Eig Jlfv13	1	0	0	0	0
Eig Jlfv14	0	0	1	0	0

PARTICIPATION FACTORS (Euclidean norm)

	Bus 11	Bus 12	Bus 13	Bus 14
Eig Jlfv1	0	0	0	0
Eig Jlfv2	0.00759	0	0.00012	0.0085

Eig Jlfv3	0.05254	3e-05	0.00018	0.00031
Eig Jlfv4	0.00192	0.17934	0.766	0.04716
Eig Jlfv5	0.15559	0.0022	0.00499	0.00395
Eig Jlfv6	0.11026	0.01886	0.03232	0.31905
Eig Jlfv7	0.13155	0.32558	0.16032	0.2109
Eig Jlfv8	0.11208	0.46399	0.03394	0.3535
Eig Jlfv9	0.42848	0.01	0.00213	0.05664
Eig Jlfv10	0	0	0	0
Eig Jlfv11	0	0	0	0
Eig Jlfv12	0	0	0	0
Eig Jlfv13	0	0	0	0
Eig Jlfv14	0	0	0	0

STATISTICS

NUMBER OF BUSES	14
# OF EIGS WITH $\text{Re}(\mu) < 0$	0
# OF EIGS WITH $\text{Re}(\mu) > 0$	14
# OF REAL EIGS	14
# OF COMPLEX PAIRS	0
# OF ZERO EIGS	0

7.3 Graphical User Interface

Figure 7.4 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 of participation factor map); the Jacobian matrix (state matrix A_S or one of the power flow Jacobian matrices J_{LFr} , J_{LFVr} or J_{LFDr}); and the number and the kind of eigenvalues to be computed. The “Graph” and the “Report” pushbuttons 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.

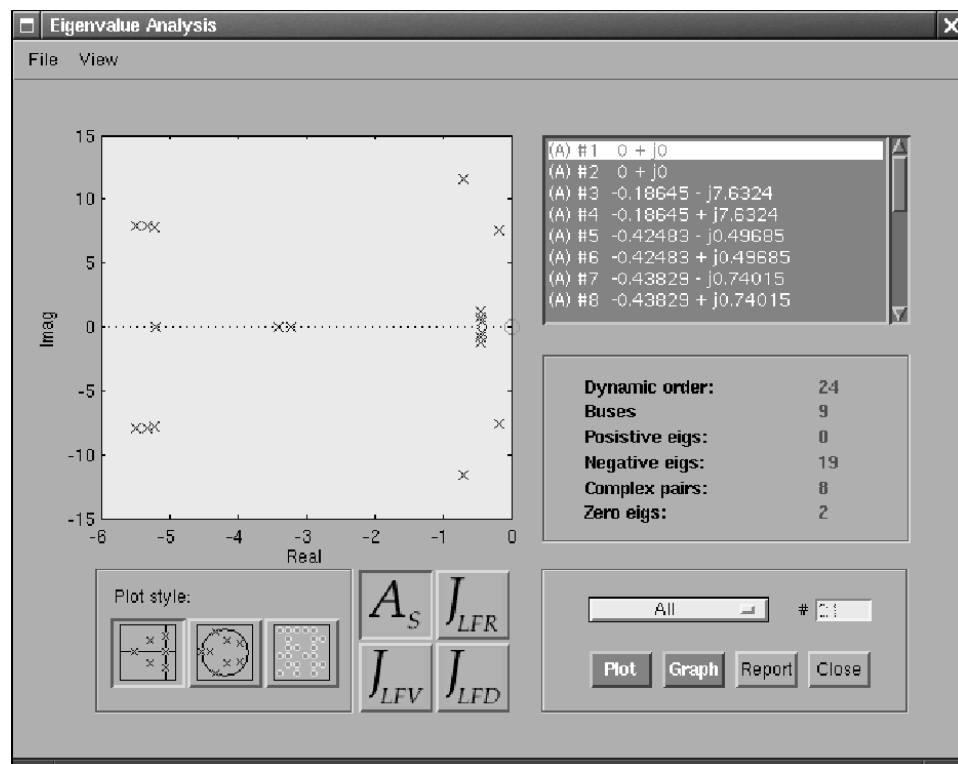


Figure 7.4: GUI for the small signal stability analysis.

Chapter 8

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 specific points during the time simulations. How to include disturbances is also described in this chapter. 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.

Observe that several programs for power system analysis 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. Then PSAT makes use of static and/or dynamic data depending on the kind of the currently running simulation.

8.1 Integration Methods

Two integration methods are available, i.e. forward Euler and trapezoidal rule, which 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 many books (e.g. [16]).

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

$$\begin{aligned} 0 &= f_n(x(t + \Delta t), y(t + \Delta t), f(t)) \\ 0 &= g(x(t + \Delta t), y(t + \Delta t)) \end{aligned} \tag{8.1}$$

where f and g represent the differential and algebraic equations and f_n is a function that depends on the integration method. Equations (8.1) are nonlinear and their solution is obtained by means of a Newton-Raphson technique which in turn consists of computing iteratively the increment Δx^i and Δy^i of the state and algebraic

variables and updating the actual variables:

$$\begin{aligned} \begin{bmatrix} \Delta x^i \\ \Delta y^i \end{bmatrix} &= -[A_c^i]^{-1} \begin{bmatrix} f_n^i \\ g^i \end{bmatrix} \\ \begin{bmatrix} x^{i+1} \\ y^{i+1} \end{bmatrix} &= \begin{bmatrix} x^i \\ y^i \end{bmatrix} + \begin{bmatrix} \Delta x^i \\ \Delta y^i \end{bmatrix} \end{aligned} \quad (8.2)$$

where A_c^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 technique repeated again. Figure 8.1 depicts the block diagram of the time domain integration. For sake of completeness, the following sections report the expressions of A_c^i and f_n^i for each method.

8.1.1 Forward Euler Method

The forward Euler integration method is a first order method. It is generally faster but less accurate than the trapezoidal method. At a generic iteration i , A_c^i and f_n^i are as follows:

$$\begin{aligned} A_c^i &= \begin{bmatrix} I_n - \Delta t F_x^i & -\Delta t F_y^i \\ G_x^i & G_y^i \end{bmatrix} \\ f_n^i &= x^i - x(t) - \Delta t f^i \end{aligned} \quad (8.3)$$

where I_n is the identity matrix of the same dimension of the dynamic order of the system, and the other matrices are the Jacobian matrices of the algebraic differential equations, i.e. $F_x = \nabla_x f$, $F_y = \nabla_y f$, $G_x = \nabla_x g$ and $G_y = \nabla_y g$.

8.1.2 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 , A_c^i and f_n^i are as follows:

$$\begin{aligned} A_c^i &= \begin{bmatrix} I_n - 0.5\Delta t F_x^i & -0.5\Delta t F_y^i \\ G_x^i & J_{LFV}^i \end{bmatrix} \\ f_n^i &= x^i - x(t) - 0.5\Delta t(f^i + f(t)) \end{aligned} \quad (8.4)$$

where the notation is the same as in (8.3).

8.2 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

¹Although the initial time could be assigned any value, it is recommended to use $t_0 = 0$ as other values have not been tested. In any case it must be $t_0 > 0$.

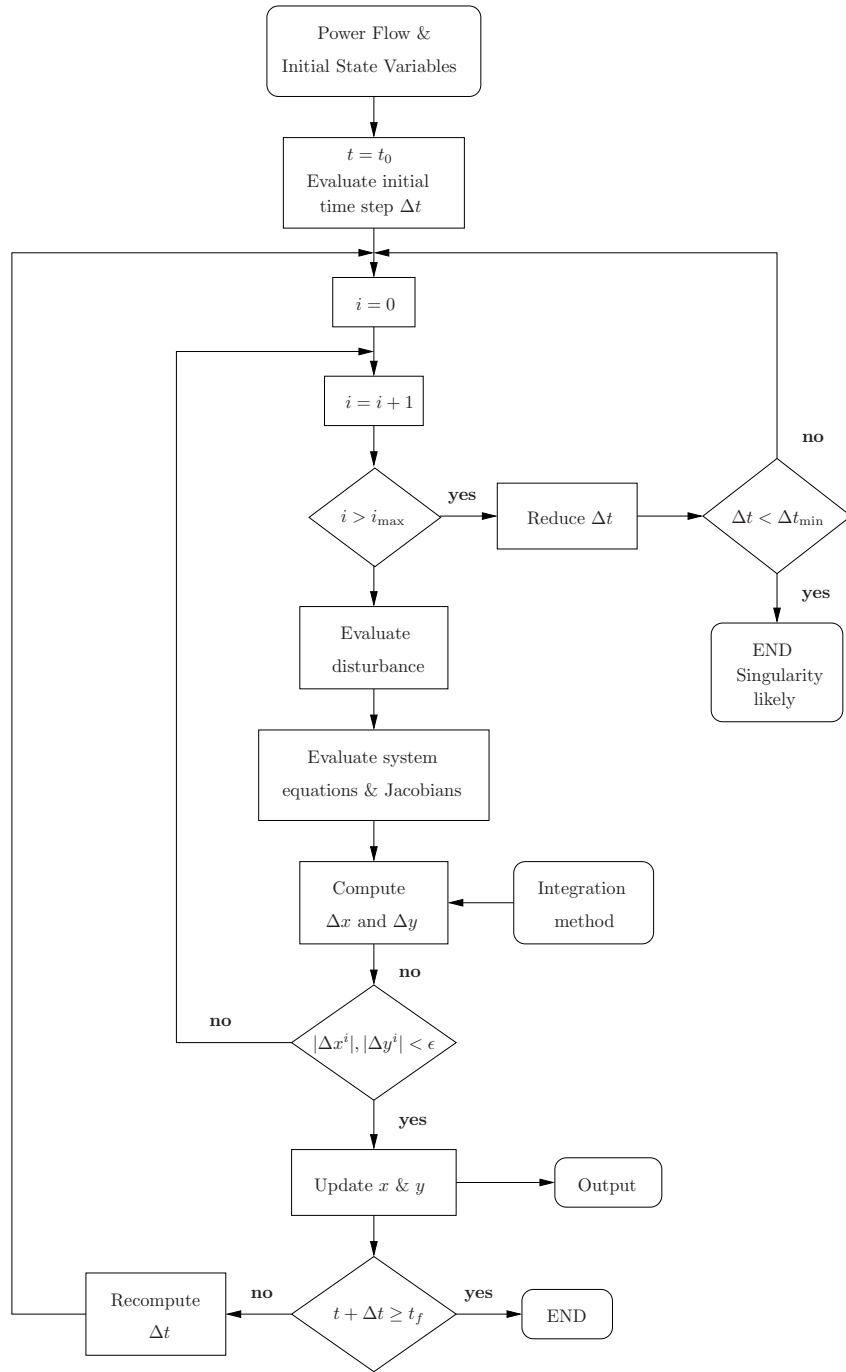


Figure 8.1: Time domain integration block diagram.

technique 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. 8.2. The following options are available for time domain simulations:

Fixed Time Step: one can enable the use of a fixed time step. this can be useful for “critical” simulations where the automatic time step flaws. If the option of the fixed time step is disabled, PSAT will compute 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 a high number of state variables.

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

Integration 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: 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]: 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): enforce the use of the Center of Inertia for synchronous machines. See Section 15.1.9 for details.

Convert PQ bus to Z: if this option is enabled, PQ buses are converted to constant impedances right before beginning the time domain simulation. This will help convergence if there are fault occurrences and breaker interventions during the time domain simulation. Refer to Sections 10.6 and 14.1 for details on the conversion of PQ loads to constant impedances.

Plot during Simulation: to enforce this option will generate a graphic of selected variables (see item *Plotting Variables*) during time domain simulations. it is typically a time consuming operation and it is disabled by default.

Plotting Variables: 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 8.3 for more details on output variable selection.

Update Simulink during TD: 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 won’t work if on Octave and for the command line usage of PSAT.

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

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

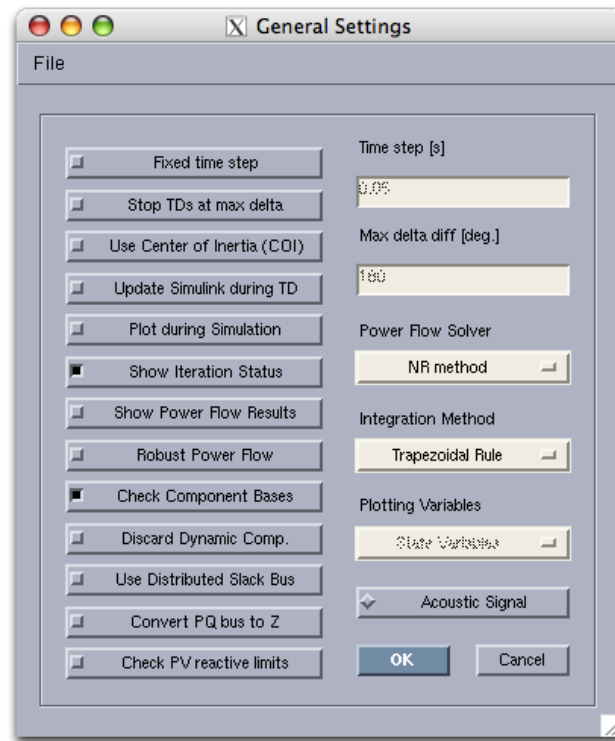


Figure 8.2: GUI for general settings.

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.

8.3 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. 8.3). By this GUI, the user can select any of the following variables:

1. State variables;
2. Bus voltage magnitudes and angles;

3. Bus voltage angles;
4. Generator mechanical powers and field voltages;
5. AVR reference voltages;
6. Over-excitation limiter currents.
7. Active and reactive power injections at buses;
8. Active and reactive power flows in transmission lines;

The selection must be done **after** running the power flow analysis and **before** running the time domain simulation.

The GUI allows selecting variables one by one or by packages. When using the command line version of PSAT, selection can be done 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 magnitudes.
- from (DAE.n + Bus.n + 1) to (DAE.n + 2*Bus.n) bus voltage angles.
- 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 $j-i$.

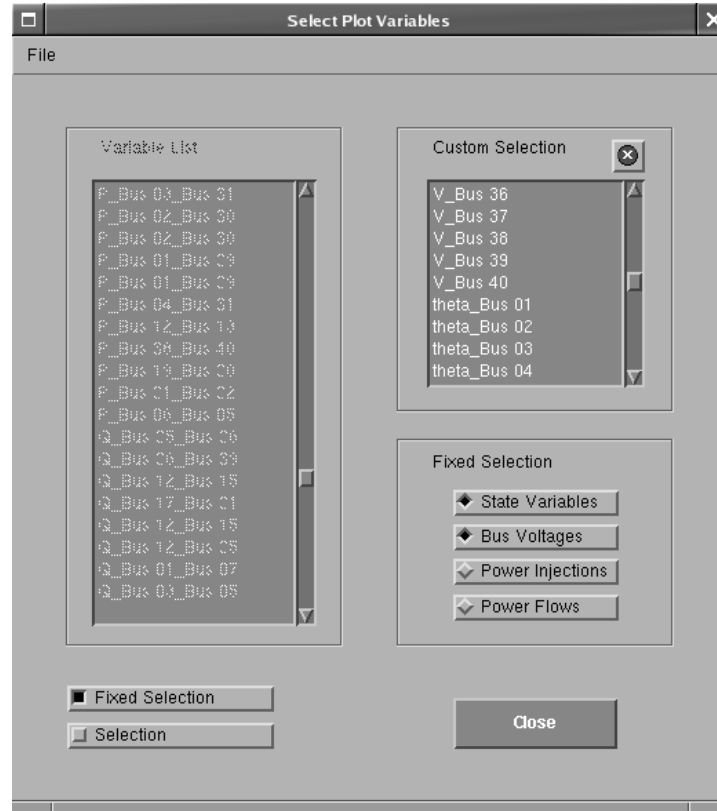


Figure 8.3: GUI for plot variable selection.

- 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+3, DAE.n+DAE.m+4];
```

The assignment must be done **after** running the power flow analysis and **before** running the time domain simulation. If this assignment is done in a function, remember to declare as **global** all needed structures.

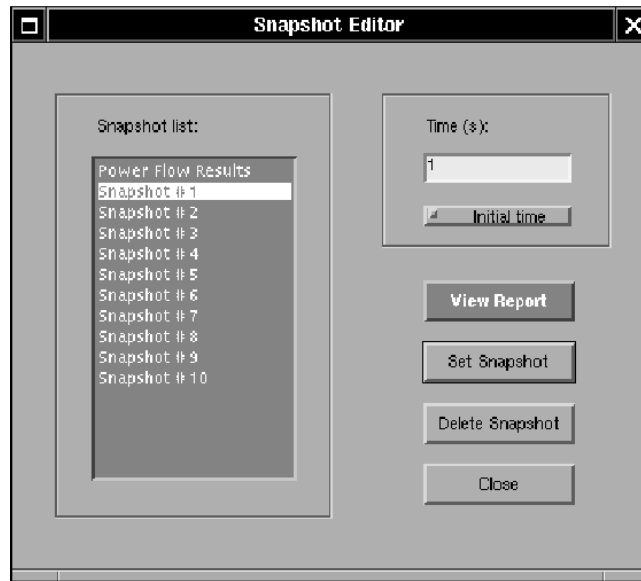


Figure 8.4: Snapshot GUI.

8.4 Snapshots

Figure 8.4 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 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 <Ctrl-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 <Ctrl-w> in the main window).

8.5 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 12. Observe that one does not need to load a perturbation file/function when using faults and/or breakers models.⁴

Generic disturbances are supported by means of user defined functions.⁵ 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.⁶ 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.

⁴One 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.

⁵Step 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.

⁶Observe that it may be necessary to call other functions. For example, after modifying a transmission line impedance, one has to call `fm.y` in order to rebuild the admittance matrix.

8.6 Examples

Figure 8.5 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) [101]. 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:

```
>> 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).

Figures 8.6, 8.7, and 8.8 depict generator speeds, generator rotor angles and bus voltages 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 reclosed at $t = 4$ s. The data for this system are reported as well in Appendix F.3.

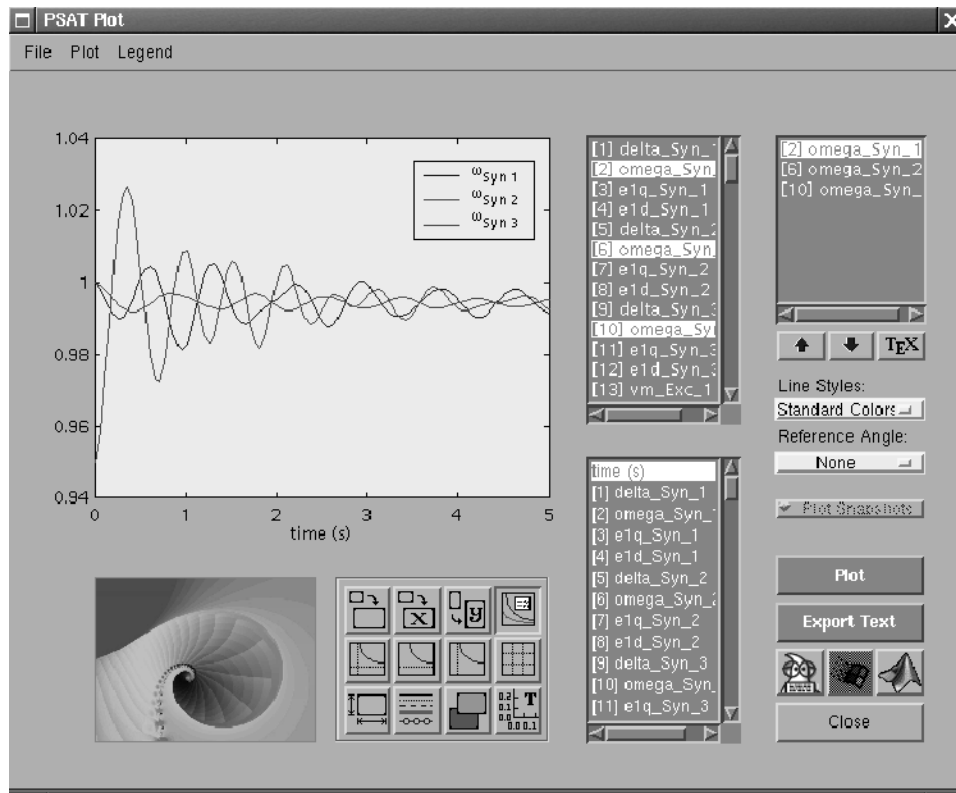


Figure 8.5: GUI for plotting time domain simulations. In this example, the speeds refer to the 9-bus test with IV order generator models and AVRs type II. The perturbation is obtained by varying the speed of generator 2 at the MATLAB prompt ($\omega_2(t_0) = 0.95$ p.u.).

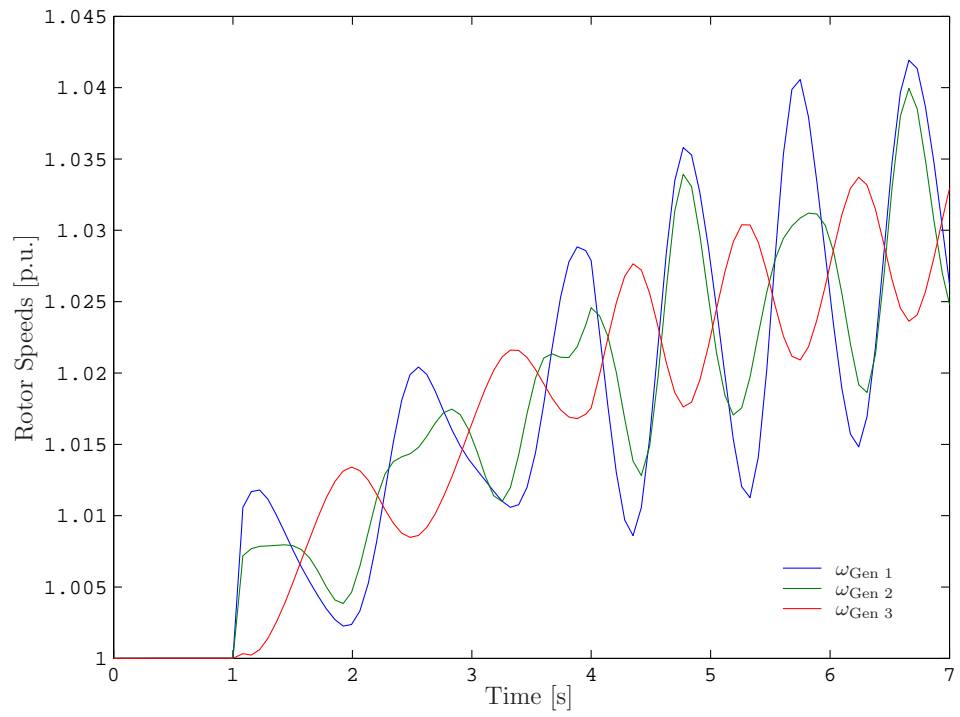


Figure 8.6: Generator speeds for the 9-bus test system with II order generator models and a fault applied at bus 7.

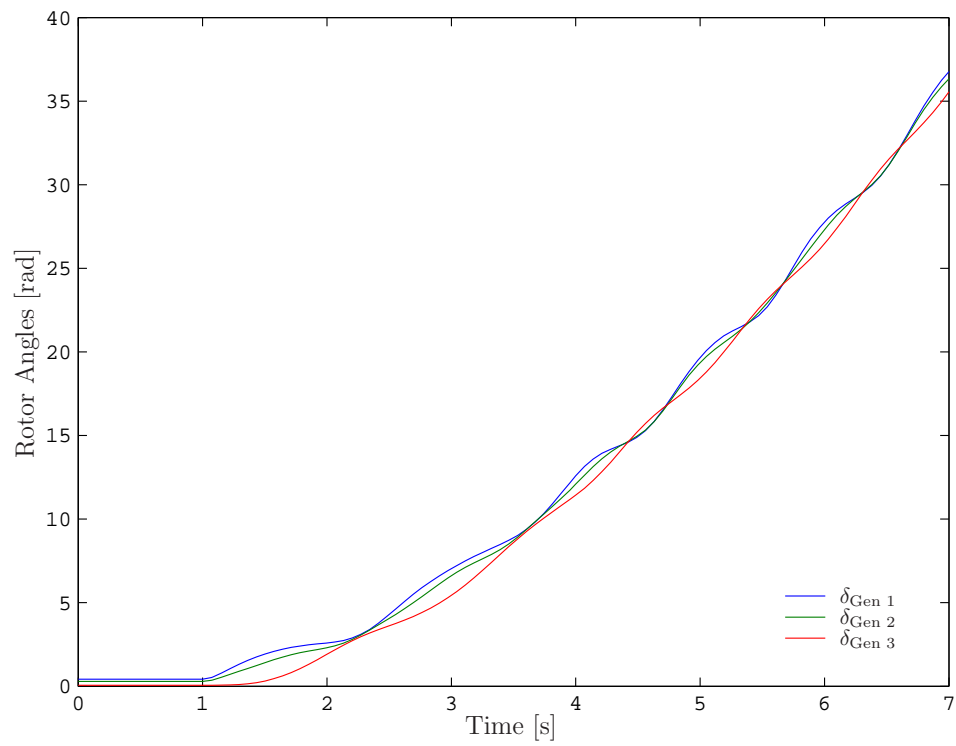


Figure 8.7: Generator rotor angles for the 9-bus test system with II order generator models and a fault applied at bus 7.

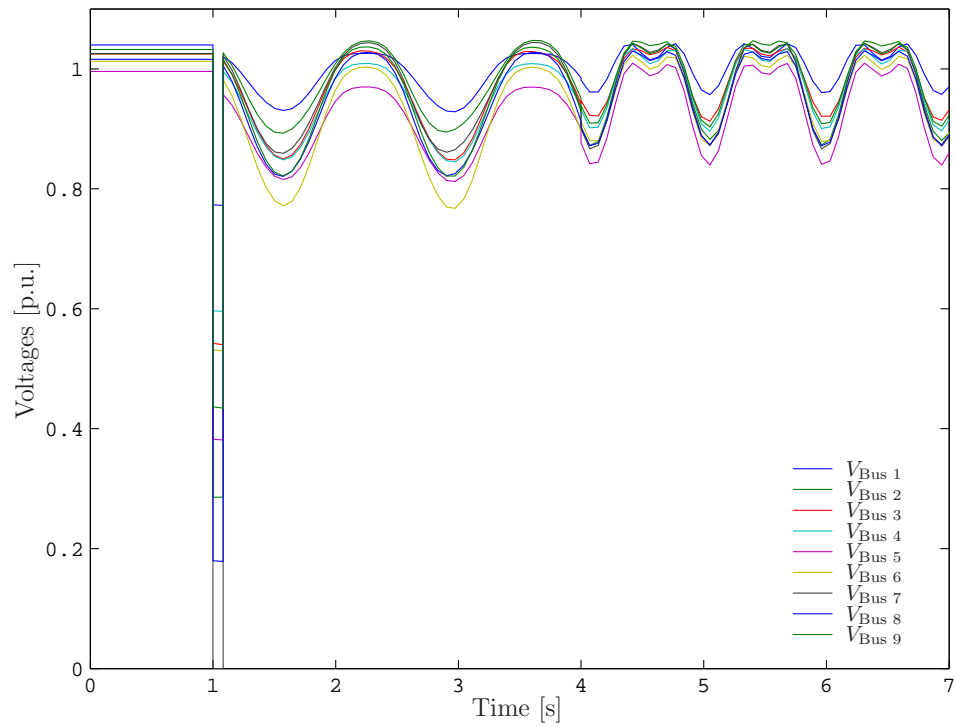


Figure 8.8: Bus voltages for the 9-bus test system with II order generator models and a fault applied at bus 7.

Chapter 9

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 [42]. 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.

9.1 Linear Static State Estimation

This section briefly describes basic concepts of power system static state estimation based on what was proposed in [102] and [38].

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

$$z = h(x) + \epsilon \quad (9.1)$$

where:

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

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

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

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

Equation (9.1) is typically solved by means of a Newton-Raphson technique [102, 3, 85]. Using devices able to provide voltage and current phasors, such as PMUs, yields a linear relationship between state variables and measurements variables, as follows:

$$z = Hx + \epsilon \quad (9.2)$$

where H ($H \in \mathbb{R}^{m \times n}$) is the “state” matrix of the system. Typically $m > n$, and the solution of (9.2) is obtained by a least mean square technique [115].

By splitting the vector z into the $m_V \times 1$ voltage and $m_I \times 1$ current subvectors, z_V and z_I , and the vector x into the $n_M \times 1$ and $n_C \times 1$ non-measured subvectors, V_M and V_C , relationship (9.2) becomes

$$\begin{bmatrix} z_V \\ z_I \end{bmatrix} = \begin{bmatrix} I & 0 \\ Y_{IM} & Y_{IC} \end{bmatrix} \begin{bmatrix} V_M \\ V_C \end{bmatrix} + \begin{bmatrix} \epsilon_V \\ \epsilon_C \end{bmatrix} \quad (9.3)$$

where I is the identity matrix, and Y_{IM} , Y_{IC} are submatrices whose elements are series and shunt admittances of the network branches. Neglecting shunts, the matrix H is as follows:

$$H = \begin{bmatrix} I & 0 \\ M_{IB}Y_{BB}A_{MB}^T & M_{IB}Y_{BB}A_{CB}^T \end{bmatrix} \quad (9.4)$$

where M_{IB} is the $m_I \times b$ measurement-to-branch incidence matrix associated with the current phasor measurements, Y_{BB} is the $b \times b$ diagonal matrix of the branch admittances, and A_{MB} and A_{CB} are the $n_M \times b$ and $n_C \times b$ calculated node-to-branch incidence submatrices, respectively [11, 37].

9.2 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. 9.1.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. 9.1.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. 9.1.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).

9.3 Algorithms

9.3.1 Depth First

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

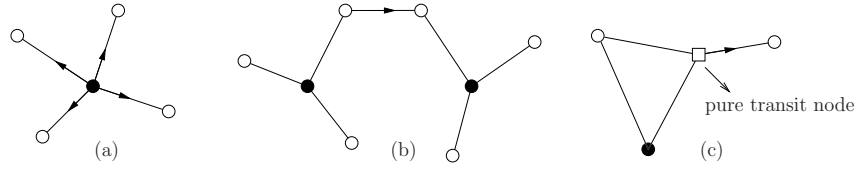


Figure 9.1: PMU placement rules.

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. 9.2.¹

9.3.2 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).

9.3.3 Bisecting Search Method

Figures 9.3 and 9.4 depict the flowchart of the bisecting search method and the pseudo-code of the simulated annealing procedure. Refer to [11] for the complete description of this method.

9.3.4 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. 9.5 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. 9.6. 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. 9.7. 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.

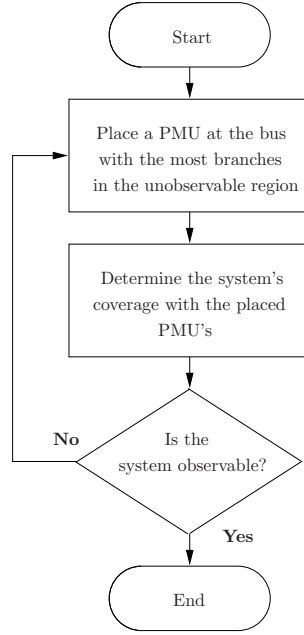


Figure 9.2: Flowchart of the Graph Theoretic Procedure.

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

9.3.5 Single Shot Security N Algorithm

This method was proposed in [42]. The algorithm is based only on topological rules, and determines a single spanning tree, as illustrated in Fig. 9.8.

9.3.6 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 [42] 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 9.9 and 9.10 depict the algorithms for obtaining the $N-1$ security placement proposed in [42]. The first method is a slightly different version of the recursive technique described

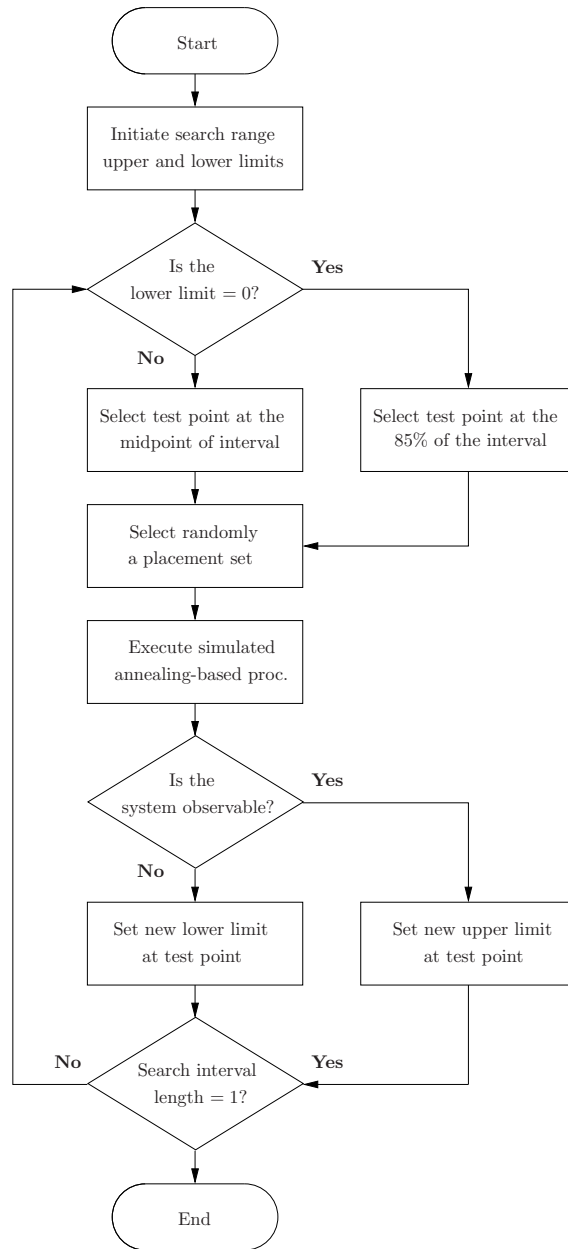


Figure 9.3: Flowchart of the Bisecting Search.

```

begin
  evaluate coverage of PMU placement set  $S$ 
   $E := N - \text{number of buses in the observed region}$ 
   $T := 15$ 
   $M := \min\{0.002\binom{N}{\nu_{\text{test}}}, M_{\text{max}}\}$ 
  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 9.4: Pseudo-code of the simulated Annealing Algorithm.

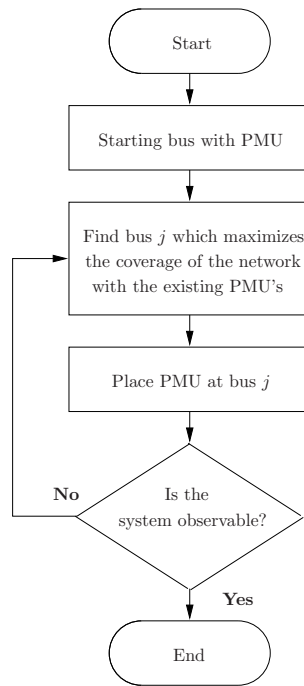


Figure 9.5: Recursive N Security Method.

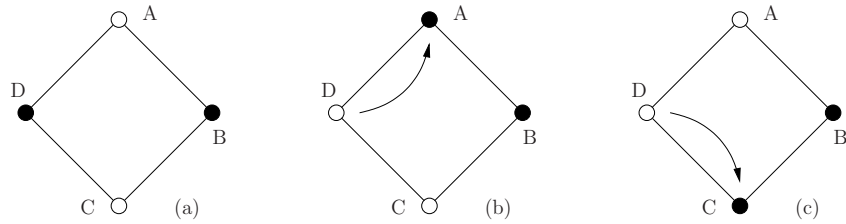


Figure 9.6: Search of alternative placement sets.

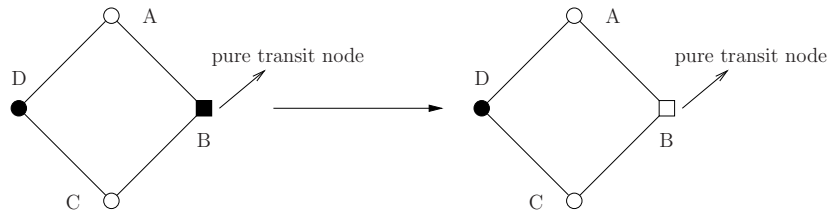


Figure 9.7: Pure transit node filtering.

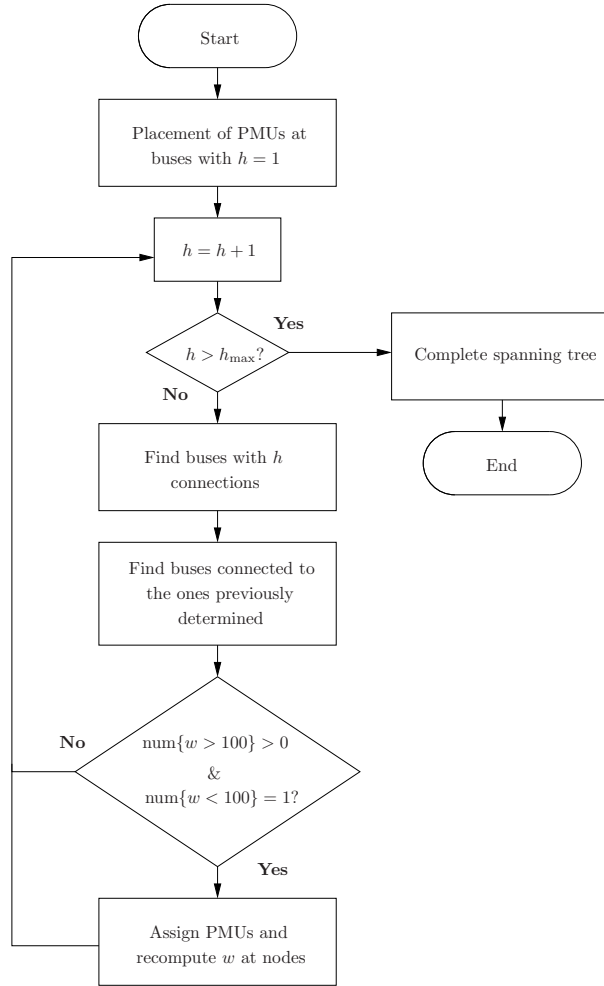


Figure 9.8: Single-Shot N Security Method.

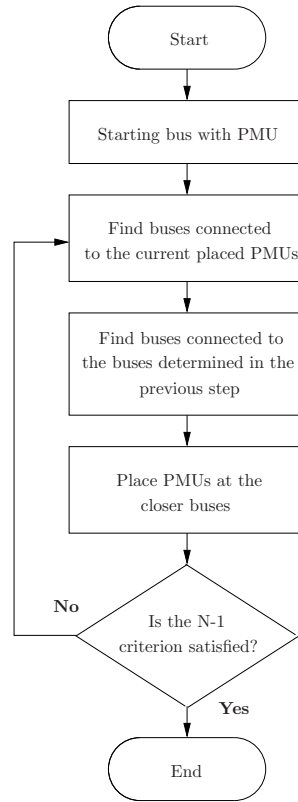


Figure 9.9: Recursive N-1 Security Method.

in Section 9.3.4, whereas the second method is a variant of the algorithm described in Section 9.3.5.

9.4 PMU Placement GUI and Settings

Figure 9.11 depicts the GUI for PMU placement, which allows to select the PMU placement method and enable to write result in a report file. The listboxes 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.

9.4.1 *Example*

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

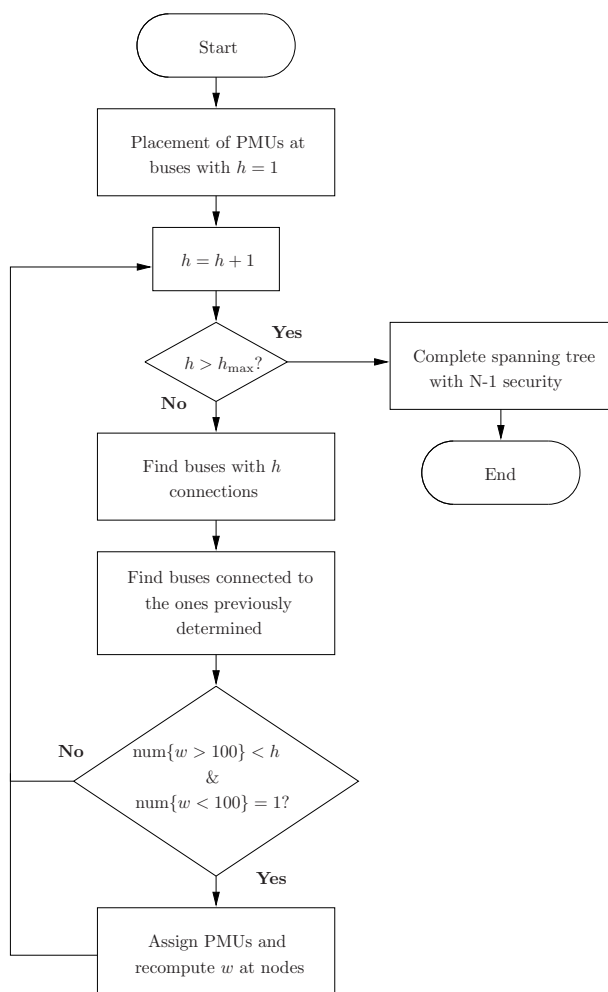


Figure 9.10: Single Shot N-1 Security Method.

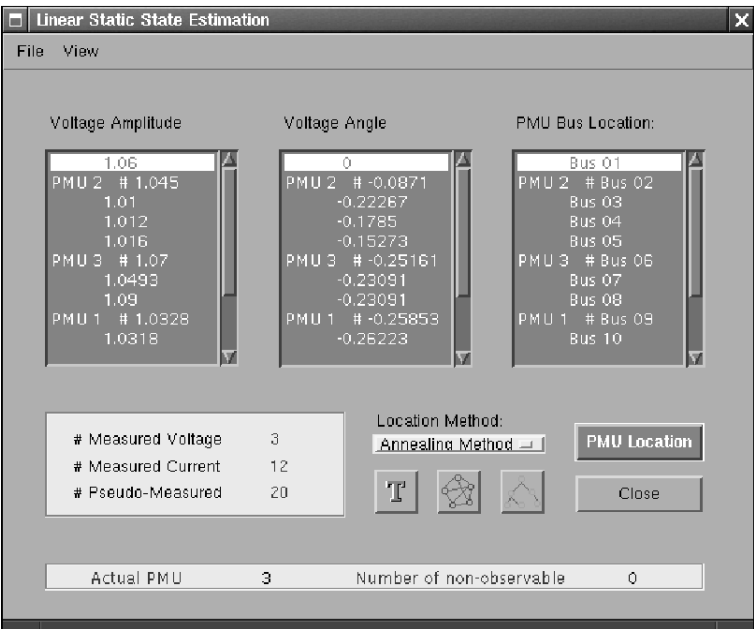


Figure 9.11: GUI for the PMU placement methods.

```
PMU PLACEMENT REPORT

P S A T  1.3.0

Author:  Federico Milano, (c) 2002-2004
e-mail:  fmilano@thunderbox.uwaterloo.ca
website: http://thunderbox.uwaterloo.ca/~fmilano

File:    ~/psatd/tests/d_014.mdl
Date:    15-Mar-2004 17:39:21

Placement Method:  Annealing Method
Elapsed Time:      0h   0m   0.59293s

STATISTICS

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

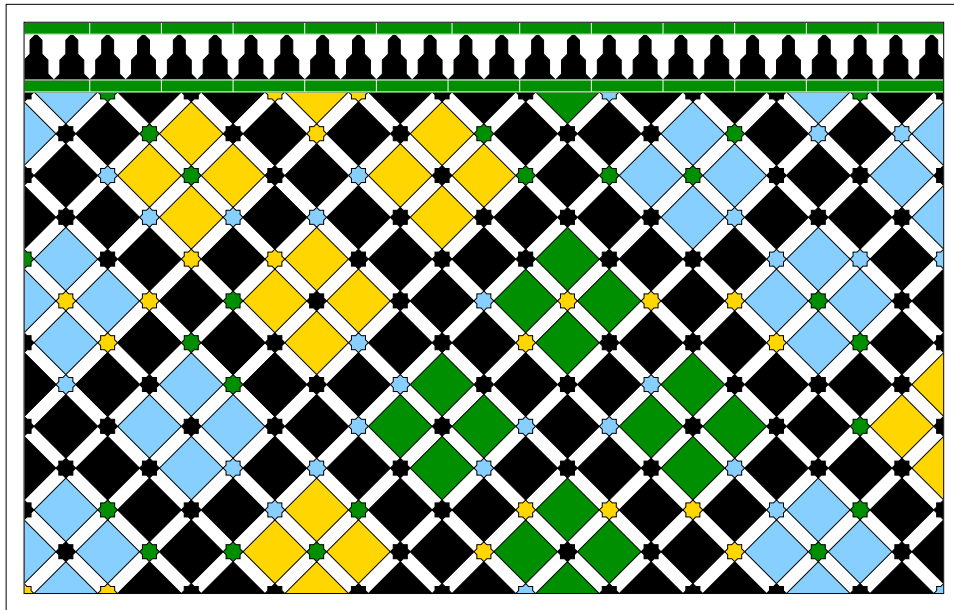
PMU PLACEMENT

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 10

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, and interchange areas.

10.1 Bus

The network topology is defined by the “bus” components, whose data format is depicted in Table 10.1.¹ 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 components are defined in the structure **Bus**, as follows:

¹In this table and in the following tables of this chapter, fields marked with a † are optional.

Table 10.1: 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	A_i	Area number (<i>not used yet...</i>)	int
† 6	R_i	Region number (<i>not used yet...</i>)	int

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.
7. **Q1**: reactive power absorbed from the network by loads.
8. **island**: indexes of island buses.
9. **names**: bus names.

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**.

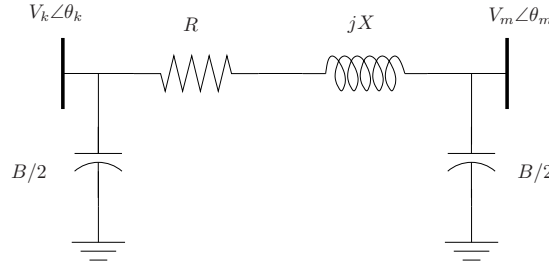
PSAT is component oriented, i.e. standard components can be connected to any bus in any number and type. The only exception is the slack generator (**SW**) that has to be unique for each bus. Refer to Chapter 22 for a detailed description of component connection rules in PSAT.

10.2 Transmission Line

Fig. 10.1 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:

$$\begin{aligned}
 P_k &= V_k^2(g_{km} + g_{k0}) - V_k V_m(g_{km} \cos(\theta_k - \theta_m) + b_{km} \sin(\theta_k - \theta_m)) \\
 Q_k &= -V_k^2(b_{km} + b_{k0}) - V_k V_m(g_{km} \sin(\theta_k - \theta_m) - b_{km} \cos(\theta_k - \theta_m)) \\
 P_m &= V_m^2(g_{km} + g_{m0}) - V_k V_m(g_{km} \cos(\theta_k - \theta_m) - b_{km} \sin(\theta_k - \theta_m)) \\
 Q_m &= -V_m^2(b_{km} + b_{m0}) + V_k V_m(g_{km} \sin(\theta_k - \theta_m) + b_{km} \cos(\theta_k - \theta_m))
 \end{aligned} \tag{10.1}$$

Transmission lines are defined in the structure **Line**, which is used also for transformers (see Section 10.3). The user can define data in absolute values or in 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 10.2 depicts the data format of transmission lines. I_{\max} , P_{\max} and S_{\max} define the limits for currents,

Figure 10.1: Transmission line π circuit.

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 5 and 6 for details.

1. **con**: transmission line data.
2. **n**: total number of lines.
3. **Y**: admittance matrix of the network.
4. **fr**: indexes of buses at which lines begin.
5. **to**: indexes of buses at which lines end.
6. **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 10.3. Transmission line data contained in the structure **Lines** are organized as follows:

1. **con**: data chart of the **Lines** components.
2. **n**: total number of alternative lines.
3. **bus1**: indexes of buses k at which the lines begin.
4. **bus2**: indexes of buses m at which the lines end.
5. **u**: connection status.

10.3 Transformers

Two kinds of static transformers can be defined, i.e. two-winding transformers and three-winding transformers. Refer to Chapter 17 for models of regulating transformers.

Table 10.2: 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	$\{0, 1\}$

Table 10.3: 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	$\{0, 1\}$

Table 10.4: 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_n	Voltage rating	kV
5	f_n	Frequency rating	Hz
6	-	<i>not used</i>	-
7	k_T	Primary and secondary 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	$\{0, 1\}$

10.3.1 Two-Winding Transformers

Two-winding transformers are modeled as series reactances without iron losses and their equations are included in (10.1). Table 10.4 depicts the transformer data format which is included in the structure `Line`. 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 $k_T \neq 0$, 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.

10.3.2 Three-Winding Transformers

Three-winding transformers are internally modeled as three two-winding transformers in a Y connection, as depicted in Fig. 10.2. 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:

$$\begin{aligned}
 \bar{z}_{12} &= \bar{z}_1 + \bar{z}_2 \\
 \bar{z}_{13} &= \bar{z}_1 + \bar{z}_3 \\
 \bar{z}_{23} &= \bar{z}_2 + \bar{z}_3
 \end{aligned} \tag{10.2}$$

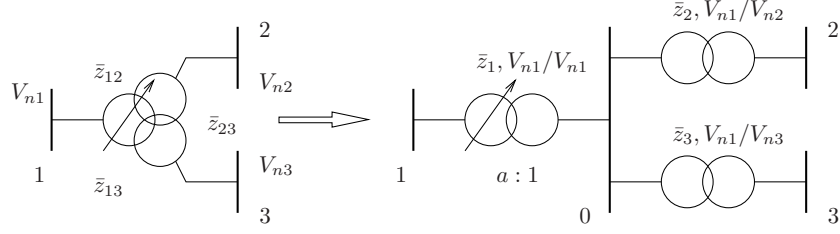


Figure 10.2: Three-winding transformer equivalent circuit.

Thus, one has:

$$\begin{aligned}\bar{z}_1 &= (\bar{z}_{12} + \bar{z}_{13} - \bar{z}_{23})/2 \\ \bar{z}_2 &= (\bar{z}_{12} + \bar{z}_{23} - \bar{z}_{13})/2 \\ \bar{z}_3 &= (\bar{z}_{13} + \bar{z}_{23} - \bar{z}_{12})/2\end{aligned}\tag{10.3}$$

Three-winding transformers are defined in the structure `Twt`, which has only the `con` field. Table 10.5 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.

10.4 $V\theta$ and Slack Generator

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

$$\begin{aligned}V &= V_0 \\ \theta &= \theta_0\end{aligned}\tag{10.4}$$

Each network must contain at least one slack generator. The angle θ_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.² Table 10.6 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_{g0} and γ are mandatory and the following additional equation holds:

$$P = (1 + \gamma k_G) P_{g0}\tag{10.5}$$

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 structure `SW`, as follows:

1. `con`: slack generator data.

²Observe that PSAT allows defining several networks in the same data file. One slack bus must be defined for each network.

Table 10.5: 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_{n1}	Voltage rating of the 1 th winding	kV
7	V_{n2}	Voltage rating of the 2 nd winding	kV
8	V_{n3}	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 10.6: 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_0	Voltage magnitude	p.u.
5	θ_0	Reference Angle	p.u.
† 6	Q_{\max}	Maximum reactive power	p.u.
† 7	Q_{\min}	Minimum reactive power	p.u.
† 8	V_{\max}	Maximum voltage	p.u.
† 9	V_{\min}	Minimum voltage	p.u.
† 10	P_{g0}	Active power guess	p.u.
† 11	γ	Loss participation coefficient	-
† 12	z	Reference bus	{0, 1}
† 13	u	Connection status	{0, 1}

2. **n**: total number of slack generators.
3. **bus**: indexes of buses to which slack generators are connected.
4. **vbus**: indexes of voltage buses of slack generators.
5. **refbus**: indexes of buses used as phase reference.
6. **store**: copy of the slack generator data. This field is used only in the command line version of PSAT (see Chapter 27).
7. **u**: connection status.

10.5 PV Generator

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

$$\begin{aligned} P &= P_g \\ V &= V_0 \end{aligned} \tag{10.6}$$

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

$$P = (1 + \gamma k_G) P_g \tag{10.7}$$

where k_G is the distributed slack bus variable and γ is the loss participation factor. Table 10.7 depicts PV generator data, which include reactive power and voltage limits needed for optimal power flow and continuation load flow analysis. Refer to

Chapters 6 and 5 for details. If the check of PV reactive limits is enforced (see GUI for General Settings, which is depicted in Fig. 4.1), 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:

$$\begin{aligned} P &= P_g \\ Q &= Q_{\max, \min} \end{aligned} \quad (10.8)$$

After solving the power flow, the PQ buses are switched again to PV buses, assuming $V_0 = V$ 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 structure **PV**, as follows:

1. **con**: PV generator data.
2. **n**: total number of PV generators.
3. **bus**: numbers of buses to which PV generators are connected.
4. **pq**: internal PQ bus data (used when generator reactive power limits are encountered):
 - (a) **con**: PQ load data.
 - (b) **n**: total number of PQ loads.
 - (c) **bus**: numbers of buses to which PQ loads are connected.
5. **u**: connection status.
6. **store**: copy of the PV generator data. This field is used only in the command line version of PSAT (see Chapter 27).

10.6 PQ Load

PQ loads are modeled as constant active and reactive powers:

$$\begin{aligned} P &= -P_L \\ Q &= -Q_L \end{aligned} \quad (10.9)$$

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

$$\begin{aligned} P &= -PV^2/V_{\lim}^2 \\ Q &= -QV^2/V_{\lim}^2 \end{aligned} \quad (10.10)$$

Table 10.7: 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	Active Power	p.u.
5	V_0	Voltage Magnitude	p.u.
† 6	Q_{\max}	Maximum reactive power	p.u.
† 7	Q_{\min}	Minimum reactive power	p.u.
† 8	V_{\max}	Maximum voltage	p.u.
† 9	V_{\min}	Minimum voltage	p.u.
† 10	γ	Loss participation coefficient	-
† 11	u	Connection status	{0,1}

where V_{\lim} is V_{\max} or V_{\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 10.8 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 8.2). 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:

$$\begin{aligned} P_0 &= -P_L/V_0^2 \\ Q_0 &= -Q_L/V_0^2 \end{aligned} \quad (10.11)$$

where V_0 is the voltage value obtained with the power flow solution.

PQ loads are defined in the structure PQ, as follows:

1. **con**: PQ load data.
2. **n**: total number of PQ loads.
3. **bus**: numbers of buses to which PQ loads are connected.
4. **gen**: 1 if it is a PQ generator, 0 otherwise.
5. **P0**: initial active power (used with non-conventional loads of Chapter 14).
6. **Q0**: initial reactive power (used with non-conventional loads of Chapter 14).
7. **u**: connection status.
8. **store**: copy of the PQ load data. This field is used only in the command line version of PSAT (see Chapter 27).

Other static and dynamic load models are discussed in Chapter 14.

Table 10.8: 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	Active Power	p.u.
5	Q_L	Reactive Power	p.u.
† 6	V_{\max}	Maximum voltage	p.u.
† 7	V_{\min}	Minimum voltage	p.u.
† 8	z	Allow conversion to impedance	{0, 1}
† 9	u	Connection status	{0, 1}

10.7 PQ Generator

PQ generators are modeled as constant active and reactive powers:

$$\begin{aligned} P &= P_g \\ Q &= Q_g \end{aligned} \quad (10.12)$$

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

$$\begin{aligned} P &= PV^2/V_{\lim}^2 \\ Q &= QV^2/V_{\lim}^2 \end{aligned} \quad (10.13)$$

where V_{\lim} is V_{\max} or V_{\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 10.9 depicts the PQ generator data format. If $z = 0$, voltage limit control is disabled.

Internally, PSAT translates PQ generators into PQ loads with:

$$\begin{aligned} P_L &= -P_g \\ Q_L &= -Q_g \end{aligned} \quad (10.14)$$

The field **gen** of the PQ structure says if the load was converted from a PQ generator (see Section 10.6). Observe that PQ generators are NOT converted into constant 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 structure **PQgen**, with the only field **con**.

10.8 Shunt

Shunt impedances are described by the following equations:

$$\begin{aligned} P &= -gV^2 \\ Q &= -bV^2 \end{aligned} \quad (10.15)$$

Table 10.9: 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	Active Power	p.u.
5	Q_g	Reactive Power	p.u.
† 6	V_{\max}	Maximum voltage	p.u.
† 7	V_{\min}	Minimum voltage	p.u.
† 8	z	Allow conversion to impedance	{0, 1}
† 9	u	Connection status	{0, 1}

Table 10.10: 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}

and are included in the network admittance matrix Y . The susceptance b is negative for inductive charges, positive for capacitive ones. Shunts are defined in the structure `Shunt`, as follows:

1. `con`: shunt impedance data.
2. `bus`: numbers of buses to which shunt are connected.
3. `g`: column vector of the conductances at each bus of the network.
4. `b`: column vector of the susceptances at each bus of the network.
5. `u`: connection status.

10.9 Area & Regions

PSAT allows defining areas and regions. These are currently used only for grouping variables to be plotted (see Section 8.3 for more details). Each area (region) is defined by a unique number, which has to correspond to one of the numbers defined in the column 5 (6) of the `Bus.con` data. Areas correspond to *loss zone data*,

Table 10.11: 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	%

while regions correspond to the *interchange area data* in the IEEE common data format [126].

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 class **Areas** and **Regions**, respectively, 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 11

CPF and OPF Data

This section describes the components needed for the OPF routines. The basic components are the slack generator, the PV generator and the PQ load. As defined in Tables 10.6, 10.7 and 10.8, the user can define the reactive power and voltage limits for the generation, and the voltage limits for the loads. Furthermore, in the definition of transmission lines and transformers, it is possible to set a limit for a maximum flow (current, active power or apparent power). Then, in the OPF window, the flow type can be specified and selected in a popup menu.

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

$$V_{\min_i} \leq V_i \leq V_{\max_i} \quad i = 1, \dots, N \quad (11.1)$$

whereas the limits for the generation are the following:

$$Q_{\min_i} \leq Q_{g_i} \leq Q_{\max_i} \quad i = 1, \dots, N_g \quad (11.2)$$

where N_g is the total number of generators given by the sum of the slack and PV generators. Finally the flows constraints are:

$$\Phi_i \leq \Phi_{\max_i} \quad i = 1, \dots, N_L \quad (11.3)$$

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. The 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 structures can be defined:

1. **Supply**: power bids, generator power directions and limits.
2. **Rsrv**: power reserve data.
3. **Rmpg**: power ramping data.

whereas for the load side the following two structures are available:

1. **Demand**: power bids, load power directions and limits.
2. **Rmp1**: power ramping data.

Each structure is composed of at least the following fields:

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

11.1 Generator Supply

The **Supply** structure defines the basic data for generations bids and costs, as depicted in Table 11.1. This structure is always required for running the OPF. The user has to define the range of the power bid and the cost parameter that can be both for active and reactive power generation, as defined by the following equations:

$$\begin{aligned} C(P_S) &= \sum_{i=1, N_S} C_{P_{0i}} + C_{P_{1i}} P_{S_i} + C_{P_{2i}} P_{S_i}^2 \\ C(Q_g) &= \sum_{i=1, N_S} C_{Q_{0i}} + C_{Q_{1i}} Q_{g_i} + C_{Q_{2i}} Q_{g_i}^2 \end{aligned} \quad (11.4)$$

Then the power supply inequalities are:

$$P_{S_{\min i}} \leq P_{S_i} \leq P_{S_{\max i}} \quad i = 1, \dots, N_S \quad (11.5)$$

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 variable u , i.e. the status of the generators.¹

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:

$$C_{TB} = k_{TB} \frac{P_S^2}{P_{S_{\max}}^2} \quad (11.6)$$

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

¹The unit commitment could be added in future versions.

Table 11.1: Power Supply Data Format (`Supply.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
† 3	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	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	Commitment variable	boolean
14	k_{TB}	Tie breaking cost	\$/MWh
15	γ	Loss participation factor	-
16	Q_g^{\max}	Maximum reactive power Q_g^{\max}	p.u.
17	Q_g^{\min}	Minimum reactive power Q_g^{\min}	p.u.
18	C^{ups}	Congestion up cost	\$/h
19	C^{dws}	Congestion down cost	\$/h
20	u	Connection status	{0, 1}

† This field is used only for the CPF analysis.

‡ This field is an output of the OPF routines and can be left zero.

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^{ups} and C^{dws} 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 5) 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.

11.2 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

Table 11.2: 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	$\{0, 1\}$

power reserve has an associated cost:

$$C(P_R) = \sum_{i=1, N_R} C_{R_i} P_{R_i} \quad (11.7)$$

and limits as for the power supplies:

$$P_{R_{\min i}} \leq P_{R_i} \leq P_{R_{\max i}} \quad i = 1, \dots, N_R \quad (11.8)$$

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:

$$\begin{aligned} P_{S_i} + P_{R_i} &\leq P_{S_{\max i}} \quad i = 1, \dots, N_R \\ \sum_{i=1, N_R} P_{R_i} &\leq \sum_{i=1, N_D} P_{D_i} \end{aligned} \quad (11.9)$$

The structure `Rsrv` defines these data, as reported in Table 11.2. The power reserve vector P_R as obtained by OPF routines are stored in the field `Pr`.

11.3 Generator Power Ramping

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 ramping constraints. The structure `Rmpg` defines the generator ramping data, as reported in Table 11.3.²

The parameters used in the optimization routine are the up and down ramp rates, i.e. R_{up} and R_{down} . 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:

$$\begin{aligned} P_{S_i}^t - P_{S_i}^{t-1} &\leq P_{S_{\max i}} R_{\text{up}i} \Delta t \quad i = 1, \dots, N_S \\ -P_{S_i}^t + P_{S_i}^{t-1} &\leq P_{S_{\max i}} R_{\text{down}i} \Delta t \end{aligned} \quad (11.10)$$

²These constraints have not been implemented yet but will be included soon in future versions.

Table 11.3: Generator Power Ramping 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_{down}	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:

$$P_{R_i} \leq R_{R_{max\ i}} \Delta t \quad i = 1, \dots, N_R \quad (11.11)$$

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:

$$P_{R_i} \leq P_{S_i} \frac{R_{R_{max\ i}} \Delta t}{P_{RLP_i}} \quad i = 1, \dots, N_R \quad (11.12)$$

Thus, the power reserve P_R will be the minimum between (11.11) and (11.12).

11.4 Load Demand

The **Demand** structure defines the basic data for load demand bids and costs, as presented in Table 11.4. 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 (11.4) for the power supplies, are:

$$\begin{aligned}
 C_1(P_D) &= \sum_{i=1, N_D} C_{P_{0i}} + C_{P_{1i}} P_{D_i} + C_{P_{2i}} P_{D_i}^2 \\
 C_2(P_D) &= \sum_{i=1, N_D} C_{Q_{0i}} + C_{Q_{1i}} P_{D_i} \tan(\phi_i) + C_{Q_{2i}} P_{D_i}^2 \tan(\phi_i)^2
 \end{aligned} \quad (11.13)$$

where

$$\tan \phi_i = \frac{Q_{D_{0i}}}{P_{D_{0i}}} \quad (11.14)$$

As for the constraints, one has:

$$P_{R_{min\ i}} \leq P_{R_i} \leq P_{R_{max\ i}} \quad i = 1, \dots, N_R \quad (11.15)$$

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 possible to set an unit commitment variable, i.e. the status of the loads.³

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:

$$C_{TB} = k_{TB} \frac{P_D^2}{P_{D_{\max}}} \quad (11.16)$$

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^{up_D} and C^{dw_D} 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 5) 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.

11.5 Demand Profile

The structure `Ypdp` defines the demand profile for multiperiod 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

³The unit commitment could be added in future versions.

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

Table 11.4: Power Demand Data Format (Demand.con)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
† 3	P_{D_0}	Active power direction	p.u.
† 4	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	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	Commitment variable	boolean
15	k_{TB}	Tie breaking cost	\$/MWh
16	C^{up}_D	Congestion up cost	\$/h
17	C^{dw}_D	Congestion down cost	\$/h
18	u	Connection status	{0, 1}

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

‡ This field is an output of the OPF routines and can be left blank.

used) and demand bid limits $P_{D_{\max}}$ and $P_{D_{\min}}$, so that:

$$P_{L0}(t) = \frac{\xi^t}{100} P_{L0} \quad \forall t \in \mathcal{T} \quad (11.17)$$

$$P_{D_{\max}}(t) = \frac{\xi^t}{100} P_{D_{\max}} \quad \forall t \in \mathcal{T} \quad (11.18)$$

$$P_{D_{\min}}(t) = \frac{\xi^t}{100} P_{D_{\min}} \quad \forall t \in \mathcal{T} \quad (11.19)$$

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:

$$\xi^t = \frac{k_{\alpha}^t(\alpha)}{100} \cdot \frac{k_{\beta}(\beta)}{100} \cdot \frac{k_{\gamma}(\gamma)}{100} 100 \quad (11.20)$$

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.

11.6 Load Ramping

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 ramping data, as reported in Table 11.6.⁵

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

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

Table 11.5: 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	$\{1, \dots, 52\}$

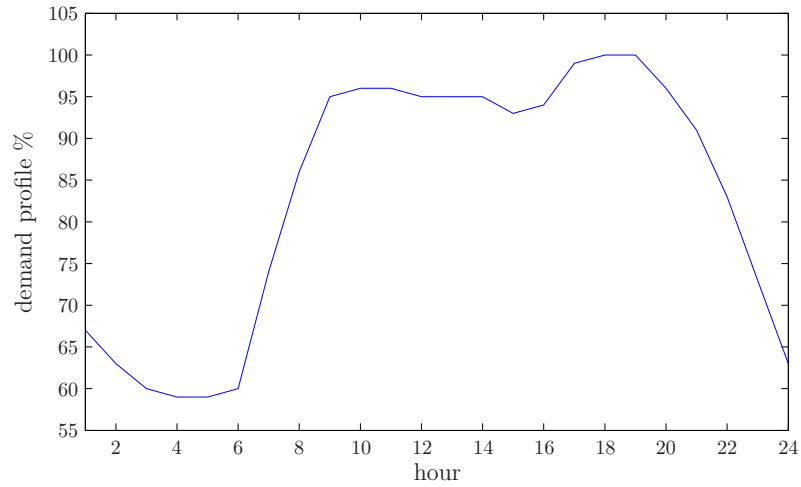


Figure 11.1: Example of daily demand profile.

Table 11.6: Load Ramping Data Format (`Rmp1.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_{down}	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	$\{0, 1\}$

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

$$\begin{aligned}
 P_{D_i}^t - P_{D_i}^{t-1} &\leq P_{D_{\max i}} R_{\text{up}i} \Delta t & i = 1, \dots, N_D \\
 -P_{D_i}^t + P_{D_i}^{t-1} &\leq P_{D_{\max i}} R_{\text{down}i} \Delta t
 \end{aligned} \tag{11.21}$$

These equations are conceptually similar to (11.10) for the generation ramp rate, and uses the same time interval Δt defined in the OPF window.

Chapter 12

Faults & Breakers

This chapter describes symmetrical three phase faults and breakers.

12.1 Fault

Table 12.1 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.

Three phase faults are defined in the **Fault** structure, as follows:

1. **con**: Fault data.
2. **n**: total number of faults.
3. **bus**: vector of bus numbers 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.

12.2 Breaker

Table 12.2 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.

Table 12.1: 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.

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 Only one breaker is needed for each line and its position (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.

Breakers are defined in the class **Breaker**, 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. **status**: boolean vector indicating the status of the breaker.

Breakers works properly **only** if connected to transmission lines. Connecting breakers to other components (e.g. synchronous machines) is not allowed and will lead to errors or unpredictable results. Refer to Chapter 22 for more details.

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

Table 12.2: 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

Chapter 13

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).

13.1 Bus Frequency Measurement

The bus frequency measurement is obtained by means of a high-pass and a low-pass filter, as depicted in Fig. 13.1. Differential equations are as follows:

$$\begin{aligned}\dot{x} &= \frac{1}{T_f} \left(\frac{1}{2\pi f_0} \frac{1}{T_f} (\theta - \theta_0) - x \right) \\ \dot{\omega} &= (\Delta\omega + 1 - \omega)/T_\omega\end{aligned}\tag{13.1}$$

where $\Delta\omega$ is:

$$\Delta\omega = -x + \frac{1}{2\pi f_0} \frac{1}{T_f} (\theta - \theta_0)\tag{13.2}$$

Bus frequency measurement data are stored in the structure **Busfreq**, with the following fields:

1. **con**: Bus frequency measurement data.

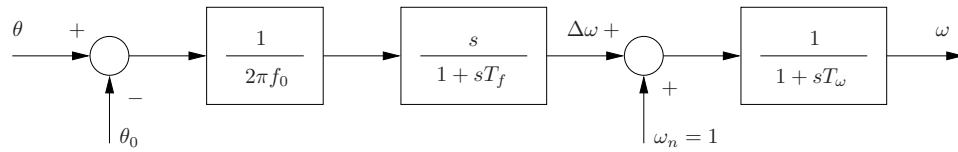


Figure 13.1: Bus frequency measurement filter.

Table 13.1: 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	$\{0, 1\}$

2. **n**: total number of components.
3. **dat**: Bus frequency measurement parameters.
4. **x**: indexes of state variables x .
5. **w**: indexes of state variables ω .
6. **u**: connection status.

Table 13.1 depicts the data format for the bus frequency measurement components.

13.2 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 [125].

Let define a sinusoidal quantity:

$$x(t) = X_M \cos(\omega t + \phi) \quad (13.3)$$

its phasor representation is:

$$X = \frac{X_M}{\sqrt{2}} e^{j\phi} \quad (13.4)$$

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. 13.2). 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:

$$X = \frac{1}{\sqrt{2}} \frac{2}{N} (X_c - jX_s) \quad (13.5)$$

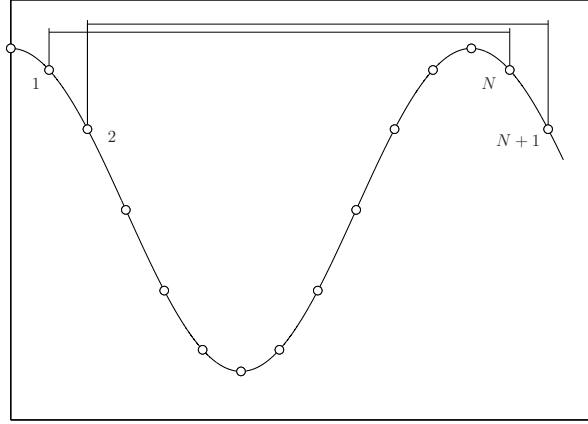


Figure 13.2: Phasors from sample data.

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

$$X_c = \sum_{k=1}^N x_k \cos k\theta \quad (13.6)$$

$$X_s = \sum_{k=1}^N x_k \sin k\theta \quad (13.7)$$

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

$$\theta = \frac{2\pi}{N} = 2\pi f_0 \tau \quad (13.8)$$

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 (13.5) 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:

$$X^{r+1} = X^r + \frac{1}{\sqrt{2}} \frac{2}{N} (x_{N+r} - x_r) e^{-jr\theta} \quad (13.9)$$

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

Table 13.2: 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	$\{0, 1\}$

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:

$$X^r(f_0 + \Delta f) = X e^{-j(N-1)\pi\Delta f\Delta t} \frac{\sin(N\Delta f\Delta t)}{N\sin(\Delta f\Delta t)} e^{j2\pi r\Delta f\Delta t} \quad (13.10)$$

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

$$\frac{d\psi}{dt} = 2\pi\Delta f \quad (13.11)$$

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:

$$\dot{v}_m = (V - v_m)/T_m \quad (13.12)$$

$$\dot{\theta}_m = (\theta - \theta_m)/T_\theta \quad (13.13)$$

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

PMU data are stored in the structure `Pmu`,¹ with the following fields:

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. `u`: connection status.

Table 13.2 depicts the data format for the PMU components.

¹Do not confuse the structure `Pmu` for PMU devices models with the structure `PMU` which is used in the PMU placement algorithms illustrated in Chapter 9.

Chapter 14

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 the following models, active and reactive load powers P and Q are positive if absorbed from the network.

14.1 Voltage Dependent Load

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

$$\begin{aligned} P &= P_0(V/V_0)^{\alpha_P} \\ Q &= Q_0(V/V_0)^{\alpha_Q} \end{aligned} \tag{14.1}$$

where V_0 is the initial voltage at the load bus as obtained by the power flow solution. Other parameters are defined in Table 14.1, which depicts the VDL data format.

VDLs can also be included directly in the power flow analysis. In this case, the initial voltage is not known V_0 , thus the following equations will be used:

$$\begin{aligned} P &= P_0 V^{\alpha_P} \\ Q &= Q_0 V^{\alpha_Q} \end{aligned} \tag{14.2}$$

The units of P_0 and Q_0 depends on the parameter z . If $z = 1$, the VDL is initialized after the power flow analysis, and P_0 and Q_0 are in percentage of the PQ load power connected at the VDL bus. Observe that if $z = 1$, it is mandatory to connect a PQ load at the VDL bus. If $u = 0$, the VDL is included in the power flow analysis, and P_0 and Q_0 are in p.u. In this case it is not necessary to connect a PQ load at the VDL bus.

Table 14.1: 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	P_0	Active power rating	% (p.u.)
5	Q_0	Reactive power rating	% (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 (14.1) are a simplification of the nonlinear general exponential voltage frequency dependent load described in Section 14.3.

Voltage dependent loads are defined in the structure **Mn**, as follows:

1. **con**: voltage dependent load data.
2. **n**: total number of voltage dependent loads.
3. **bus**: numbers of buses to which voltage dependent loads are connected.
4. **init**: status for power flow computations.
5. **u**: connection status.
6. **store**: copy of the voltage dependent load data. This field is used only in the command line version of PSAT (see Chapter 27).

14.2 ZIP Load

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

$$\begin{aligned} P &= g(V/V_0)^2 + I_P(V/V_0) + P_n \\ Q &= b(V/V_0)^2 + I_Q(V/V_0) + Q_n \end{aligned} \quad (14.3)$$

where V_0 is the initial voltage at the load bus as obtained by the power flow solution. Other parameters are defined in Table 14.2, which depicts the ZIP load data format.

ZIP loads can also be included directly in the power flow analysis. In this case, the initial voltage is not known V_0 , thus the following equations will be used:

$$\begin{aligned} P &= gV^2 + I_P V + P_n \\ Q &= bV^2 + I_Q V + Q_n \end{aligned} \quad (14.4)$$

Table 14.2: 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	g	Conductance	% (p.u.)
6	I_P	Active current	% (p.u.)
7	P_n	Active power	% (p.u.)
8	b	Susceptance	% (p.u.)
9	I_Q	Reactive current	% (p.u.)
10	Q_n	Reactive power	% (p.u.)
11	z	Initialize after power flow	{1,0}
12	u	Connection status	{1,0}

The units of P_0 and Q_0 depends on the status parameter z . If $z = 1$, the ZIP load is initialized after the power flow analysis, and P_0 and Q_0 are in percentage of the PQ load power connected at the ZIP load bus. Observe that if $z = 1$, it is mandatory to connect a PQ load at the ZIP load bus. If $z = 0$, the ZIP load is included in the power flow analysis, and P_0 and Q_0 are in p.u. In this case it is not necessary to connect a PQ load at the ZIP load bus.

ZIP loads are defined in the structure P1, as follows:

1. **con**: ZIP load data.
2. **n**: total number of ZIP loads.
3. **bus**: bus numbers to which ZIP loads are connected.
4. **init**: status for power flow computations.
5. **u**: connection status.

14.3 Frequency Dependent Load

A generalized exponential voltage frequency dependent load is modeled as follows [57]:

$$\begin{aligned}
 P &= \frac{k_P}{100} \left(\frac{V}{V_0} \right)^{\alpha_P} (1 + \Delta\omega)^{\beta_P} \\
 Q &= \frac{k_Q}{100} \left(\frac{V}{V_0} \right)^{\alpha_Q} (1 + \Delta\omega)^{\beta_Q}
 \end{aligned} \tag{14.5}$$

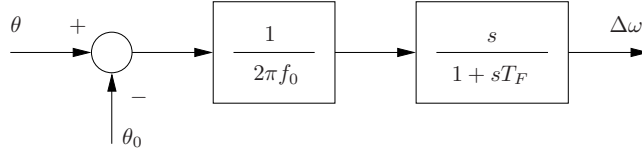


Figure 14.1: Measure of frequency deviation.

where $\Delta\omega$ represents the frequency deviation at the load bus, determined by filtering and differentiating the bus voltage phase angle θ as follows (see Fig. 14.1):

$$\begin{aligned}\dot{x} &= -\frac{1}{T_F} \left(\frac{1}{2\pi f_0} \frac{1}{T_F} (\theta - \theta_0) + x \right) \\ \Delta\omega &= x + \frac{1}{2\pi f_0} \frac{1}{T_F} (\theta - \theta_0)\end{aligned}\tag{14.6}$$

and V_0 and θ_0 are the voltage magnitude and phase angle determined in the power flow solution. This component is initialized after power flow computations. 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 14.3 reports the data format for the component whereas Table 14.4 depicts some typical coefficients for characteristic loads [13].

Frequency dependent loads are defined in the structure **F1**, as follows:

1. **con**: frequency dependent load data.
2. **n**: total number of polynomial power loads.
3. **bus**: bus numbers to which frequency dependent loads are connected.
4. **a0**: initial bus voltage phase angles.
5. **Dw**: bus frequency deviations $\Delta\omega$.
6. **x**: indexes of filter state variables x .
7. **u**: connection status.

14.4 Exponential Recovery Load

An exponential recovery load is included in PSAT based on what was proposed in [55,63]. Equations are as follows:

$$\begin{aligned}\dot{x}_P &= -x_P/T_P + P_s - P_t \\ P &= x_P/T_P + P_t\end{aligned}\tag{14.7}$$

Table 14.3: 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}

Table 14.4: 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

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

$$\begin{aligned} P_s &= P_0(V/V_0)^{\alpha_s} \\ P_t &= P_0(V/V_0)^{\alpha_t} \end{aligned} \quad (14.8)$$

Similar equations hold for the reactive power:

$$\begin{aligned} \dot{x}_Q &= -x_Q/T_Q + Q_s - Q_t \\ Q &= x_Q/T_Q + Q_t \end{aligned} \quad (14.9)$$

and:

$$\begin{aligned} Q_s &= Q_0(V/V_0)^{\beta_s} \\ Q_t &= Q_0(V/V_0)^{\beta_t} \end{aligned} \quad (14.10)$$

The power flow solution and the PQ load data are used for determining the values of P_0 , Q_0 and V_0 . A PQ load has to be connected to the exponential recovery load bus. The data format is depicted in Table 14.5.

Exponential recovery loads are defined in the structure **Exload**, as follows:

Table 14.5: Exponential Recovery Load Data Format (`Exload.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	V_n	Active power voltage coefficient	kV
4	f_n	Active power frequency coefficient	Hz
5	T_P	Real power time constant	s
6	T_Q	Reactive power time constant	s
7	α_s	Static real power exponent	-
8	α_t	Dynamic real power exponent	-
9	β_s	Static reactive power exponent	-
10	β_t	Dynamic reactive power exponent	-
11	u	Connection status	{1,0}

1. `con`: Exponential recovery load data.
2. `bus`: number of buses to which the 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. `u`: connection status.

14.5 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. 14.2 which represents the following equations:

$$\begin{aligned}
 \dot{T} &= (T_a - T + K_1 P)/T_1 \\
 \dot{x} &= K_i(T_{\text{ref}} - T)/T_i \\
 G &= K_p(T_{\text{ref}} - T) + x \\
 P &= GV^2
 \end{aligned} \tag{14.11}$$

where the state variable x undergoes an anti-windup limiter and the algebraic variable G undergoes a windup limiter.

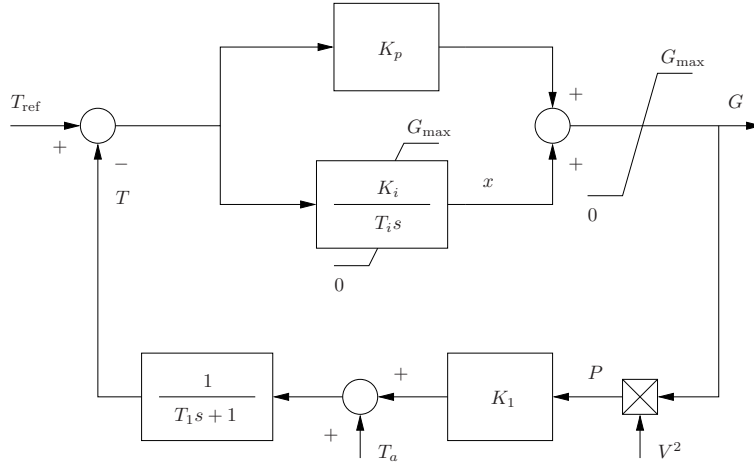


Figure 14.2: Thermostatically controlled load.

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:

$$\begin{aligned} K_1 &= \frac{T_{\text{ref}} - T_0}{P_0} \\ G_{\max} &= K_L G_0 \end{aligned} \quad (14.12)$$

where $G_0 = P_0/V_0^2$ and K_L ($K_L > 1$) is the ceiling conductance output ratio. If $K_L = 0$, the value of G_{\max} is defined by the user in the component data matrix (see Table 14.6). The structure **Thload** is organized as follows:

1. **con**: data of the **Thload** components.
2. **bus**: bus number to which the components are connected.
3. **n**: total number of components.
4. **T**: indexes of the state variable T .
5. **G**: indexes of the state variable G .
6. **u**: connection status.

Table 14.6 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. G_{\max} and K_1 are computed and stored in the data matrix during the initialization step.

Table 14.6: Thermostatically Controlled Load Data Format (**Thload.con**)

Column	Variable	Description	Unit
1	-	Bus number	int
2	-	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	T_a	Ambient temperature	°F or °C
8	T_{ref}	Reference temperature	°F or °C
9	G_{max}	Maximum conductance	p.u./p.u.
10	K_1	Active power gain	(°F or °C)/p.u.
11	K_L	Ceiling conductance output	p.u./p.u.
12	u	Connection status	{0, 1}

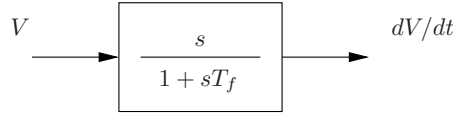


Figure 14.3: Jimma's load.

14.6 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 [61, 122]. 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 derivative is defined using a service state variable x and a high-pass filter (see Fig. 14.3). The differential equation is as follows:

$$\begin{aligned} \dot{x} &= (-V/T_f - x)/T_f \\ \frac{dV}{dt} &= x + V/T_f \end{aligned} \quad (14.13)$$

The power injections are defined as follows:

$$P = P_{Lz} \left(\frac{V_L}{V_{L0}} \right)^2 + P_{Li} \left(\frac{V_L}{V_{L0}} \right) + P_{LP} \quad (14.14)$$

$$Q = Q_{Lz} \left(\frac{V_L}{V_{L0}} \right)^2 + Q_{Li} \left(\frac{V_L}{V_{L0}} \right) + Q_{LP} + K_V \frac{dV_L}{dt} \quad (14.15)$$

The power flow solution provides the initial voltage V_0 that is needed for computing the power injections. The structure **Jimma** is organized as follows:

Table 14.7: 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	P_{LZ}	Percentage of active power $\propto V^2$	%
7	P_{LI}	Percentage of active power $\propto V$	%
8	P_{LP}	Percentage of constant active power	%
9	Q_{LZ}	Percentage of reactive power $\propto V^2$	%
10	Q_{LI}	Percentage of reactive power $\propto V$	%
11	Q_{LP}	Percentage of constant reactive power	%
12	K_V	Coefficient of the voltage time derivative	1/s
13	u	Connection status	$\{0, 1\}$

1. `con`: data of the `Jimma` components.
2. `dat`: vector of initial voltages V_0 .
3. `bus`: bus number to which the components are connected.
4. `n`: total number of components.
5. `x`: indexes of the state variable x .
6. `u`: connection status.

Table 14.7 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 $Q = 0$.

14.7 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. 14.3 and 14.1). The differential equations are as follows:

$$\begin{aligned} \dot{x} &= (-V/T_{fv} - x)/T_{fv} \\ \Rightarrow \frac{dV}{dt} &= x + V/T_{fv} \end{aligned} \quad (14.16)$$

$$\begin{aligned} \dot{y} &= -\frac{1}{T_{ft}} \left(\frac{1}{2\pi f_0} \frac{1}{T_{ft}} (\theta - \theta_0) + y \right) \\ \Rightarrow \Delta\omega &= y + \frac{1}{2\pi f_0} \frac{1}{T_{ft}} (\theta - \theta_0) \end{aligned} \quad (14.17)$$

The bus power injections P and Q are defined as follows:

$$P = K_{pf}\Delta\omega + K_{pv} \left[\frac{V^\alpha}{V_0} + T_{pv} \frac{dV}{dt} \right] \quad (14.18)$$

$$Q = K_{qf}\Delta\omega + K_{qv} \left[\frac{V^\beta}{V_0} + T_{qv} \frac{dV}{dt} \right] \quad (14.19)$$

The power flow solution provides the initial voltage V_0 that is needed for computing the power injections. The structure **Mixload** is organized as follows:

1. **con**: data of the **Mixload** components.
2. **dat**: vector of initial voltages V_0 and θ_0 .
3. **bus**: bus number to which the components 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. **u**: connection status.

Table 14.8 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 mixed load a warning message is displayed and it is assumed that $P = 0$ and $Q = 0$.

14.8 Note on the Use of Non-conventional Loads

Observe that, in general, all non-conventional loads need a PQ load connected at the same bus. Only voltage dependent and ZIP loads can be used alone if the “Initialize after power flow” parameter is set to zero.

The powers used for initializing non-conventional loads but the exponential recovery 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. Observe that PSAT does not check

Table 14.8: 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_{pv}	Frequency coefficient for the active power	p.u.
6	K_{pv}	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_{pv}	Frequency coefficient for the reactive power	p.u.
10	K_{pv}	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	$\{0, 1\}$

if the total sum of non-conventional load percentages is greater than 100%. When this happens the resulting PQ load will show negative powers.

Finally, observe that exponential recovery loads are processed last, clearing the PQ load powers. Thus, be aware that when exponential recovery loads and other non-conventional loads are used together, the actual powers used by exponential recovery loads are the powers remaining after the initialization of the other loads (*not* the initial PQ load powers).

Refer to Section 22.3.8 for details on the usage of non-conventional load within SIMULINK models.

Chapter 15

Machines

This chapter describes the synchronous machine and the induction motor models. These components are described by the general equations:

$$\begin{aligned}\dot{x} &= f(x, y, u) \\ P &= g_P(x, y) \\ Q &= g_Q(x, y)\end{aligned}\tag{15.1}$$

where x are the state variables, y the algebraic variables (i.e. V and θ) and u input variables.

With regard to the induction motor models, u (i.e. the mechanical torque T_m) is set by the user, and equations (15.1) are included in the power flow analysis.

The 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 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 and input variables, the latter being the field voltage v_f and the mechanical torque T_m . Thus, the following system is solved:

$$\begin{aligned}0 &= f(x, y_0, u) \\ P_0 &= g_P(x, y_0) \\ Q_0 &= g_Q(x, y_0)\end{aligned}\tag{15.2}$$

At the end of the initialization procedure, the PV and/or slack generators connected at the generator buses are removed.

Synchronous machines controls such as AVRs or Turbine Governors are not included in the machine models. Refer to Chapter 16 for a detailed description of generator control systems.

15.1 Synchronous Machine

The Park-Concordia model is used for synchronous machine equations, whose scheme is depicted in Fig. 15.1. Various simplification levels are applied, from the

classical swing equations to an eight order model with field saturation. Fig. 15.2 depicts the d and q -axis block diagrams of stator fluxes for the VI order model while Fig. 15.3 illustrates the field saturation characteristic of the synchronous machine. The link between the network phasors and the machine voltage is as follows:

$$\begin{aligned} v_d &= V \sin(\delta - \theta) \\ v_q &= V \cos(\delta - \theta) \end{aligned} \quad (15.3)$$

The expressions of d and q -axis currents depend on the model, and in general terms are defined as follows:

$$\begin{aligned} 0 &= g_1(x, i_d, i_q, V, \theta) \\ 0 &= g_2(x, i_d, i_q, V, \theta) \end{aligned} \quad (15.4)$$

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:

$$0 = v_d i_d + v_q i_q - P \quad (15.5)$$

$$0 = v_q i_d - v_d i_q - Q \quad (15.6)$$

$$0 = p_{m0} - p_m \quad (15.7)$$

$$0 = v_{f0} - v_f \quad (15.8)$$

where v_d and v_q are defined in (15.3), and p_{m0} and v_{f0} are the mechanical power and the field voltage algebraic variables, respectively. Observe that, 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.

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:

$$v_f^* = v_f + K_\omega(\omega - 1) - K_P(P(x, V, \theta) - P_0) \quad (15.9)$$

where P_0 is the initial electric power generated by the machine. Equation (15.9) implements a simple oscillation stabilizer and is implied where the notation v_f^* is used.

Table 15.1 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 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 15.2 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 needed

Table 15.1: 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	p.u.	all
7	r_a	Armature resistance	p.u.	all
8	x_d	d -axis synchronous reactance	p.u.	III, IV, V.1, V.2, V.3, VI, VIII
9	x'_d	d -axis transient reactance	p.u.	II, III, IV, V.1, V.2, V.3, VI, VIII
10	x''_d	d -axis subtransient reactance	p.u.	V.2, VI, VIII
11	T'_{d0}	d -axis open circuit transient time constant	s	III, IV, V.1, V.2, V.3, VI, VIII
12	T''_{d0}	d -axis open circuit subtransient time constant	s	V.2, VI, VIII
13	x_q	q -axis synchronous reactance	p.u.	III, IV, V.1, V.2, V.3, VI, VIII
14	x'_q	q -axis transient reactance	p.u.	IV, V.1, VI, VIII
15	x''_q	q -axis subtransient 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 subtransient 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	III, IV, V.1, V.2, VI
† 21	K_P	Active power feedback gain	gain	III, IV, V.1, V.2, VI
† 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	-	III, IV, V.1, V.2, VI, VIII
† 26	$S(1.2)$	Second saturation factor	-	III, IV, V.1, V.2, VI, VIII
† 27	n_{COI}	Center of inertia number	int	all
† 28	u	Connection status	{0,1}	all

† optional fields

[illegible]

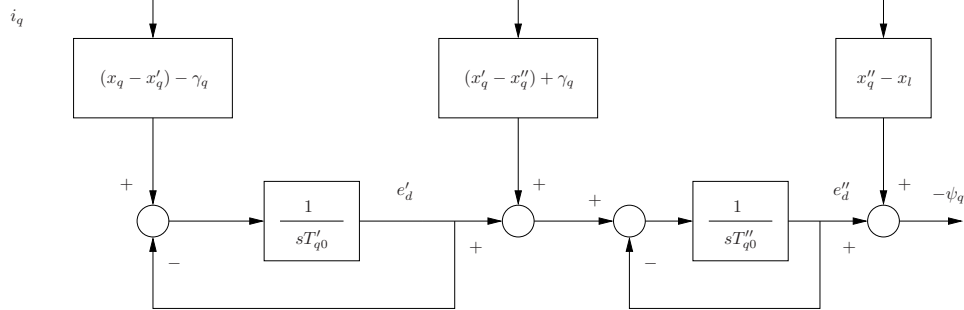


Figure 15.2: d and q -axis block diagrams of the stator fluxes for the most detailed synchronous machine model. Coefficients γ_d and γ_q are defined as follows:

$$\gamma_d = \frac{T''_{d0}}{T'_{d0}} \frac{x''_d}{x'_d} (x_d - x'_d)$$

$$\gamma_q = \frac{T''_{q0}}{T'_{q0}} \frac{x''_q}{x'_q} (x_q - x'_q)$$

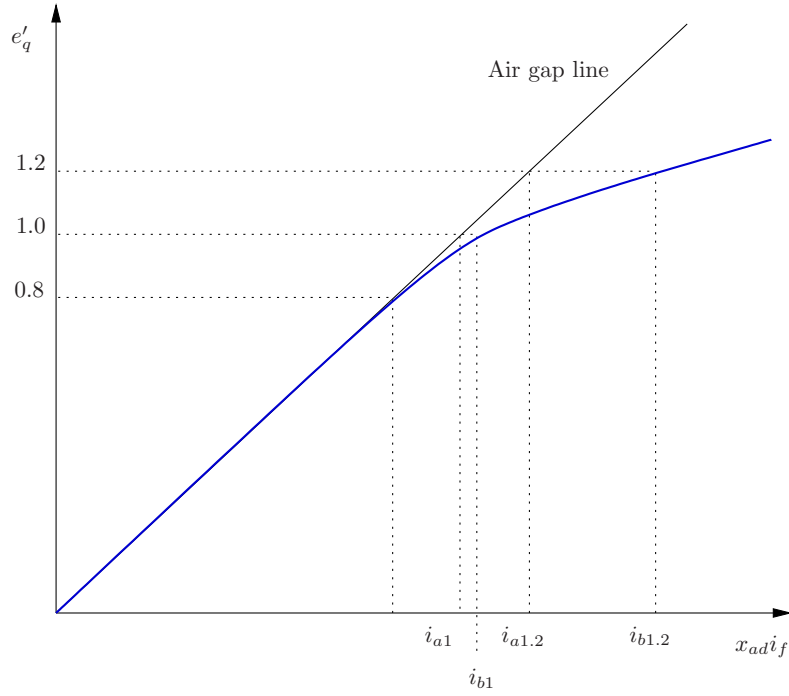


Figure 15.3: Field saturation characteristic of synchronous machines. Saturation factors are defined as follows:

$$S(1.0) = 1 - \frac{i_{a1}}{i_{b1}}$$

$$S(1.2) = 1 - \frac{i_{a1.2}}{i_{b1.2}}$$

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 $T_{AA} = 0$.

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 **Syn** structure, with the following fields:

1. **con**: Synchronous machine data.
2. **n**: total number of synchronous machines.
3. **bus**: numbers 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 subtransient voltage e''_q indexes.
11. **e2d**: d -axis subtransient 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_{m0} .
20. **vf0**: initial field voltage v_{f0} .
21. **u**: connection status.

15.1.1 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:

$$\begin{aligned}\dot{\delta} &= \Omega_b(\omega - 1) \\ \dot{\omega} &= (p_m - p_e - D(\omega - 1))/M\end{aligned}\quad (15.10)$$

where the electrical power p_e is defined as follows:

$$p_e = (v_q + r_a i_q) i_q + (v_d + r_a i_d) i_d \quad (15.11)$$

Finally, the following relationships between voltages and currents hold:

$$\begin{aligned}0 &= v_q + r_a i_q - e'_q + (x'_d - x_l) i_d \\ 0 &= v_d + r_a i_d - (x'_d - x_l) i_q\end{aligned}\quad (15.12)$$

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.

15.1.2 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:

$$\begin{aligned}\dot{\delta} &= \Omega_b(\omega - 1) \\ \dot{\omega} &= (p_m - p_e - D(\omega - 1))/M \\ \dot{e}'_q &= (-f_s(e'_q) - (x_d - x'_d) i_d + v_f^*)/T'_{d0}\end{aligned}\quad (15.13)$$

where the electrical power p_e is (15.11) and the voltage and current link is described by the equations:

$$\begin{aligned}0 &= v_q + r_a i_q - e'_q + (x'_d - x_l) i_d \\ 0 &= v_d + r_a i_d - (x_q - x_l) i_q\end{aligned}\quad (15.14)$$

This model is the simplest one to which an Automatic Voltage Regulator can be connected.

15.1.3 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 :

$$\begin{aligned}\dot{\delta} &= \Omega_b(\omega - 1) \\ \dot{\omega} &= (p_m - p_e - D(\omega - 1))/M \\ \dot{e}'_q &= (-f_s(e'_q) - (x_d - x'_d)i_d + v_f^*)/T'_{d0} \\ \dot{e}'_d &= (-e'_d + (x_q - x'_q)i_q)/T'_{q0}\end{aligned}\tag{15.15}$$

where the electrical power p_e is (15.11) and the voltage and current link is described by the equations:

$$\begin{aligned}0 &= v_q + r_a i_q - e'_q + (x'_d - x_l)i_d \\ 0 &= v_d + r_a i_d - e'_d - (x'_q - x_l)i_q\end{aligned}\tag{15.16}$$

A similar fourth order model can be formulated using the subtransient d -axis voltage e''_d instead of e'_d . The corresponding differential equation is:

$$\dot{e}''_d = (-e''_d + (x_q - x''_q)i_q)/T''_{q0}\tag{15.17}$$

15.1.4 Order V, Type 1

In this model, it is assumed:

$$x'_d \approx x''_d \approx x''_q\tag{15.18}$$

which leads to a single d -axis equation for the variable e'_q . The d -axis transient and subtransient 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:

$$\begin{aligned}\dot{\delta} &= \Omega_b(\omega - 1) \\ \dot{\omega} &= (p_m - p_e - D(\omega - 1))/M \\ \dot{e}'_q &= (-f_s(e'_q) - (x_d - x'_d)i_d + v_f^*)/T'_{d0} \\ \dot{e}'_d &= (-e'_d + (x_q - x'_q - \frac{T''_{q0}}{T'_{q0}} \frac{x'_d}{x'_q} (x_q - x'_q))i_q)/T'_{q0} \\ \dot{e}''_d &= (-e''_d + e'_d + (x'_q - x'_d + \frac{T''_{q0}}{T'_{q0}} \frac{x'_d}{x'_q} (x_q - x'_q))i_q)/T''_{q0}\end{aligned}\tag{15.19}$$

where the electrical power p_e is (15.11) and the voltage and current link is as follows:

$$\begin{aligned}0 &= v_q + r_a i_q - e'_q + (x'_d - x_l)i_d \\ 0 &= v_d + r_a i_d - e''_d - (x'_q - x_l)i_q\end{aligned}\tag{15.20}$$

15.1.5 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:

$$\begin{aligned}
 \dot{\delta} &= \Omega_b(\omega - 1) \\
 \dot{\omega} &= (p_m - p_e - D(\omega - 1))/M \\
 \dot{e}_q' &= (-f_s(e_q') - (x_d - x_d' - \frac{T_{d0}''}{T_{d0}'} \frac{x_d''}{x_d'}(x_d - x_d'))i_d + (1 - \frac{T_{AA}}{T_{d0}'}v_f^*)/T_{d0}' \\
 \dot{e}_q'' &= (-e_q'' + e_q' - (x_d' - x_d'' + \frac{T_{d0}''}{T_{d0}'} \frac{x_d''}{x_d'}(x_d - x_d'))i_d + \frac{T_{AA}}{T_{d0}'}v_f^*)/T_{d0}'' \\
 \dot{e}_d'' &= (-e_d'' + (x_q - x_q'')i_q)/T_{q0}''
 \end{aligned} \tag{15.21}$$

where the electrical power p_e is (15.11) and the voltage and current link is as follows:

$$\begin{aligned}
 0 &= v_q + r_a i_q - e_q'' + (x_d'' - x_l)i_d \\
 0 &= v_d + r_a i_d - e_d'' - (x_q'' - x_l)i_q
 \end{aligned} \tag{15.22}$$

15.1.6 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:

$$\begin{aligned}
 \dot{\delta} &= \Omega_b(\omega - 1) \\
 \dot{\omega} &= (p_m - p_e - D(\omega - 1))/M \\
 \dot{\psi}_f &= (v_f - e_q')/T_{d0}' \\
 \dot{\psi}_q &= \Omega_b(v_q + r_a i_q - \omega \psi_d) \\
 \dot{\psi}_d &= \Omega_b(v_d + r_a i_d + \omega \psi_q)
 \end{aligned} \tag{15.23}$$

where the electrical power p_e is:

$$p_e = \psi_d i_q - \psi_q i_d \tag{15.24}$$

To complete the model, three algebraic constraints are needed:

$$\begin{aligned}
 \psi_f &= e_q' - (x_d - x_d')i_d \\
 \psi_d &= e_d' - (x_d - x_l)i_d \\
 \psi_q &= -(x_q - x_l)i_q
 \end{aligned} \tag{15.25}$$

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 :

$$\dot{e}_q' = \frac{x_d - x_l}{x_d' - x_l} \left(\frac{1}{T_{d0}'} (v_f - e_q') - \frac{x_d - x_d'}{x_d - x_l} \dot{\psi}_d \right) \tag{15.26}$$

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.

15.1.7 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:

$$\begin{aligned}
 \dot{\delta} &= \Omega_b(\omega - 1) \\
 \dot{\omega} &= (p_m - p_e - D(\omega - 1))/M \\
 \dot{e}'_q &= (-f_s(e'_q) - (x_d - x'_d - \frac{T''_{d0}}{T'_{d0}} \frac{x''_d}{x'_d} (x_d - x'_d))i_d + (1 - \frac{T_{AA}}{T'_{d0}})v_f^*)/T'_{d0} \\
 \dot{e}'_d &= (-e'_d + (x_q - x'_q - \frac{T''_{q0}}{T'_{q0}} \frac{x''_q}{x'_q} (x_q - x'_q))i_q)/T'_{q0} \\
 \dot{e}''_q &= (-e''_q + e'_q - (x'_d - x''_d + \frac{T''_{d0}}{T'_{d0}} \frac{x''_d}{x'_d} (x_d - x'_d))i_d + \frac{T_{AA}}{T'_{d0}}v_f^*)/T''_{d0} \\
 \dot{e}''_d &= (-e''_d + e'_d + (x'_q - x''_q + \frac{T''_{q0}}{T'_{q0}} \frac{x''_q}{x'_q} (x_q - x'_q))i_q)/T''_{q0}
 \end{aligned} \tag{15.27}$$

where the electrical power p_e is (15.11) and the algebraic constraints are as follows:

$$\begin{aligned}
 0 &= v_q + r_a i_q - e''_q + (x''_d - x_l)i_d \\
 0 &= v_d + r_a i_d - e''_d - (x''_q - x_l)i_q
 \end{aligned} \tag{15.28}$$

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 $\omega\psi_q \approx \psi_q$.

15.1.8 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:

$$\begin{aligned}
 \dot{\delta} &= \Omega_b(\omega - 1) \\
 \dot{\omega} &= (p_m - p_e - D(\omega - 1))/M \\
 \dot{e}'_q &= (-f_s(e'_q) - (x_d - x'_d - \frac{T''_{d0}}{T'_{d0}} \frac{x''_d}{x'_d} (x_d - x'_d))i_d + (1 - \frac{T_{AA}}{T'_{d0}})v_f)/T'_{d0} \\
 \dot{e}'_d &= (-e'_d + (x_q - x'_q - \frac{T''_{q0}}{T'_{q0}} \frac{x''_q}{x'_q} (x_q - x'_q))i_q)/T'_{q0} \\
 \dot{e}''_q &= (-e''_q + e'_q - (x'_d - x''_d + \frac{T''_{d0}}{T'_{d0}} \frac{x''_d}{x'_d} (x_d - x'_d))i_d + \frac{T_{AA}}{T'_{d0}}v_f)/T''_{d0} \\
 \dot{e}''_d &= (-e''_d + e'_d + (x'_q - x''_q + \frac{T''_{q0}}{T'_{q0}} \frac{x''_q}{x'_q} (x_q - x'_q))i_q)/T''_{q0} \\
 \dot{\psi}_q &= \Omega_b(v_q + r_a i_q - \omega\psi_d) \\
 \dot{\psi}_d &= \Omega_b(v_d + r_a i_d + \omega\psi_q)
 \end{aligned} \tag{15.29}$$

where the electrical power p_e is (15.24). The following algebraic relationship complete the model:

$$\begin{aligned}\psi_d &= e_q'' - (x_d'' - x_l)i_d \\ \psi_q &= -e_d'' - (x_q'' - x_l)i_q\end{aligned}\tag{15.30}$$

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.

15.1.9 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:

$$\dot{\delta} = \Omega_b(\omega - 1)\tag{15.31}$$

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:

$$\delta_{\text{COI}} = \frac{\sum_i M_i \delta_i}{\sum_i M_i}\tag{15.32}$$

$$\omega_{\text{COI}} = \frac{\sum_i M_i \omega_i}{\sum_i M_i}\tag{15.33}$$

Then, (15.31) becomes:

$$\dot{\delta} = \Omega_b(\omega - \omega_{\text{COI}})\tag{15.34}$$

To enable the usage of the COI, one has to set

```
>> Setting.coi = 1;
```

or checking the box “Use Center of Inertia (COI)” in the general settings GUI.

In PSAT, the COI is defined by the structure `COI`. This structure is created by 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 15.1). Each number defines a COI. Use same COI numbers to group synchronous machines in the same COI.

15.2 Induction Motor

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:

$$T_m = a + b\omega + c\omega^2\tag{15.35}$$

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:

$$T_m = \alpha + \beta\sigma + \gamma\sigma^2 \quad (15.36)$$

where

$$\begin{aligned} \alpha &= a + b + c \\ \beta &= -b - 2c \\ \gamma &= c \end{aligned}$$

Table 15.3 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 **Mot** structure, which has the following fields:

1. **con**: induction motor data.
2. **n**: total number of induction motors.
3. **bus**: numbers 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 2st cage voltage e''_r indexes.
9. **e2m**: imaginary part of 2st cage voltage e''_m indexes.
10. **u**: connection status.

Following subsections describe the detailed models of the three induction motor models.

15.2.1 Order I

The electrical circuit used for the first order induction motor is depicted in Fig. 15.4. Only the mechanical state variable is considered, being the circuit in steady-state condition. The differential equation is as follows:

$$\dot{\sigma} = \frac{1}{2H_m} \left(T_m(\sigma) - \frac{r_{R1}V^2/\sigma}{(r_S + r_{R1}/\sigma)^2 + (x_S + x_{R1})^2} \right) \quad (15.37)$$

Table 15.3: Induction Motor Data Format (`Mot.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	boolean	all
7	r_S	Stator resistance	p.u.	III, V
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 $T_m(\omega)$	p.u.	all
16	b	2 nd coeff. of $T_m(\omega)$	p.u.	all
17	c	3 rd coeff. of $T_m(\omega)$	p.u.	all
18	t_{up}	Start up time	s	all
19	-	Allow working as brake	$\{0, 1\}$	all
20	u	Connection status	$\{0, 1\}$	all

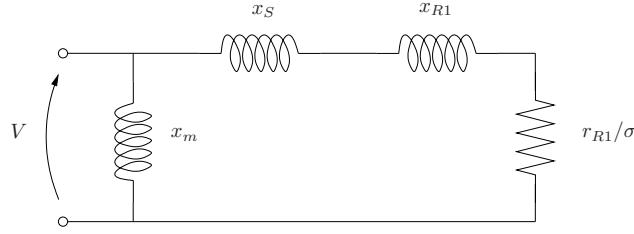


Figure 15.4: Order I induction motor: electrical circuit.

whereas the power injections are:

$$\begin{aligned} P &= -\frac{r_{R1}V^2/\sigma}{(r_S + r_{R1}/\sigma)^2 + (x_S + x_{R1})^2} \\ Q &= -\frac{V^2}{x_m} - \frac{(x_S + x_{R1})V^2}{(r_S + r_{R1}/\sigma)^2 + (x_S + x_{R1})^2} \end{aligned} \quad (15.38)$$

15.2.2 Order III (single cage)

The simplified electrical circuit used for the single-cage induction motor is depicted in Fig. 15.5. 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 stator machine voltages is as follows:

$$\begin{aligned} v_r &= -V \sin \theta \\ v_m &= V \cos \theta \end{aligned} \quad (15.39)$$

Using the notation of Fig. 15.5, the power absorptions are:

$$\begin{aligned} P &= -(v_r i_r + v_m i_m) \\ Q &= -(v_m i_r - v_r i_m) \end{aligned} \quad (15.40)$$

The differential equations in terms of the voltage behind the the stator resistance r_S are:

$$\begin{aligned} \dot{e}'_r &= \Omega_b \sigma e'_m - (e'_r + (x_0 - x')i_m)/T'_0 \\ \dot{e}'_m &= -\Omega_b \sigma e'_r - (e'_m - (x_0 - x')i_r)/T'_0 \end{aligned} \quad (15.41)$$

whereas the link between voltages, currents and state variables is as follows:

$$\begin{aligned} v_r - e'_r &= r_S i_r - x' i_m \\ v_m - e'_m &= r_S i_m + x' i_r \end{aligned} \quad (15.42)$$

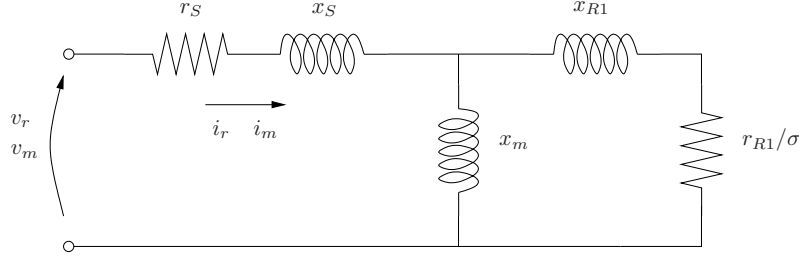


Figure 15.5: Order III induction motor: electrical circuit.

where x_0 , x' and T_0 can be obtained from the motor parameters:

$$\begin{aligned} x_0 &= x_S + x_m \\ x' &= x_S + \frac{x_{R1}x_m}{x_{R1} + x_m} \\ T'_0 &= \frac{x_{R1} + x_m}{\Omega_b r_{R1}} \end{aligned} \quad (15.43)$$

Finally, the mechanical equation is as follows:

$$\dot{\sigma} = (T_m(\sigma) - T_e)/(2H_m) \quad (15.44)$$

where the electrical torque is:

$$T_e = e'_r i_r + e'_m i_m \quad (15.45)$$

15.2.3 Order V (double cage)

The electrical circuit for the double-cage induction machine model is depicted in Fig. 15.6. In analogy with the single-cage model, machine real and imaginary axis are defined with respect to the network reference angle, and (15.39) and (15.40) apply. Two voltages behind the stator resistance r_S model the cage dynamics, as follows:

$$\begin{aligned} \dot{e}'_r &= \Omega_b \sigma e'_m - (e'_r + (x_0 - x')i_m)/T'_0 \\ \dot{e}'_m &= -\Omega_b \sigma e'_r - (e'_m - (x_0 - x')i_r)/T'_0 \\ \dot{e}''_r &= -\Omega_b \sigma (e'_m - e''_m) + \dot{e}'_r - (e'_r - e''_m - (x' - x'')i_m)/T''_0 \\ \dot{e}''_m &= \Omega_b \sigma (e'_r - e''_r) + \dot{e}'_m - (e'_m - e''_r + (x' - x'')i_r)/T''_0 \end{aligned} \quad (15.46)$$

and the links between voltages and currents are:

$$\begin{aligned} v_r - e''_r &= r_S i_r - x'' i_m \\ v_m - e''_m &= r_S i_m + x'' i_r \end{aligned} \quad (15.47)$$

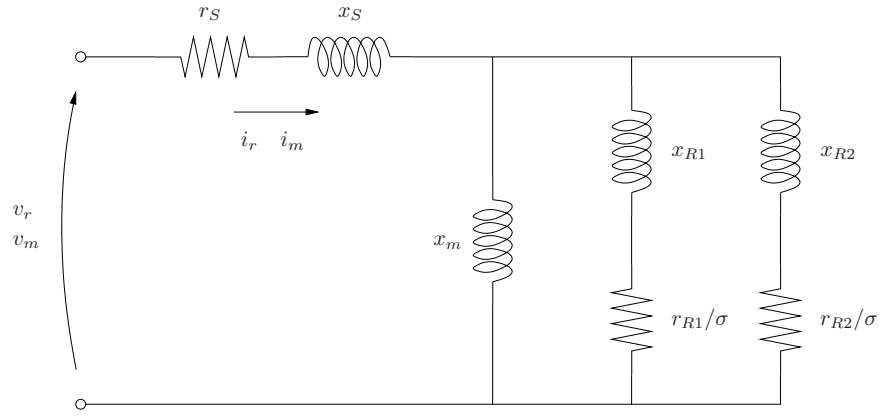


Figure 15.6: Order V induction motor: electrical circuit.

where the parameters are determined from the circuit resistances and reactances and are given by equations (15.43) and:

$$x'' = x_S + \frac{x_{R1}x_{R2}x_m}{x_{R1}x_{R2} + x_{R1}x_m + x_{R2}x_m} \quad (15.48)$$

$$T_0'' = \frac{x_{R2} + x_{R1}x_m/(x_{R1} + x_m)}{\Omega_b r_{R2}}$$

The differential equation for the slip is the (15.44), while the electrical torque is defined as follows:

$$T_e = e_r'' i_r + e_m'' i_m \quad (15.49)$$

Chapter 16

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 AVRs and SVCs, and a Power Oscillation Damper. Control models are described by means of a set of differential equations, as follows:

$$\begin{aligned}\dot{x} &= f(x, y, z_{\text{in}}) \\ z_{\text{out}} &= z_{\text{out}}(x, y, z_{\text{in}})\end{aligned}\tag{16.1}$$

where x are the state variable of the component, y the algebraic variables (e.g. bus voltages in case of AVRs), z_{in} are the input variables (e.g. the rotor speed in case of TGs), and z_{out} are the output variables (e.g. the synchronous machine field voltage and mechanical torque).

16.1 Turbine Governor

Turbine Governors (TGs) define the primary frequency regulation of synchronous machines. TG data are stored in the structure **Tg**, 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.
5. **dat2**: computed parameters for TG type 2.
6. **tg**: indexes of state variable t_g .
7. **tg1**: indexes of state variable t_{g1} .

8. **tg2**: indexes of state variable t_{g2} .
9. **tg3**: indexes of state variable t_{g3} .
10. **pm**: indexes of algebraic variable p_m .
11. **wref**: indexes of algebraic variable ω_{ref} .
12. **u**: connection status.

The droop R and mechanical torque limits are in p.u. with respect to the machine power rating. During initialization, the droop are converted to the system power base, as follows:

$$R_{\text{system}} = \frac{S_{\text{system}}}{S_{\text{machine}}} R_{\text{machine}} \quad (16.2)$$

Mechanical torque 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:

$$0 = p_m - p_m^{\text{syn}} \quad (16.3)$$

$$0 = \omega_{\text{ref0}} - \omega_{\text{ref}} \quad (16.4)$$

where (16.3) represents the link in between the turbine governor and the synchronous machines, being p_m^{syn} the algebraic variable that defines the synchronous machine mechanical power (see also (15.7)). Equation (16.4) defines the turbine governor reference rotor speed.

16.1.1 TG Type I

The TG type I is depicted in Fig. 16.1 and described by the following equations:

$$\begin{aligned} T_{in}^* &= T_{\text{order}} + \frac{1}{R}(\omega_{\text{ref}} - \omega) & (16.5) \\ T_{in} &= \begin{cases} T_{in}^* & \text{if } T_{\min} \leq T_{in}^* \leq T_{\max} \\ T_{\max} & \text{if } T_{in}^* > T_{\max} \\ T_{\min} & \text{if } T_{in}^* < T_{\min} \end{cases} \\ \dot{t}_{g1} &= (T_{in} - t_{g1})/T_s \\ \dot{t}_{g2} &= ((1 - \frac{T_3}{T_c})t_{g1} - t_{g2})/T_c \\ \dot{t}_{g3} &= ((1 - \frac{T_4}{T_5})(t_{g2} + \frac{T_3}{T_c}t_{g1}) - t_{g3})/T_5 \\ T_{\text{mech}} &= t_{g3} + \frac{T_4}{T_5}(t_{g2} + \frac{T_3}{T_c}t_{g1}) \end{aligned}$$

Table 16.1 depicts the data format of the TG type I.

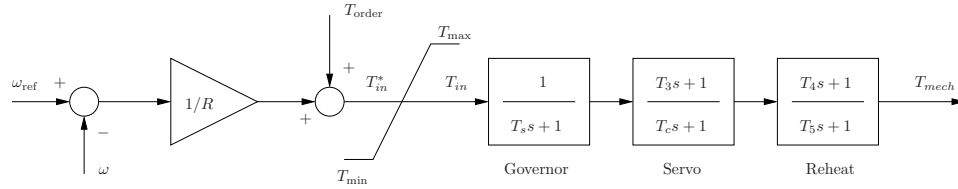


Figure 16.1: Turbine governor type I.

Table 16.1: Turbine Governor Type I Data Format (`Tg.con`)

Column	Variable	Description	Unit
1	-	Generator number	int
2	1	Turbine governor type	int
3	ω_{ref0}	Reference speed	p.u.
4	R	Droop	p.u.
5	T_{max}	Maximum turbine output	p.u.
6	T_{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	$\{0, 1\}$

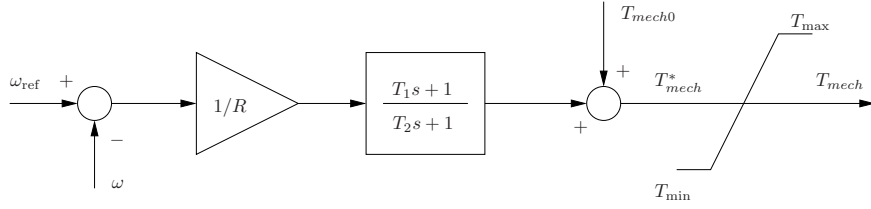


Figure 16.2: Turbine governor type II.

Table 16.2: 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	T_{max}	Maximum turbine output	p.u.
6	T_{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}

16.1.2 TG Type II

The TG type II is depicted in Fig. 16.2 and described by the following equations:

$$\begin{aligned}
 t_g &= \left(\frac{1}{R} \left(1 - \frac{T_1}{T_2} \right) (\omega_{\text{ref}} - \omega) - t_g \right) / T_2 \\
 T_{\text{mech}}^* &= t_g + \frac{1}{R} \frac{T_1}{T_2} (\omega_{\text{ref}} - \omega) + T_{\text{mech}0} \\
 T_{\text{mech}} &= \begin{cases} T_{\text{mech}}^* & \text{if } T_{\text{min}} \leq T_{\text{mech}}^* \leq T_{\text{max}} \\ T_{\text{max}} & \text{if } T_{\text{mech}}^* > T_{\text{max}} \\ T_{\text{min}} & \text{if } T_{\text{mech}}^* < T_{\text{min}} \end{cases}
 \end{aligned} \tag{16.6}$$

Table 16.2 depicts the data format of the TG type II.

16.2 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 structure `Exc`, which has the following fields:

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. `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:

$$0 = v_f - v_f^{\text{syn}} \quad (16.7)$$

$$0 = v_{\text{ref}0} - v_{\text{ref}} \quad (16.8)$$

where (16.7) 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 (15.8)). Equation (16.8) defines the AVR reference voltage.

16.2.1 AVR Type I

The AVR Type I is depicted in Fig. 16.3 and described by the following equations:

$$\dot{v}_m = (V - v_m)/T_r \quad (16.9)$$

$$\dot{v}_{r1} = (\mu_0(1 - \frac{T_2}{T_1})(v_{\text{ref}} - v_m) - v_{r1})/T_1$$

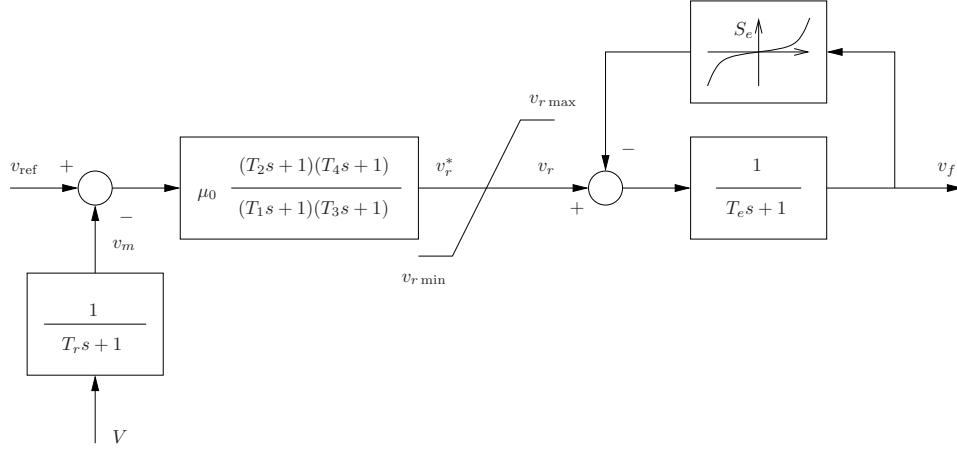


Figure 16.3: Exciter Type I.

$$\begin{aligned}
 \dot{v}_{r2} &= ((1 - \frac{T_3}{T_4})(v_{r1} + \mu_0 \frac{T_2}{T_1}(v_{\text{ref}} - v_m)) - v_{r2})/T_4 \\
 v_r^* &= v_{r2} + \frac{T_3}{T_4}(v_{r1} + \mu_0 \frac{T_2}{T_1}(v_{\text{ref}} - v_m) + v_{r1}) \\
 v_r &= \begin{cases} v_r^* & \text{if } v_{r \min} \leq v_r^* \leq v_{r \max}, \\ v_{r \max} & \text{if } v_r^* > v_{r \max}, \\ v_{r \min} & \text{if } v_r^* < v_{r \min}. \end{cases} \\
 \dot{v}_f &= -(v_f(1 + S_e(v_f)) - v_r)/T_e
 \end{aligned}$$

where the ceiling function S_e is:

$$S_e(v_f) = A_e(e^{B_e|v_f|} - 1) \quad (16.10)$$

Table 16.3 depicts the data format of AVR Type I.

16.2.2 AVR Type II

The AVR Type II is depicted in Fig. 16.4 and described by the following equations:

$$\begin{aligned}
 \dot{v}_m &= (V - v_m)/T_r \\
 \dot{v}_{r1} &= (K_a(v_{\text{ref}} - v_m - v_{r2} - \frac{K_f}{T_f}v_f) - v_{r1})/T_a \\
 v_r &= \begin{cases} v_{r1} & \text{if } v_{r \min} \leq v_{r1} \leq v_{r \max}, \\ v_{r \max} & \text{if } v_{r1} > v_{r \max}, \\ v_{r \min} & \text{if } v_{r1} < v_{r \min}. \end{cases}
 \end{aligned} \quad (16.11)$$

Table 16.3: 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	μ_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	$\{0, 1\}$

$$\begin{aligned}\dot{v}_{r2} &= -(\frac{K_f}{T_f}v_f + v_{r2})/T_f \\ \dot{v}_f &= -(v_f(1 + S_e(v_f)) - v_r)/T_e\end{aligned}$$

where the ceiling function S_e is (16.10). The amplifier block is subjected to an anti-windup limit. Table 16.4 depicts the data format of AVR Type II.

16.2.3 AVR Type III

The AVR Type III is depicted in Fig. 16.5 and described by the following equations:

$$\begin{aligned}\dot{v}_m &= (V - v_m)/T_r \\ \dot{v}_r &= (\mu_0(1 - \frac{T_1}{T_2})(v_{\text{ref}} - v_m) - v_r)/T_2 \\ \dot{v}_f &= ((v_r + \mu_0\frac{T_1}{T_2}(v_{\text{ref}} - v_m) + v_{f0})\frac{V}{V_0} - v_f)/T_e\end{aligned}\tag{16.12}$$

The initial field voltage V_{f0} and bus voltage V_0 are set at the initialization step. The field voltage v_f is subjected to an anti-windup limiter. Table 16.5 depicts the data format of AVR Type III. The signal V/V_0 is disabled if the value of V_0 is set to zero.

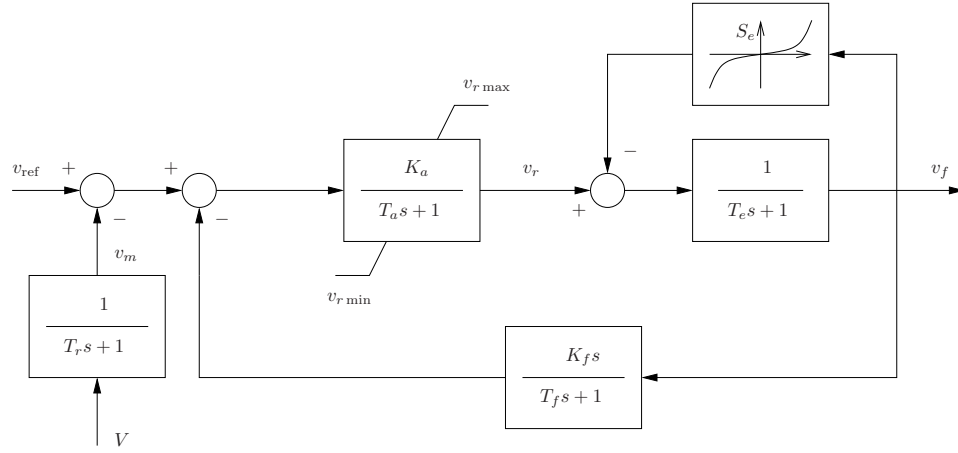


Figure 16.4: Exciter Type II.

Table 16.4: 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	-	(not used)	-
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}

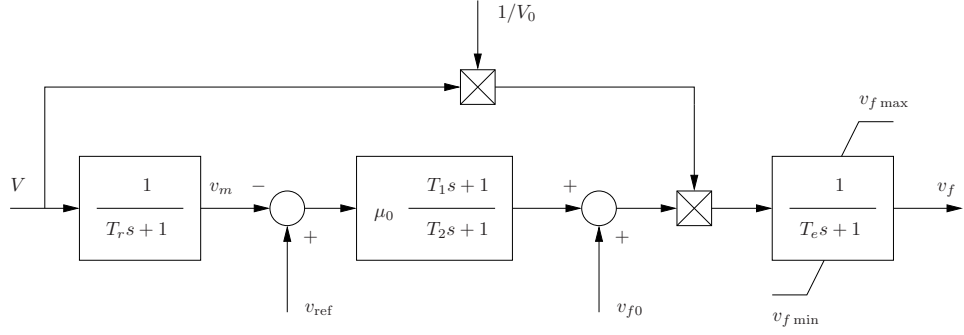


Figure 16.5: Exciter Type III.

Table 16.5: 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	μ_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	V_0	Bus voltage offset	p.u.
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}

16.3 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_g 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.¹ Note that PSSs cannot be used with order II generators.

Each PSS model has two algebraic equations, as follows:

$$0 = g_s(x, y) - v_{ss} \quad (16.13)$$

$$0 = v_{\text{ref}0} - v_{\text{ref}} + v_{ss} \quad (16.14)$$

where (16.13) defines the PSS signal v_{ss} , and (16.14) sums the signal v_{ss} to the AVR reference voltage (see also (16.8)).

PSSs are defined by the structure **Pss**, as follows:

1. **con**: PSS data.
2. **n**: total number of PSSs.
3. **bus**: bus numbers.
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_{ss} .
11. **vref**: indexes of the algebraic variable v_{ref} .
12. **s1**: current status of switches s_1 .
13. **u**: connection status.

The complete PSS data format is depicted in Table 16.6.

¹Observe that T_ϵ is not defined by the user. However it can be changed directly in the function `fm.pss.m`

Table 16.6: 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 $\Rightarrow P_g$, 3 $\Rightarrow V_g$	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	$v_{a_{\min}}^*$	Max additional signal (windup)	p.u.	IV, V
18	$v_{s_{\max}}^*$	Max output signal (before adding v_a)	p.u.	IV, V
19	$v_{s_{\min}}^*$	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

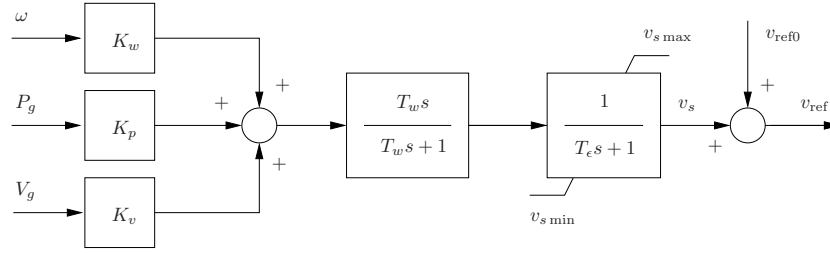


Figure 16.6: Power system stabilizer Type I.

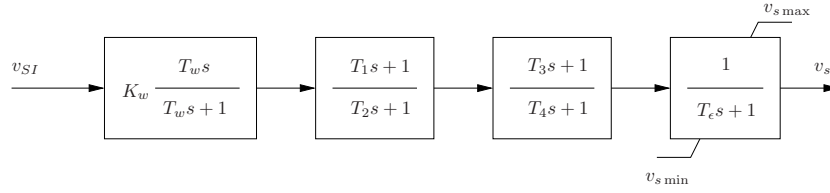


Figure 16.7: Power system stabilizer Type II.

16.3.1 Type I

PSS Type I is depicted in Fig. 16.6, and is described by the following differential equation:

$$\begin{aligned}\dot{v}_1 &= -(K_w \omega + K_p P_g + K_v V_g + v_1)/T_w \\ \dot{v}_s &= (K_w \omega + K_p P_g + K_v V_g + v_1 - v_s)/T_\epsilon\end{aligned}\quad (16.15)$$

where ω , P_g and V_g are the rotor speed, the active power and the voltage magnitude of the generator to which the PSS is connected through the AVR.

16.3.2 Type II

The PSS Type II is depicted in Fig. 16.7, and is described by the equations:

$$\begin{aligned}\dot{v}_1 &= -(K_w v_{SI} + v_1)/T_w \\ \dot{v}_2 &= ((1 - \frac{T_1}{T_2})(K_w v_{SI} + v_1) - v_2)/T_2 \\ \dot{v}_3 &= ((1 - \frac{T_3}{T_4})(v_2 + (\frac{T_1}{T_2}(K_w v_{SI} + v_1))) - v_3)/T_4 \\ \dot{v}_s &= (v_3 + \frac{T_3}{T_4}(v_2 + \frac{T_1}{T_2}(K_w v_{SI} + v_1)) - v_s)/T_\epsilon\end{aligned}\quad (16.16)$$

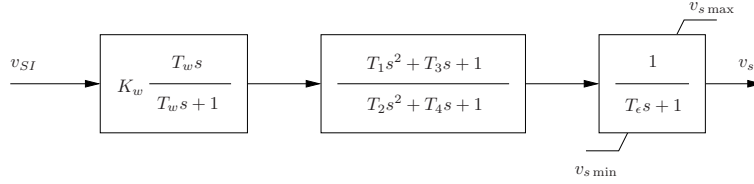


Figure 16.8: Power system stabilizer Type III.

16.3.3 Type III

The PSS Type III is depicted in Fig. 16.8, and is described by the equations:

$$\begin{aligned}
 \dot{v}_1 &= -(K_w v_{SI} + v_1)/T_w \\
 \dot{v}_2 &= a_1 v_3 + a_2 (K_w v_{SI} + v_1) \\
 \dot{v}_3 &= -v_2 + a_3 v_3 + a_4 (K_w v_{SI} + v_1) \\
 \dot{v}_s &= (v_2 + \frac{T_3}{T_4} (K_w v_{SI} + v_1) - v_s)/T_e
 \end{aligned} \tag{16.17}$$

where

$$\begin{aligned}
 a_1 &= \frac{1}{T_4} \\
 a_2 &= \frac{1}{T_4} (T_1 - T_2 \frac{T_3}{T_4}) \\
 a_3 &= -\frac{T_2}{T_4} \\
 a_4 &= 1 - \frac{T_3}{T_4} - \frac{T_2}{T_4} (T_1 - T_2 \frac{T_3}{T_4})
 \end{aligned} \tag{16.18}$$

16.3.4 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. 16.9 and 16.10. 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 $\Delta\omega \geq 0$.

16.4 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

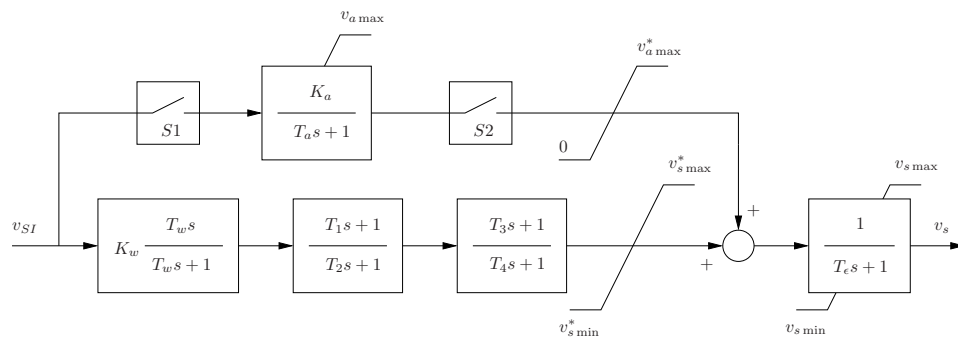


Figure 16.9: Power system stabilizer Type IV.

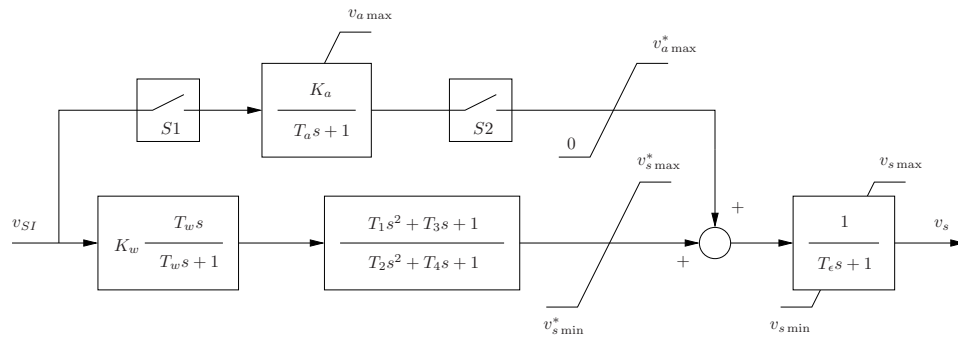


Figure 16.10: Power system stabilizer Type V.

pure integrator, with anti-windup hard limits (see Fig. 16.11). This regulator is generally sleeping, i.e. $v_{\text{OXL}} = 0$, unless the field current is greater than its thermal limit ($i_f > i_{f\text{lim}}$). It is implicitly assumed that at the initial condition given by the power flow solution, all $i_f \leq i_{f\text{lim}}$, thus leading to $v_{\text{OXL}} = 0$ at $t = 0$. If the field 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:

$$\begin{aligned} \dot{v}_{\text{OXL}} &= (i_f - i_{f\text{lim}})/T_0 & \text{if } i_f > i_{f\text{lim}} \\ \dot{v}_{\text{OXL}} &= 0 & \text{if } i_f \leq i_{f\text{lim}} \end{aligned} \quad (16.19)$$

Each OXL model has also two algebraic equations, as follows:

$$0 = \sqrt{(V_g + \gamma_q)^2 + P_g^2} + \left(\frac{x_d}{x_q} + 1\right) \frac{\gamma_q(V_g + \gamma_q) + \gamma_p}{\sqrt{(V_g + \gamma_q)^2 + P_g^2}} - i_f \quad (16.20)$$

$$0 = v_{\text{ref0}} - v_{\text{ref}} + v_{\text{OXL}} \quad (16.21)$$

where

$$\begin{aligned} \gamma_p &= x_q P_g / V_g \\ \gamma_q &= x_q Q_g / V_g \end{aligned}$$

and V_g is the voltage at the generator bus, and P_g and Q_g are the active and the reactive power of the generator, respectively. Equation (16.20) approximates the synchronous machine field current i_f , and (16.21) sums the signal v_{OXL} to the AVR reference voltage (see also (16.8)).

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 16.7.

OXLs are stored in the structure `Oxl`, that has the following fields:

1. **con**: data chart of the `Oxl` 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 `Oxl.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} .

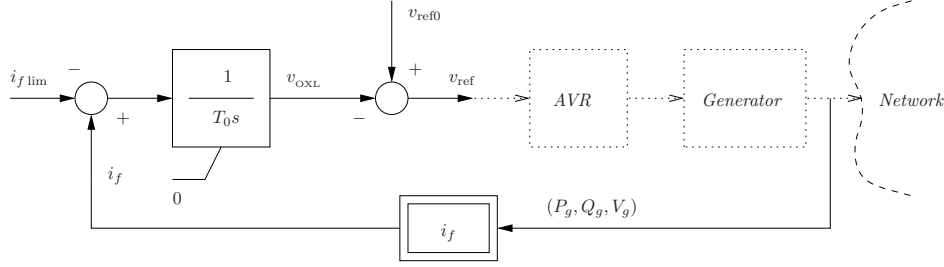


Figure 16.11: Over excitation limiter.

Table 16.7: Over Excitation Limiter Data Format (Oxl.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	$I_{f\text{lim}}$	Maximum field current	p.u.
7	v_{max}	Maximum output signal	p.u.
8	u	Connection status	{0, 1}

9. u : connection status.

The output signal v_{OXL} is added to the reference voltage $v_{\text{ref}0}$ 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.

16.5 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 AVRs and SVCs.² Figure 16.12 depicts the secondary voltage control scheme.

CAC equations are as follows:

$$\begin{aligned}\dot{q}_1 &= K_I(V_{P_{\text{ref}}} - V_P) \\ q &= q_1 + K_P(V_{P_{\text{ref}}} - V_P)\end{aligned}\tag{16.22}$$

²These models were realized in collaboration with Sameh Kamel Mena Kodsí, Ph.D. candidate at University of Waterloo.

whereas the CC equations are:

$$\begin{aligned}\dot{v}_{sg} &= \frac{1}{T_g}(x_{tg} + x_{eqg})(Q_{gr}q - Q_g) \\ \dot{v}_{svc} &= \frac{1}{T_{svc}}x_{eqsvc}(Q_{svcr}q - Q_{svc})\end{aligned}\quad (16.23)$$

where v_{sg} and v_{svc} are the output signals of CCs for AVR and SVC regulators respectively, x_{tg} are the reactances of the transformers connected to the generators and x_{eqg} and x_{eqsvc} are equivalent reactances 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 structures **CAC** and **Cluster**, 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. **u**: connection status

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 AVRs
6. **svc**: indexes of SVCs
7. **Vs**: indexes of the state variable V_s
8. **u**: connection status

Tables 16.8 and 16.9 depicts the data format of Central Area and Cluster Controllers.

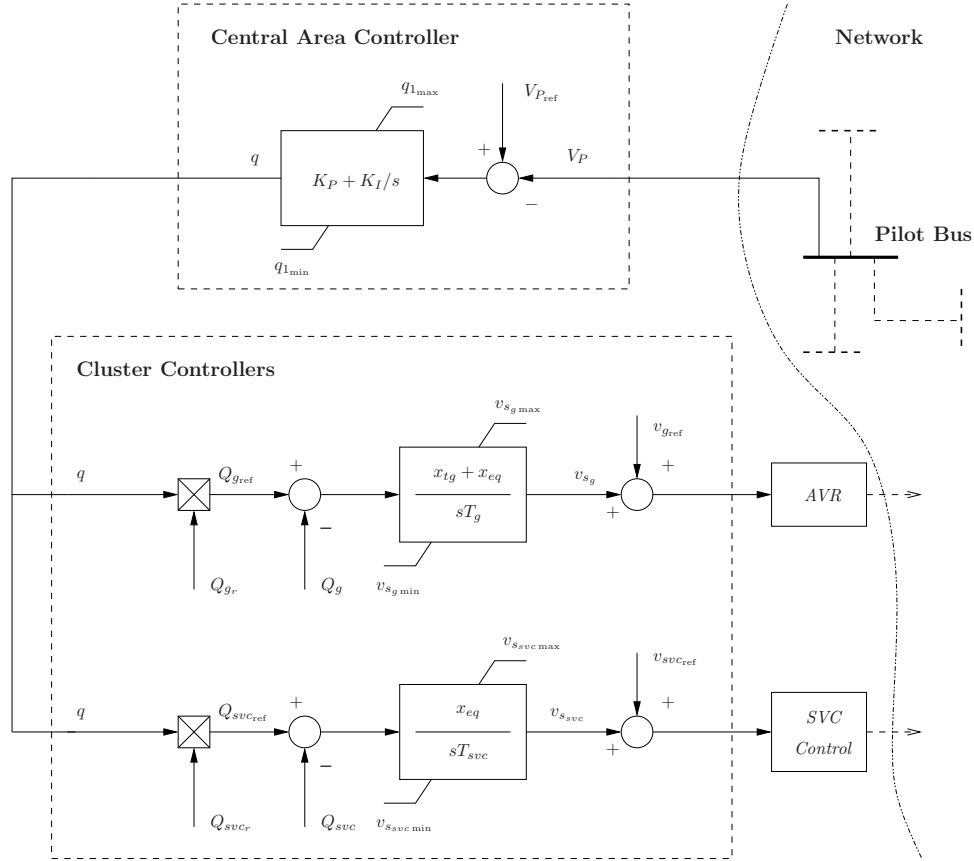


Figure 16.12: Secondary voltage control scheme.

Table 16.8: 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	$q_{1_{max}}$	Maximum output signal	p.u.
9	$q_{1_{min}}$	Minimum output signal	p.u.
10	u	Connection status	{0, 1}

Table 16.9: 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_{tg}	Generator transformer reactance	p.u.
6	x_{eqg} (x_{eqsvc})	Equivalent reactance	p.u.
7	Q_{gr} (Q_{svcr})	Reactive power ratio	p.u.
8	V_{smax}	Maximum output signal	p.u.
9	V_{smin}	Minimum output signal	p.u.
10	u	Connection status	$\{0, 1\}$

16.6 Power Oscillation Damper

This section describes a Power Oscillation Damper (POD). This model has been implemented by Hugo M. Ayres and Marcelo S. Castro.³ An important contribution was given also by Dr. Alberto Del Rosso.⁴ The differential equations as well as the control scheme of the POD are the same as the PSS Type II (see equations (16.16) and Fig. 16.7). The output signal for the POD can be used with SVC, TCSC, STATCOM, SSSC and UPFC components (see Chapter 18 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.

PODs are stored in the structure `Pod`, that has the following fields:

³Hugo 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

⁴Dr. Alberto Del Rosso is with Mercados Energeticos, Buenos Aires, Madrid, Washington D.C. and with National University of Technology, Buenos Aires. E-mail: adelrosso@mercadosenergeticos.com

Table 16.10: Power Oscillation Dampe Data Format (Pod.con)

Column	Variable	Description	Unit
1	-	Bus or line number	int
2	-	FACTS number	int
3	-	<div> <div>Input signal</div> <div> 1 Bus voltage V 2 Line active power P_{ij} 3 Line active power P_{ji} 4 Line current I_{ij} 5 Line current I_{ji} 6 Line reactive power Q_{ij} 7 Line reactive power Q_{ji} </div> </div>	int
4	-	<div> <div>FACTS type</div> <div> 1 SVC 2 TCSC 3 STATCOM 4 SSSC 5 UPFC </div> </div>	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 (used for ω in model I)	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	T_r	Low pass time constant for output signal	s
14	u	Connection status	$\{0, 1\}$

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. v: indexes of the state variable v_1 .
5. v: indexes of the state variable v_2 .
6. v: indexes of the state variable v_3 .
7. Vs: indexes of the state variable v_s .
8. u: connection status.

The data format of PODs components is depicted in Table 16.10.

Chapter 17

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 refactorization.

17.1 Under Load Tap Changer

The equivalent π circuit of the Under Load Tap Changer (ULTC) transformer is depicted in Fig. 17.1. No magnetising shunt is considered. The algebraic equations of the power injections are as follows:

$$\begin{aligned} P_k &= V_k^2 g_T - m V_k V_m (g_T \cos \theta_{km} + b_T \sin \theta_{km}) \\ Q_k &= -V_k^2 b_T - m V_k V_m (g_T \sin \theta_{km} - b_T \cos \theta_{km}) \\ P_m &= m^2 V_m^2 g_T - m V_k V_m (g_T \cos \theta_{km} - b_T \sin \theta_{km}) \\ Q_m &= -m^2 V_m^2 b_T + m V_k V_m (g_T \sin \theta_{km} + b_T \cos \theta_{km}) \end{aligned} \quad (17.1)$$

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 17.2 depicts the ULTC control block diagrams. Three quantities can be controlled, i.e. the secondary voltage V_m (type 1), the reactive power Q_m (type 2), and the remote voltage V_r (type 3).

If the tap ratio step $\Delta m = 0$, the ULTC model is continuous and differential equations are used for the controls. The voltage control equation is as follow:

$$\dot{m} = -Hm + K(V_{m(r)} - v_{\text{ref}}) \quad (17.2)$$

where the negative sign for the error $V_{m(r)} - v_{\text{ref}}$ is due to the stability characteristic of the non-linear control loop. For the reactive power control, a similar equation holds:

$$\dot{m} = -Hm + K(Q_{\text{ref}} + Q_m) \quad (17.3)$$

where it is assumed that Q_m is inductive and injected at the bus m . The tap ratio is subjected to an anti-windup limiter.

If the tap ratio step $\Delta m > 0$, a discrete model is used, as follows:

$$m_{k+1} = m_k + \Delta m R \quad (17.4)$$

where R is a relay type function:

$$R = \begin{cases} 1 & \text{if } u - u_{ref} > \Delta u \\ -1 & \text{if } u - u_{ref} < -\Delta u \\ 0 & \text{if } |u - u_{ref}| \leq \Delta u \end{cases} \quad (17.5)$$

where u is the input signal (voltage or reactive power), u_{ref} the reference signal and Δu the error tolerance.

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 considered equal to 1. Table 17.1 reports the ULTC data format. The bus number r can be $r = 0$ if local voltage or reactive power control are used. ULTC are defined in the structure `Ltc`, as follows:

1. `con`: ULTCs data.
2. `n`: total number of ULTCs.
3. `bus1`: numbers of buses k (primary winding).
4. `bus2`: numbers of buses m (secondary winding).
5. `dat`: ULTC parameters.
6. `m`: indexes of the state variable m .
7. `u`: connection status.

17.2 Load Tap Changer With Embedded Load

Figure 17.3 depicts a simplified model of ULTC with embedded voltage dependent load.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 17.2.

The algebraic equations of the component are as follows:

$$\begin{aligned} P &= -P_n \left(\frac{V}{m} \right)^\alpha \\ Q &= -Q_n \left(\frac{V}{m} \right)^\beta \end{aligned} \quad (17.6)$$

¹A similar, more detailed model can be obtained using ULTCs (Section 17.1) and voltage dependent loads (Section 14.1).

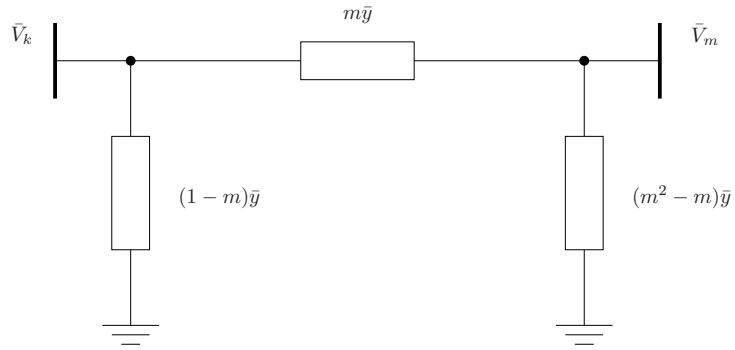


Figure 17.1: Under Load Tap Changer: equivalent π circuit.

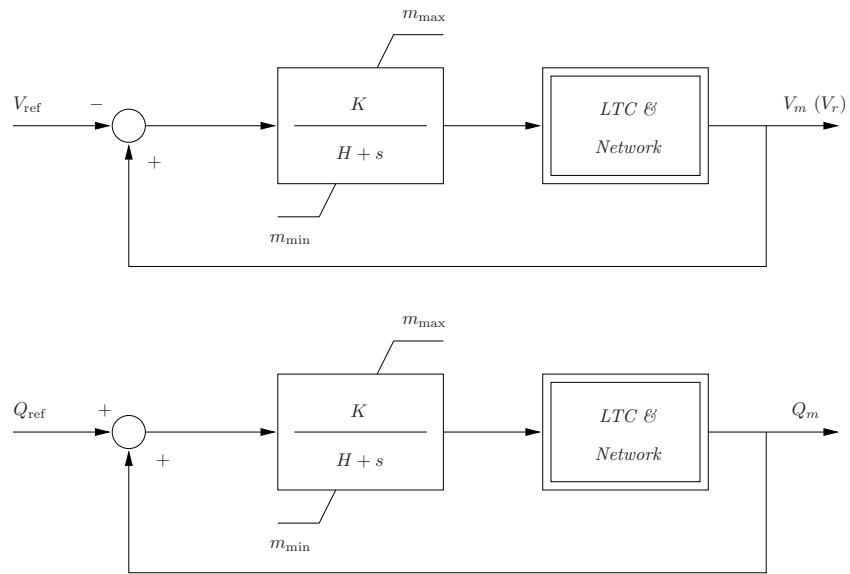


Figure 17.2: Under Load Tap Changer: voltage and reactive power controls.

Table 17.1: 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	int
		2 Reactive power Q_m	
		3 Remote voltage V_r	
17	u	Connection status	$\{0, 1\}$

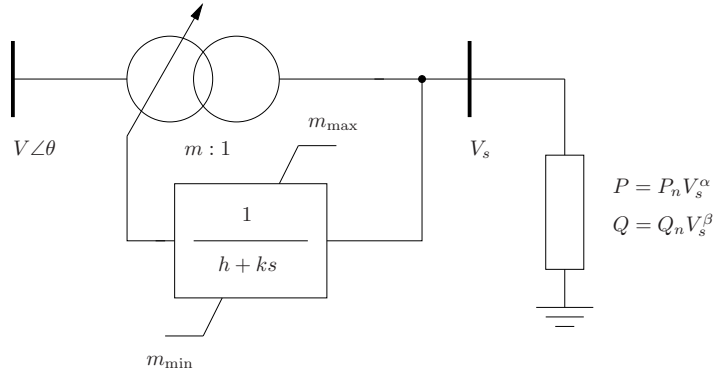


Figure 17.3: Load Tap Changer with embedded load.

and the scalar differential equation is:

$$\dot{m} = -hm + k\left(\frac{V}{m} - v_{\text{ref}}\right) \quad (17.7)$$

ULTCs with embedded voltage dependent load are defined in the structure **Tap**, as follows:

1. **con**: ULTC with embedded load data.
2. **bus**: number of buses to which the ULTCs are connected.
3. **n**: total number of ULTCs.
4. **m**: indexes of the state variable m .
5. **u**: connection status.

17.3 Phase Shifting Transformer

The equivalent circuit of the Phase Shifting Transformer (PST) is depicted in Fig. 17.4. No magnetising shunt is considered. The algebraic equations of the power injections are as follows:

$$\begin{aligned} P_k &= V_k^2 g_T - m V_k V_m (g_T \cos(\theta_{km} - \alpha) + b_T \sin(\theta_{km} - \alpha)) \\ Q_k &= -V_k^2 b_T - m V_k V_m (g_T \sin(\theta_{km} - \alpha) - b_T \cos(\theta_{km} - \alpha)) \\ P_m &= m^2 V_m^2 g_T - m V_k V_m (g_T \cos(\theta_{km} - \alpha) - b_T \sin(\theta_{km} - \alpha)) \\ Q_m &= -m^2 V_m^2 b_T + m V_k V_m (g_T \sin(\theta_{km} - \alpha) + b_T \cos(\theta_{km} - \alpha)) \end{aligned} \quad (17.8)$$

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 17.5 depicts the PST control block diagrams. The measure

Table 17.2: Tap Changer with Embedded Load 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 behaviour	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_n	Nominal active power	p.u.
10	Q_n	Nominal 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}

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:

$$\begin{aligned}\dot{\alpha} &= K_p(P_k - P_{mes})/T_m + K_i(P_{mes} - P_{ref}) \\ \dot{P}_{mes} &= (P_k - P_{mes})/T_m\end{aligned}\quad (17.9)$$

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 PSTs, as this would lock the total real power transfer between the two areas. The data format is reported in Table 17.3.

Phase angle regulating transformers are defined in the structure **Phs**, as follows:

1. **con**: PST data.
2. **n**: total number of PSTs.
3. **bus1**: numbers of buses k (primary winding).
4. **bus2**: numbers of buses m (secondary winding).
5. **alpha**: indexes of the state variable α .
6. **Pm**: indexes of the state variable P_{mes} .
7. **u**: connection status.

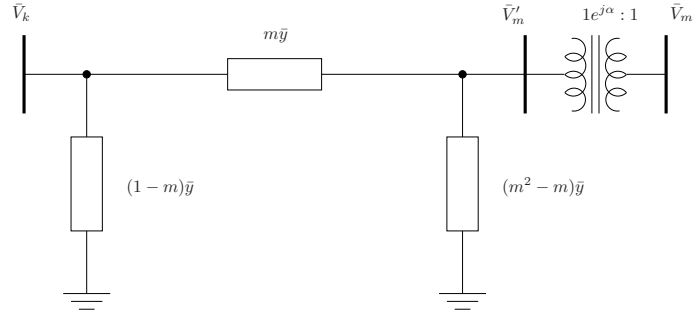


Figure 17.4: Phase shifting transformer circuit.

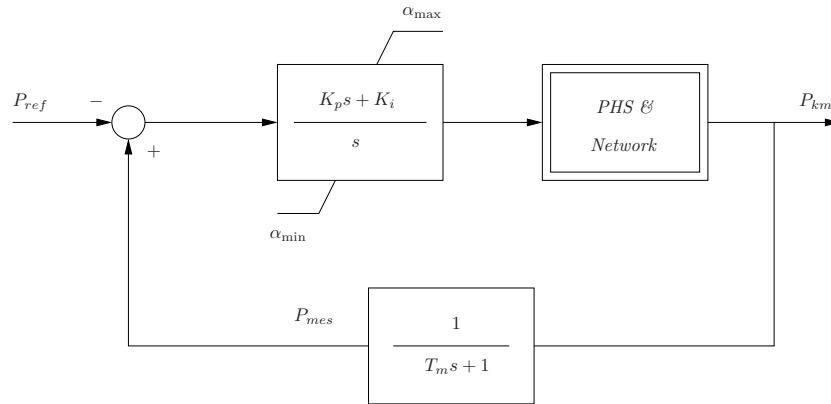


Figure 17.5: Phase shifting transformer control scheme.

Table 17.3: 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 18

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). Each model is described by a set of differential algebraic equations:

$$\begin{aligned}\dot{x}_c &= f_c(x_c, x_s, V, \theta, u) \\ \dot{x}_s &= f_s(x_c, x_s, V, \theta) \\ P &= g_p(x_c, x_s, V, \theta) \\ Q &= g_q(x_c, x_s, V, \theta)\end{aligned}\tag{18.1}$$

where x_c are the control system variables, x_s are the controlled state variables (e.g. firing angles), and the algebraic variables V and θ are the voltage amplitudes and phases at the buses at which the components are connected, they are vectors in case of series components. Finally, the variables u represent the input control parameters, such as reference voltages or reference power flows.

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_{POD} , which is the output of the Power Oscillation Damper described in Section 16.6.

Since version 2, PSAT is provided with new FACTS models, which basically substitute the old ones. These new models have been implemented by Hugo M. Ayres and Marcelo S. Castro.¹ An important contribution was given also by Dr. Alberto

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

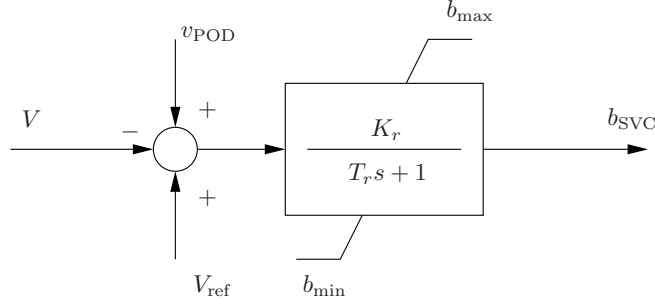


Figure 18.1: SVC Type 1 Regulator.

Alberto Del Rosso.²

18.1 SVC

Two SVC regulators are implemented in the program. The first one assumes a time constant regulator, as depicted in Fig. 18.1. In this model, a total reactance b_{SVC} is assumed and the following differential equation holds:

$$\dot{b}_{SVC} = (K_r(V_{ref} + v_{POD} - V) - b_{SVC})/T_r \quad (18.2)$$

The model is completed by the algebraic equation expressing the reactive power injected at the SVC node:

$$Q = b_{SVC}V^2 \quad (18.3)$$

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 18.1 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:

$$\begin{aligned} \dot{v}_M &= (K_M V - v_M)/T_M \\ \dot{\alpha} &= (-K_D \alpha + K \frac{T_1}{T_2 T_M} (v_M - K_M V) + K(V_{ref} + v_{POD} - v_M))/T_2 \\ Q &= \frac{2\alpha - \sin 2\alpha - \pi(2 - x_L/x_C)}{\pi x_L} V^2 = b_{SVC}(\alpha)V^2 \end{aligned} \quad (18.4)$$

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

²Dr. Alberto Del Rosso is with Mercados Energeticos, Buenos Aires, Madrid, Washington D.C. and with National University of Technology, Buenos Aires.
E-mail: adelrosso@mercadosenergeticos.com

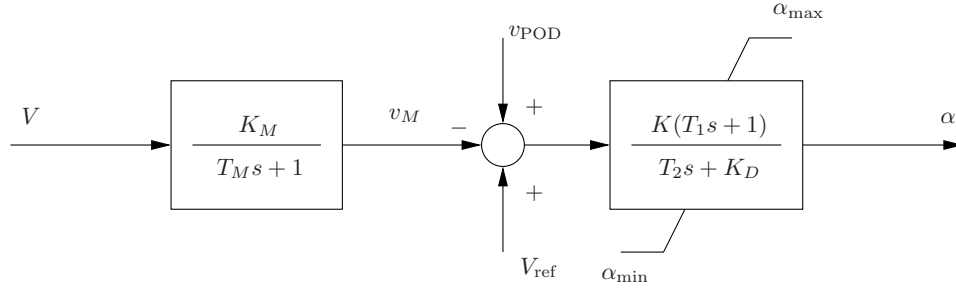


Figure 18.2: SVC Type 2 Regulator.

Table 18.1: 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}

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 18.1 and Fig. 18.2 report the complete data format and the control block diagram for the SVC model 2.

Finally, The SVC components are defined in the structure `Svc` with the following fields:

1. `con`: SVC data.
2. `n`: total number of SVC.
3. `bus`: SVC bus numbers.
4. `vbus`: indexes of bus voltages.
5. `bcv`: indexes of the state variable b_{SVC} .
6. `alpha`: indexes of the state variable α .

Table 18.2: 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	$\alpha_{f \max}$	Maximum firing angle	rad
10	$\alpha_{f \min}$	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}

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. **u**: connection status.

18.2 TCSC

TCSC regulator is depicted in Fig. 18.3. The system undergoes the algebraic equations:

$$\begin{aligned}
 P_{km} &= V_k V_m (Y_{km} + B) \sin(\theta_k - \theta_m) \\
 P_{mk} &= -P_{km} \\
 Q_{km} &= V_k^2 (Y_{km} + B) - V_k V_m (Y_{km} + B) \cos(\theta_k - \theta_m) \\
 Q_{mk} &= V_m^2 (Y_{km} + B) - V_k V_m (Y_{km} + B) \cos(\theta_k - \theta_m)
 \end{aligned} \tag{18.5}$$

where the indexes k and m stand for the sending and receiving bus indices, respectively, and Y_{km} is the admittance of the line at which the TCSC is connected.

The TCSC differential equation are as follows:

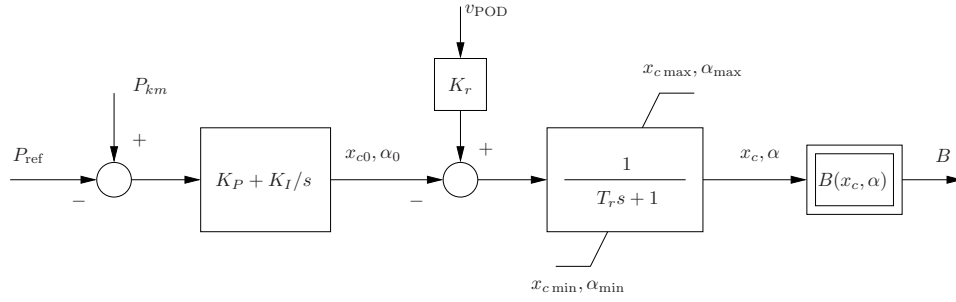


Figure 18.3: TCSC Regulator.

$$\begin{aligned}\dot{x}_1 &= (\{x_{c0}, \alpha_0\} + K_r v_{\text{POD}} - x_1)/T_r \\ \dot{x}_2 &= K_I(P_{km} - P_{\text{ref}})\end{aligned}\quad (18.6)$$

where

$$\{x_{c0}, \alpha_0\} = K_P(P_{km} - P_{\text{ref}}) + x_2 \quad (18.7)$$

The state variable $x_1 = \{x_c, \alpha_0\}$, 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:

$$B(x_c) = -\frac{x_c/x_{km}}{x_{km}(1 - x_c/x_{km})} \quad (18.8)$$

or

$$\begin{aligned}B(\alpha) &= \pi(k_x^4 - 2k_x^2 + 1) \cos k_x(\pi - \alpha) / \\ &\quad \left[x_C \left(\pi k_x^4 \cos k_x(\pi - \alpha) \right. \right. \\ &\quad - \pi \cos k_x(\pi - \alpha) - 2k_x^4 \alpha \cos k_x(\pi - \alpha) \\ &\quad + 2\alpha k_x^2 \cos k_x(\pi - \alpha) - k_x^4 \sin 2\alpha \cos k_x(\pi - \alpha) \\ &\quad + k_x^2 \sin 2\alpha \cos k_x(\pi - \alpha) - 4k_x^3 \cos^2 \alpha \sin k_x(\pi - \alpha) \\ &\quad \left. \left. - 4k_x^2 \cos \alpha \sin \alpha \cos k_x(\pi - \alpha) \right) \right] \\ k_x &= \sqrt{\frac{x_C}{x_L}}\end{aligned}\quad (18.9)$$

During the power flow analysis the TCSC is modeled as a constant capacitive reactance that modifies the line reactance x_{km} , as follows:

$$x'_{km} = (1 - c_p)x_{km} \quad (18.10)$$

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_c and/or α anti-windup limits is performed. In case of limit violation a warning message is displayed. Table 18.3 reports the data format of TCSCs components.

The `Tcsc` class has the following public fields:

1. `con`: TCSC data.
2. `n`: total number of TCSCs.
3. `line`: line number i .
4. `bus1`: bus numbers k (from).
5. `bus2`: bus numbers 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_0 .
12. `Pref`: reference power flow P_{ref} .
13. `y`: line admittance $1/x_{km}$.
14. `u`: connection status.

The TCSC data format is depicted in Table 18.3.

18.3 STATCOM

The implemented STATCOM model is a current injection model which is based on [36, 95, 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. 18.4 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:

$$\dot{i}_{\text{SH}} = (K_r(V_{\text{ref}} + v_{\text{POD}} - V) - i_{\text{SH}})/Tr \quad (18.11)$$

$$Q = i_{\text{SH}}V \quad (18.12)$$

Table 18.3: TCSC Data Format (`Tcsc.con`)

Column	Variable	Description	Unit
1	i	Line number	int
2	-	Model type 1 Reactance x_C 2 Firing angle α	int
3	-	Operation mode 1 Constant x_C 2 Constant P_{km}	int
4	-	Scheduling strategy 1 Constant P_{km} 2 Constant 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	$x_C^{\max} (\alpha^{\max})$	Maximum reactance (firing angle)	rad
11	$x_C^{\min} (\alpha^{\min})$	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	$\{0, 1\}$

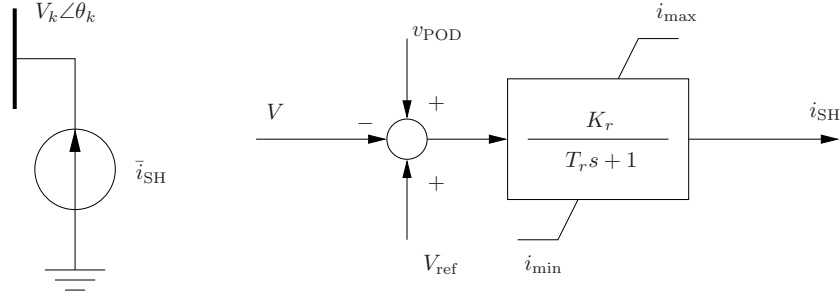


Figure 18.4: STATCOM circuit and control block diagram.

Table 18.4: 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	T_r	Regulator time constant	s
6	K_r	Regulator gain	p.u./p.u.
7	I_{\max}	Maximum current	p.u.
8	I_{\min}	Minimum current	p.u.
9	u	Connection status	$\{0, 1\}$

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. Table 18.4 reports the data and control parameters format for the STATCOM block, and its components are defined in the structure `Statcom` with the following fields:

1. `con`: STATCOM data.
2. `n`: total number of STATCOM.
3. `bus`: STATCOM bus numbers.
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. `u`: connection status.

The STATCOM data format is depicted in Table 18.4.

18.4 SSSC

The implemented SSSC model is based on [109, 67, 74]. The SSSC is represented by a series voltage source \bar{v}_S , as depicted in Fig. 18.5. 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 a SSSC are as follows:

$$\begin{aligned} P_{km} &= (1 + \epsilon) \frac{V_k V_m}{x_{km}} \sin(\theta_k - \theta_m) \\ P_{mk} &= -P_{km} \\ Q_{km} &= (1 + \epsilon) \frac{V_k}{x_{km}} (V_k - V_m \cos(\theta_k - \theta_m)) \\ Q_{mk} &= (1 + \epsilon) \frac{V_m}{x_{km}} (V_m - V_k \cos(\theta_k - \theta_m)) \end{aligned} \quad (18.13)$$

where

$$\epsilon = \frac{v_S}{\sqrt{V_k^2 + V_m^2 - 2V_k v_m \cos \theta_{km}}} \quad (18.14)$$

The SSSC dynamic model is illustrated in Fig. 18.6 and the differential equation that describes the dynamic behavior of the SSSC is:

$$\dot{v}_S = (v_{S0} + v_{\text{POD}-v_S})/T_r \quad (18.15)$$

The input v_{S0} 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 voltage v_{S0} 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 $v_{S0} = \text{const.}$
- 2) **Constant reactance:** The magnitude of the voltage v_S varies proportionally to the line current keeping constant the total impedance (reactance in fact) of the transmission line where the SSSC is installed. In this operation mode the input v_{S0} is as follows:

$$v_{S0} = k x_{km} I_{km} \quad (18.16)$$

where k is the degree of series compensation, I_{km} is the magnitude of the line current, and x_{km} is the reactance of the transmission line.

- 3) **Constant power flow:** Constant power control mode: In this mode, the voltage v_{S0} is the output of a PI controller used to control the power flow in transmission systems, as shown in Fig. 18.6. Two strategies are implemented for the constant power flow control mode:

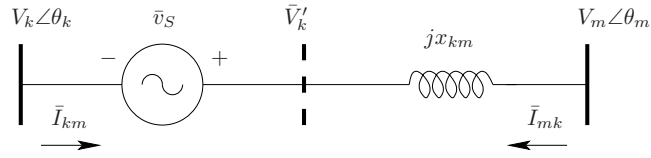


Figure 18.5: SSSC circuit.

- 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.

The SSSC components are stored in the structure **Sssc**, which has the following fields:

1. **con**: SSSC data.
2. **n**: total number of SSSC.
3. **line**: line numbers i .
4. **bus1**: bus numbers k (from).
5. **bus2**: bus numbers 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 $1/x_{km}$.
14. **u**: connection status.

Table 18.5 reports the complete data format for the SSSC.

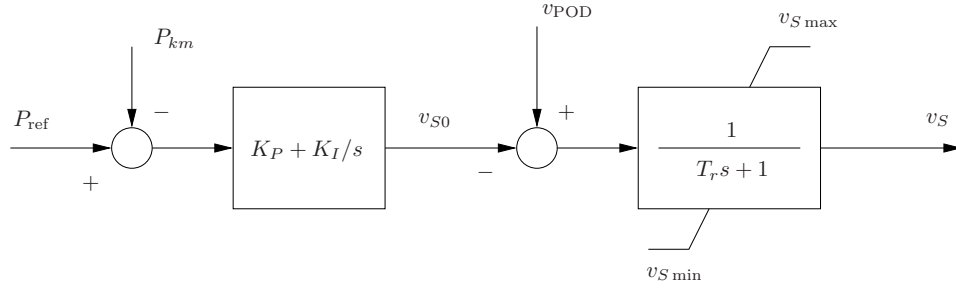


Figure 18.6: SSSC control block diagram.

Table 18.5: SSSC Data Format (`Sssc.con`)

Column	Variable	Description	Unit
1	i	Line number	int
2	-	Operation mode 1 Constant voltage 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
9	v_s^{\max}	Maximum series voltage v_s	p.u.
10	v_s^{\min}	Minimum series voltage v_s	p.u.
11	-	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}

18.5 UPFC

The implemented UPFC model is based on [89, 66, 73]. 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. 18.7. The series voltage source and the shunt current source are defined as follows:

$$\begin{aligned}\bar{v}_S &= (v_p + v_q)e^{j\phi} = r\bar{V}_k e^{j\gamma} \\ \bar{i}_{SH} &= (i_p + i_q)e^{j\theta_k}\end{aligned}\quad (18.17)$$

The equivalent circuit vector diagram of series voltage source is shown in Fig. 18.8, and the power equations that describe the power injection model of the UPFC are:

$$\begin{aligned}P_{km} &= brV_k V_m \sin(\gamma + \theta_k - \theta_m) \\ Q_{km} &= brV_k^2 \cos \gamma - i_q V_k \\ P_{mk} &= -brV_k V_m \sin(\gamma + \theta_k - \theta_m) \\ Q_{mk} &= -brV_k V_m \cos(\gamma + \theta_k - \theta_m)\end{aligned}\quad (18.18)$$

The UPFC dynamic model has a 3rd order, as depicted in Fig. 18.9. 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:

$$\begin{aligned}\dot{v}_p &= \frac{1}{T_r}(v_{p0} + u_1 v_{POD} - v_p) \\ \dot{v}_q &= \frac{1}{T_r}(v_{q0} + u_2 v_{POD} - v_q) \\ \dot{i}_q &= \frac{1}{T_r}[K_r(V_{ref} + u_3 v_{POD} - V_k) - i_q]\end{aligned}\quad (18.19)$$

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_{p0} 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_{q0} 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 .

The UPFC components are stored in the structure `Upfc`, which has the following fields:

1. `con`: UPFC data.
2. `n`: total number of UPFC.
3. `line`: line numbers i .
4. `bus1`: bus numbers k (from).
5. `bus2`: bus numbers 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_{p0} .
10. `vq0`: initial compensation voltage v_{q0} .
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. `u`: connection status.

Table 18.6 illustrates the complete UPFC data format.

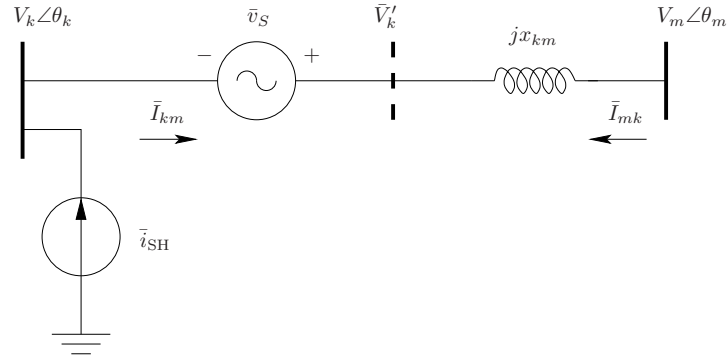


Figure 18.7: UPFC circuit.

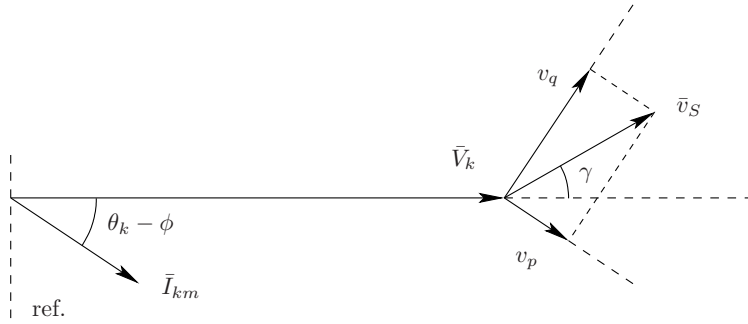


Figure 18.8: UPFC phasor diagram.

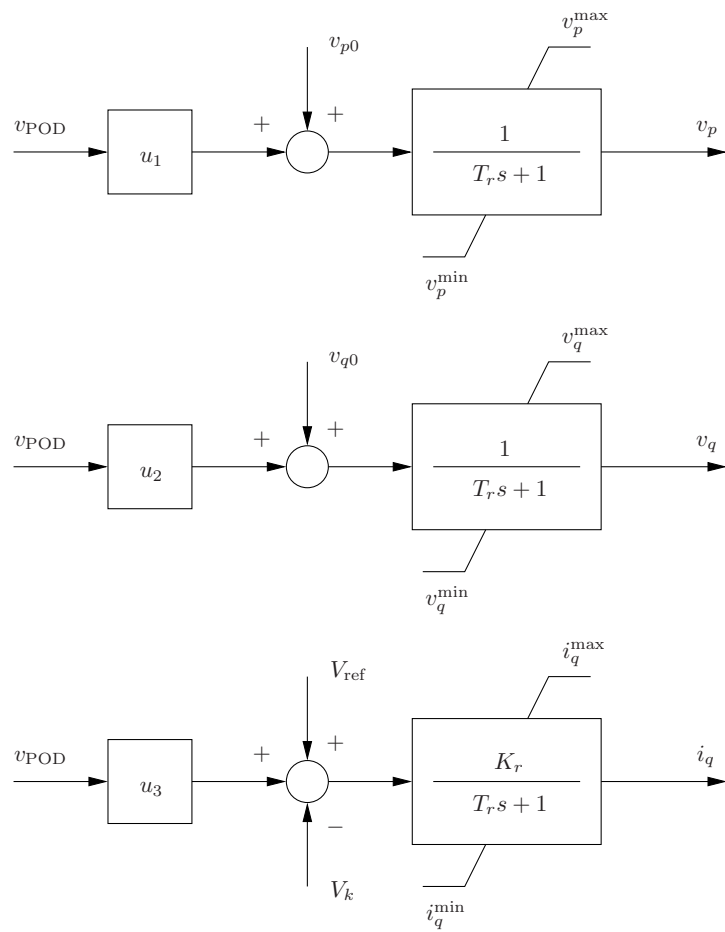


Figure 18.9: UPFC control block diagrams.

Table 18.6: 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}

18.6 HVDC

A simple HVDC system is implemented in PSAT, representing two ac/dc converters connected by a single dc line (see Fig. 18.10). 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. 18.11. The controllers regulate the current or the power flow in the dc line. By default, the dc current I_{dc} is assumed to flows fomr 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 $\gamma = \gamma_{\min}$.

The differential and algebraic equations of the HVDC are as follows:

$$\begin{aligned}
 \dot{I}_{dc} &= (V_{R_{dc}} - V_{I_{dc}} - R_{dc}I_{dc})/L_{dc} \\
 \dot{x}_R &= K_I(y_R - I_{dc}) \\
 \dot{x}_I &= K_I(I_{dc} - y_I) \\
 P_R &= \frac{V_{n_{dc}}I_{n_{dc}}}{S_n} V_{R_{dc}} I_{dc} \\
 Q_R &= \sqrt{S_R^2 - P_R^2} \\
 P_I &= \frac{V_{n_{dc}}I_{n_{dc}}}{S_n} V_{I_{dc}} I_{dc} \\
 Q_I &= \sqrt{S_I^2 - P_I^2} \\
 \cos \alpha &= x_R + K_P(y_R - I_{dc}) \\
 V_{R_{dc}} &= \frac{3\sqrt{2}}{\pi} V_R \cos \alpha - \frac{3}{\pi} X_{t_R} I_{dc} \\
 S_R &= \frac{3\sqrt{2}}{\pi} \frac{V_{n_{dc}}I_{n_{dc}}}{S_n} V_R I_{dc} \\
 y_R &= u_P P_{ord}/V_{R_{dc}} + u_I I_{ord} + u_V V_{ord} \\
 \cos(\pi - \gamma) &= x_I + K_P(I_{dc} - I_{I0}) \\
 V_{I_{dc}} &= \frac{3\sqrt{2}}{\pi} V_I \cos(\pi - \gamma) - \frac{3}{\pi} X_{t_I} I_{dc} \\
 S_I &= \frac{3\sqrt{2}}{\pi} \frac{V_{n_{dc}}I_{n_{dc}}}{S_n} V_I I_{dc} \\
 y_I &= u_P P_{ord}/V_{I_{dc}} + u_I I_{ord} + u_V V_{ord}
 \end{aligned} \tag{18.20}$$

where the index R is used for the rectifier quantities and I for the inverter ones. $\bar{S}_R = P_R + jQ_R$ and $\bar{S}_I = P_I + jQ_I$ are the compex power injected from the ac grid at the rectifier and the inverter sides. 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. $V_{R_{dc}}$ 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

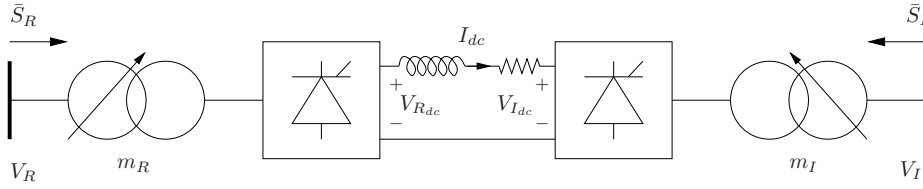


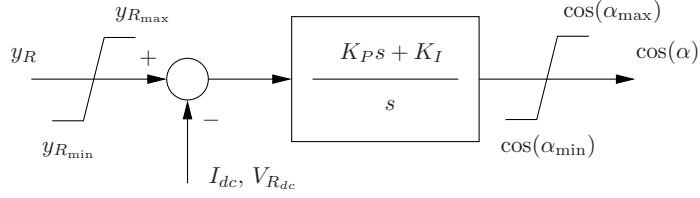
Figure 18.10: HVDC scheme.

and the inverter controllers. u_P , u_I and u_V are 1 or 0, depending of the selected control type. Other parameters are defined in Table 18.7. The HVDC components are stored in the structure `Hvdc`, which has the following fields:

1. `con`: HVDC data.
2. `n`: total number of HVDC.
3. `bus1`: converter bus numbers R .
4. `bus2`: inverter bus numbers I .
5. `dat`: HVDC parameters.
6. `Idc`: indexes of the state variable I_d .
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_{Rdc} .
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_{R0} .
16. `Ii0`: indexes of the algebraic variable I_{I0} .
17. `u`: connection status u .

Table 18.7 reports the complete data format for the HVDC.

Rectifier



Inverter

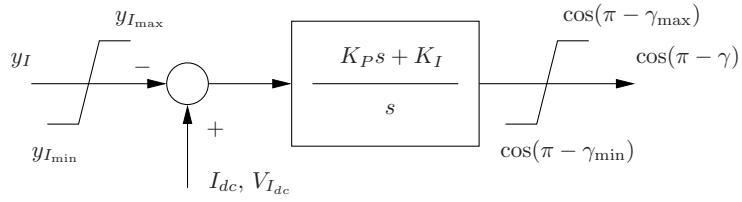


Figure 18.11: HVDC current control.

Note on the per unit system for dc quantities

In [68], the following per unit system is defined for the the dc quantities:

$$\begin{aligned}
 V_{dc}^{\text{base}} &= \frac{3\sqrt{2}}{\pi} V_{ac}^{\text{base}} \\
 I_{dc}^{\text{base}} &= I_{dc}^{\text{rate}} = I_{dc}^n \\
 S_{dc}^{\text{base}} &= V_{dc}^{\text{base}} I_{dc}^{\text{base}} \\
 Z_{dc}^{\text{base}} &= V_{dc}^{\text{base}} / I_{dc}^{\text{base}}
 \end{aligned} \tag{18.21}$$

In Table 18.7, dc p.u. data are referred to the dc voltage and the dc current rates. The ac power rate is considered equal to the ac one, i.e. $S_{dc}^{\text{base}} = S_n$. In order to avoid inconsistencies, it should be $S_n \approx V_{dc}^n I_{dc}^n$. Before running the power flow analysis, PSAT converts all ac and dc p.u. data to system bases.

Table 18.7: 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_R^n	ac voltage rate at rectifier side	kV
5	V_I^n	ac voltage rate at inverter side	kV
6	f_n	Frequency rate	Hz
7	V_{dc}^n	dc voltage rate	kV
8	I_{dc}^n	dc current rate	kA
9	X_{tR}	Transformer reactance (rectifier)	p.u.
10	X_{tI}	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	Ω
16	L_{dc}	Inductance of the dc connection	H
17	$\alpha_{R \max}$	Maximum firing angle α	deg
18	$\alpha_{R \min}$	Minimum firing angle α	deg
19	$\gamma_{I \max}$	Maximum extinction angle γ	deg
20	$\gamma_{I \min}$	Minimum extinction angle γ	deg
21	$y_{R \max}$	Maximum reference current or voltage (rectifier)	p.u.
22	$y_{R \min}$	Minimum reference current or voltage (rectifier)	p.u.
23	$y_{I \max}$	Maximum reference current or voltage (inverter)	p.u.
24	$y_{I \min}$	Minimum reference current or voltage (inverter)	p.u.
25	-	Control type (1: current, 2: power)	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 19

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 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_{w0} , which is used as the average wind speed v_{wa} 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 [108].

19.1 Wind Models

Wind speed models included in PSAT are the Weibull distribution and a composite model which includes average speed, ramp, gust and turbulence. Real measurement data can be used as well. Observe that, regardless the wind speed model, the first value of the wind speed sequence will be the initial average speed ($v_w(t_0) = v_{wa}$) as computed at the initialization step of the wind turbines (see Section 19.2).

Table 19.1 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).

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

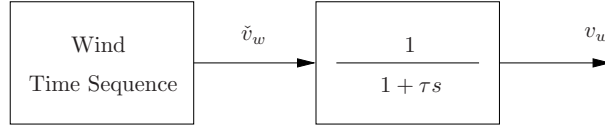


Figure 19.1: Low-pass filter to smooth wind speed variations.

used for calculating the mechanical power of wind turbines are the output of a low-pass filter with time constant τ (see Fig. 19.1), in order to simulate the smoothing of high-frequency wind speed variations over the rotor surface:

$$\dot{v}_m = (\check{v}_w(t) - v_w)/\tau \quad (19.1)$$

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 are stored in the structure `Wind`, with the following fields:

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 .

19.1.1 Weibull Distribution

A common way to describe the wind speed is by means of the Weibull distribution, which is as follows:

$$f(v_w, c, k) = \frac{k}{c^k} v_w^{k-1} e^{-(\frac{v_w}{c})^k} \quad (19.2)$$

where v_w is the wind speed and c and k are constants as defined in the wind model data matrix. Time variations $\nu_w(t)$ of the wind speed are then obtained by means of a Weibull distribution, as follows:

$$\nu_w(t) = \left(-\frac{\ln \iota(t)}{c} \right)^{\frac{1}{k}} \quad (19.3)$$

where $\iota(t)$ is a generator of random numbers ($\iota \in [0, 1]$). Usually the shape factor $k = 2$, which leads to the Rayleigh 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_{wa} determined at the initialization step as mean speed:

$$\check{v}_w(t) = (1 + \nu_w(t) - \widehat{\nu}_w) v_{wa} \quad (19.4)$$

where $\widehat{\nu}_w$ is the mean value of $\nu_w(t)$.

Table 19.1: Wind Speed Data Format (`Wind.con`)

Column	Variable	Description	Unit
1	-	1 Measurement data Wind model 2 Weibull distribution 3 Composite 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 distribution	-
7	k	Shape factor for Weibull distribution	-
8	t_{sr}	Starting ramp time	s
9	t_{er}	Ending ramp time	s
10	v_{wr}	Ramp speed magnitude	m/s
11	t_{sg}	Starting gust time	s
12	t_{eg}	Ending gust time	s
13	v_{wg}	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

19.1.2 Composite Wind Model

A composite wind model is also included in PSAT similar to what proposed in [124, 5]. This model considers the wind as composed of four parts, as follows:

1. average and initial wind speed v_{wa} ;
2. ramp component of the wind speed v_{wr} ;
3. gust component of the wind speed v_{wg} ;
4. wind speed turbulence v_{wt} ;

thus the resulting wind speed \check{v}_w is:

$$\check{v}_w(t) = v_{wa} + v_{wr}(t) + v_{wg}(t) + v_{wt}(t) \quad (19.5)$$

where all components are time-dependent except for the initial average speed v_{wa} .

Wind Ramp Component

The wind ramp component is defined by an amplitude A_{wr} and starting and ending times, t_{sr} and t_{er} respectively:

$$\begin{aligned} t < t_{sr} : & \quad v_{wr}(t) = 0 \\ t_{sr} \leq t \leq t_{er} : & \quad v_{wr}(t) = A_{wr} \left(\frac{t - t_{sr}}{t_{er} - t_{sr}} \right) \\ t > t_{er} : & \quad v_{wr}(t) = A_{wr} \end{aligned} \quad (19.6)$$

Wind Gust Component

The wind gust component is defined by an amplitude A_{wg} and starting and ending times, t_{sg} and t_{eg} respectively:

$$\begin{aligned} t < t_{sg} : & \quad v_{wg}(t) = 0 \\ t_{sg} \leq t \leq t_{eg} : & \quad v_{wg}(t) = \frac{A_{wg}}{2} \left(1 - \cos \left(2\pi \frac{t - t_{sg}}{t_{eg} - t_{sg}} \right) \right) \\ t > t_{eg} : & \quad v_{wg}(t) = A_{wg} \end{aligned} \quad (19.7)$$

Wind Turbulence Component

The wind turbulence component is described by a power spectral density as follows:

$$S_{wt} = \frac{\frac{1}{(\ln(h/z_0))^2} \ell v_{wa}}{\left(1 + 1.5 \frac{\ell f}{v_{wa}}\right)^{\frac{5}{3}}} \quad (19.8)$$

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:

$$\begin{aligned} h < 30 : & \quad \ell = 20h \\ h \geq 30 : & \quad \ell = 600 \end{aligned} \quad (19.9)$$

Table 19.2: Roughness length z_0 for various ground surfaces [92, 107]

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	$1 \div 5$

Table 19.2 depicts roughness values z_0 for various ground surfaces.

The spectral density is then converted in a time domain cosine series as illustrated in [108]:

$$v_{wt}(t) = \sum_{i=1}^n \sqrt{S_{wt}(f_i) \Delta f} \cos(2\pi f_i t + \phi_i + \Delta\phi) \quad (19.10)$$

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.

19.1.3 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 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.

19.2 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 [108]. Figure 19.2 depicts three wind turbines types, while Table 19.3 illustrates a few recent wind turbines data as documented in [46].

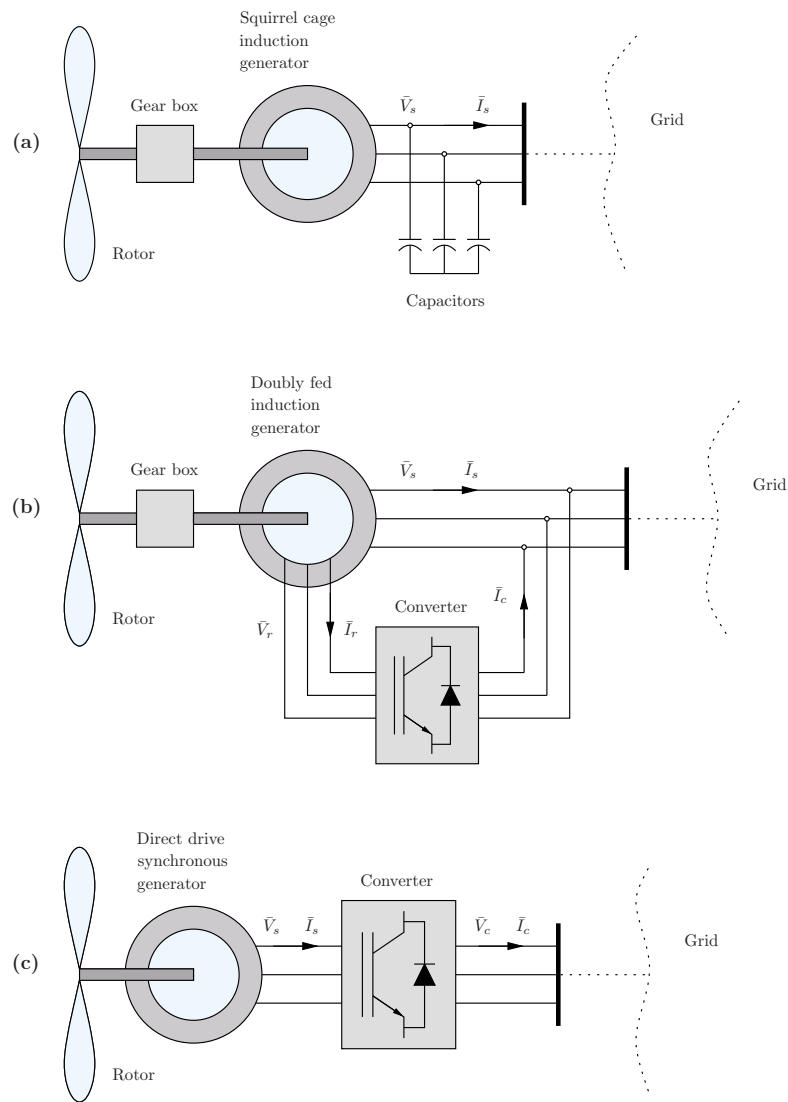


Figure 19.2: Wind turbine types. **(a)** Constant speed wind turbine with squirrel cage induction generator; **(b)** Variable speed wind turbine with doubly fed induction generator; **(c)** Variable speed wind turbine with direct drive synchronous generator.

Table 19.3: 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 TS	gearbox drive two speed	DD PC	direct drive pitch control	VS PS	variable speed shift pitch by stall

19.2.1 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. 15.5, 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 stator machine voltages is as follows:

$$\begin{aligned} v_r &= V \sin(-\theta) \\ v_m &= V \cos(\theta) \end{aligned} \quad (19.11)$$

and the power absorptions are:

$$\begin{aligned} P &= v_r i_r + v_m i_m \\ Q &= v_m i_r - v_r i_m + b_c (v_r^2 + v_m^2) \end{aligned} \quad (19.12)$$

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:

$$\begin{aligned} e'_r - v_r &= r_S i_r - x'_m i_m \\ e'_m - v_m &= r_S i_m + x'_r i_r \end{aligned} \quad (19.13)$$

whereas the link between voltages, currents and state variables is as follows:

$$\begin{aligned} \dot{e}'_r &= \Omega_b (1 - \omega_m) e'_m - (e'_r - (x_0 - x') i_m) / T'_0 \\ \dot{e}'_m &= -\Omega_b (1 - \omega_m) e'_r - (e'_m + (x_0 - x') i_r) / T'_0 \end{aligned} \quad (19.14)$$

where ω_m is the rotor angular speed, and x_0 , x' and T_0 can be obtained from the generator parameters:

$$\begin{aligned} x_0 &= x_S + x_m \\ x' &= x_S + \frac{x_R x_m}{x_R + x_m} \\ T_0' &= \frac{x_R + x_m}{\Omega_b r_R} \end{aligned} \quad (19.15)$$

The mechanical differential equations which take into account the turbine and rotor inertias H_t and H_m , respectively, and shaft stiffness K_s are as follows:

$$\begin{aligned} \dot{\omega}_t &= (T_t - K_s \gamma) / (2H_t) \\ \dot{\omega}_m &= (K_s \gamma - T_e) / (2H_m) \\ \dot{\gamma} &= \Omega_b (\omega_t - \omega_m) \end{aligned} \quad (19.16)$$

where ω_t is the wind turbine angular speed, and the electrical torque T_e is defined as:

$$T_e = e_r' i_r + e_m' i_m \quad (19.17)$$

The mechanical torque T_t is:

$$T_t = \frac{P_w}{\omega_t} \quad (19.18)$$

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:

$$P_w = \frac{\rho}{2} c_p(\lambda) A_r v_w^3 \quad (19.19)$$

in which 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 :

$$\lambda = \frac{v_{bt}}{v_w} = \eta_{GB} \frac{2R\omega_t}{p v_w} \quad (19.20)$$

where η_{GB} is the gear box ratio, p the number of poles of the induction generator and R the rotor radius. Finally, the $c_p(\lambda)$ curve is approximated as follows:

$$c_p = 0.44 \left(\frac{125}{\lambda_i} - 6.94 \right) e^{-\frac{16.5}{\lambda_i}} \quad (19.21)$$

with

$$\lambda_i = \frac{1}{\frac{1}{\lambda} + 0.002} \quad (19.22)$$

To simulate the *tower shadow* effect, a periodic torque pulsation is added to T_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:

$$\tilde{T}_t = T_t \left(1 + 0.025 \sin \left(\eta_{GB} \frac{\Omega_b \omega_t}{n_b} t \right) \right) \quad (19.23)$$

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 **Cswt** structure, which has the following fields:

1. **con**: constant speed wind turbine data.
2. **n**: total number of constant speed wind turbines.
3. **bus**: numbers of buses to which wind turbines are connected.
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. **u**: connection status.

Table 19.4 depicts the data format of the constant speed wind turbine with squirrel cage induction generator.

19.2.2 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:

$$\begin{aligned}
 v_{ds} &= -r_S i_{ds} + ((x_S + x_m) i_{qs} + x_m i_{qr}) \\
 v_{qs} &= -r_S i_{qs} - ((x_S + x_m) i_{ds} + x_m i_{dr}) \\
 v_{dr} &= -r_R i_{dr} + (1 - \omega_m)((x_R + x_m) i_{qr} + x_m i_{qs}) \\
 v_{qr} &= -r_R i_{qr} - (1 - \omega_m)((x_R + x_m) i_{dr} + x_m i_{ds})
 \end{aligned} \tag{19.24}$$

where the stator voltages are functions of the grid voltage magnitude and phase:

$$\begin{aligned}
 v_{ds} &= V \sin(-\theta) \\
 v_{qs} &= V \cos(\theta)
 \end{aligned} \tag{19.25}$$

The generator active and reactive powers depend on the stator and converter currents, as follows:

$$\begin{aligned}
 P &= v_{ds} i_{ds} + v_{qs} i_{qs} + v_{dc} i_{dc} + v_{qc} i_{qc} \\
 Q &= v_{qs} i_{ds} - v_{ds} i_{qs} + v_{qc} i_{dc} - v_{dc} i_{qc}
 \end{aligned} \tag{19.26}$$

Table 19.4: 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_m	Magnetizing reactance	m/s
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	p	Number of poles	int
16	n_b	Number of blades	int
17	η_{GB}	Gear box ratio	-
18	u	Connection status	$\{0, 1\}$

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:

$$\begin{aligned} P_c &= v_{dc}i_{dc} + v_{qc}i_{qc} \\ Q_c &= v_{qc}i_{dc} - v_{dc}i_{qc} \end{aligned} \quad (19.27)$$

whereas, on the rotor side:

$$\begin{aligned} P_r &= v_{dr}i_{dr} + v_{qr}i_{qr} \\ Q_r &= v_{qr}i_{dr} - v_{dr}i_{qr} \end{aligned} \quad (19.28)$$

Assuming a lossless 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:

$$\begin{aligned} P &= v_{ds}i_{ds} + v_{qs}i_{qs} + v_{dr}i_{dr} + v_{qr}i_{qr} \\ Q &= -\frac{x_m V i_{dr}}{x_S + x_m} - \frac{V^2}{x_m} \end{aligned} \quad (19.29)$$

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:

$$\begin{aligned}\dot{\omega}_m &= (T_m - T_e)/2H_m \\ T_e &= \psi_{ds}i_{qs} - \psi_{qs}i_{ds}\end{aligned}\quad (19.30)$$

where the link between stator fluxes and generator currents is as follows:

$$\begin{aligned}\psi_{ds} &= -(x_s + x_m)i_{ds} + x_m i_{dr} \\ \psi_{qs} &= -(x_s + x_m)i_{qs} + x_m i_{qr}\end{aligned}\quad (19.31)$$

Thus the electrical torque T_e results:

$$T_e = x_m(i_{qr}i_{ds} - i_{dr}i_{qs}) \quad (19.32)$$

To simplify computations, the electrical torque T_e is approximated as follows:

$$T_e \approx -\frac{x_m V i_{qr}}{\omega_b(x_s + x_m)} \quad (19.33)$$

where ω_b is the system frequency rate in rad/s. The mechanical torque is:

$$T_m = \frac{P_w}{\omega_m} \quad (19.34)$$

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 . P_w can be approximated as follows:

$$P_w = \frac{\rho}{2} c_p(\lambda, \theta_p) A_r v_w^3 \quad (19.35)$$

in which parameters and variables are the same as in (19.19) and the speed tip ratio λ is defined as in (19.20). The $c_p(\lambda, \theta_p)$ curve is approximated as follows:

$$c_p = 0.22 \left(\frac{116}{\lambda_i} - 0.4\theta_p - 5 \right) e^{-\frac{12.5}{\lambda_i}} \quad (19.36)$$

with

$$\frac{1}{\lambda_i} = \frac{1}{\lambda + 0.08\theta_p} - \frac{0.035}{\theta_p^3 + 1} \quad (19.37)$$

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 19.3 and 19.4. Differential equations for the converter currents are as follows:

$$\begin{aligned}\dot{i}_{qr} &= \left(-\frac{x_s + x_m}{x_m V} P_w^*(\omega_m)/\omega_m - i_{qr} \right) \frac{1}{T_e} \\ \dot{i}_{dr} &= K_V(V - V_{\text{ref}}) - V/x_m - i_{dr}\end{aligned}\quad (19.38)$$

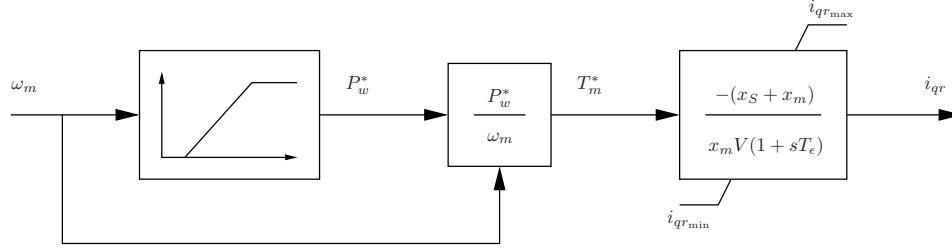


Figure 19.3: Rotor speed control scheme.

where $P_w^*(\omega_m)$ is the power-speed characteristic which roughly optimizes the wind energy capture and is calculated using the current rotor speed value (see Fig. 19.5). It is assumed that $P_w^* = 0$ if $\omega_m < 0.5$ 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. Rotor current limits are computed based on active and reactive limits, and assuming bus voltage $V \approx 1$ as follows:

$$\begin{aligned}
 i_{qr_{\max}} &\approx -\frac{x_S + x_m}{x_m} P_{\min} \\
 i_{qr_{\min}} &\approx -\frac{x_S + x_m}{x_m} P_{\max} \\
 i_{dr_{\max}} &\approx -\frac{x_S + x_m}{x_m} Q_{\min} - \frac{x_S + x_m}{x_m^2} \\
 i_{dr_{\min}} &\approx -\frac{x_S + x_m}{x_m} Q_{\max} - \frac{x_S + x_m}{x_m^2}
 \end{aligned} \tag{19.39}$$

Finally the pitch angle control is illustrated in Fig. 19.6 and described by the differential equation:

$$\dot{\theta}_p = (K_p \phi(\omega_m - \omega_{\text{ref}}) - \theta_p) / T_p \tag{19.40}$$

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 `Dfig` structure, which has the following fields:

1. `con`: doubly fed induction generator data.
2. `n`: total number of doubly fed induction generators.

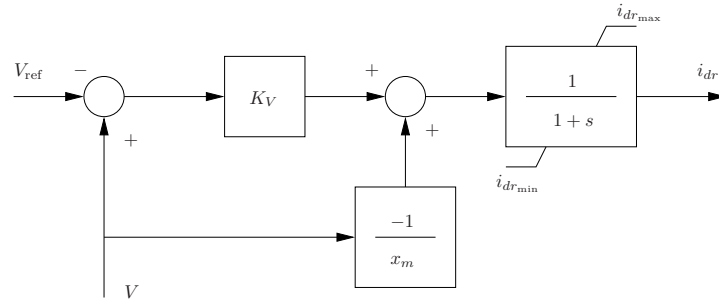


Figure 19.4: Voltage control scheme.

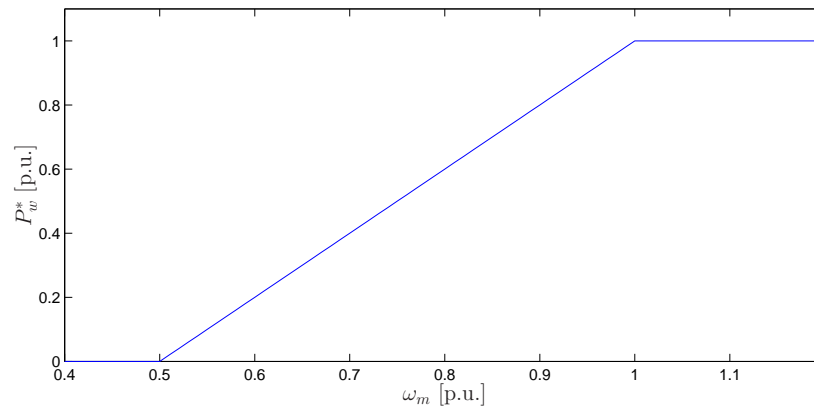


Figure 19.5: Power-speed characteristic.

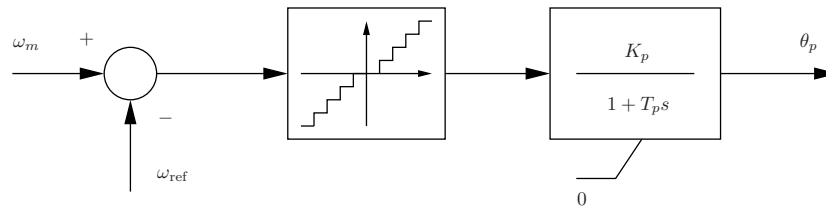


Figure 19.6: Pitch angle control scheme.

Table 19.5: 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_m	Magnetizing reactance	m/s
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	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	u	Connection status	$\{0, 1\}$

3. **bus**: numbers of buses to which generators are connected.
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. **u**: connection status.

Table 19.5 depicts the data format of the wind turbine with doubly fed induction generator.

19.2.3 Direct Drive Synchronous Generator

Steady-state electrical equations of the direct drive synchronous 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:

$$\begin{aligned} v_{ds} &= -r_s i_{ds} + \omega_m x_q i_{qs} \\ v_{qs} &= -r_s i_{qs} - \omega_m (x_d i_{ds} - \psi_p) \end{aligned} \quad (19.41)$$

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:

$$\begin{aligned} P_s &= v_{ds} i_{ds} + v_{qs} i_{qs} \\ Q_s &= v_{qs} i_{ds} - v_{ds} i_{qs} \end{aligned} \quad (19.42)$$

while the active and reactive powers injected into the grid depend only on the grid side currents of the converter:

$$\begin{aligned} P_c &= v_{dc} i_{dc} + v_{qc} i_{qc} \\ Q_c &= v_{qc} i_{dc} - v_{dc} i_{qc} \end{aligned} \quad (19.43)$$

where the converter voltages are functions of the grid voltage magnitude and phase, as follows:

$$\begin{aligned} v_{dc} &= V \sin(-\theta) \\ v_{qc} &= V \cos(\theta) \end{aligned} \quad (19.44)$$

Assuming a lossless converter and a power factor equal to 1, the output powers of the generator becomes:

$$\begin{aligned} P_s &= P_c \\ Q_s &= 0 \end{aligned} \quad (19.45)$$

Furthermore, the reactive power injected in the grid is controlled by means of the converter direct current i_{dc} , which allows rewriting the second equation of (19.43), as follows:

$$Q_c = \frac{1}{\cos(\theta)} V i_{dc} + \tan(\theta) P_s \quad (19.46)$$

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:

$$\begin{aligned} \dot{\omega}_m &= (T_m - T_e)/2H_m \\ T_e &= \psi_{ds} i_{qs} - \psi_{qs} i_{ds} \end{aligned} \quad (19.47)$$

where the link between stator fluxes and generator currents is as follows:

$$\begin{aligned} \psi_{ds} &= -x_d i_{ds} + \psi_p \\ \psi_{qs} &= -x_q i_{qs} \end{aligned} \quad (19.48)$$

The mechanical torque and power are modeled as in the doubly fed induction motor, thus equations from (19.34) to (19.37) 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} , i_{ds} and i_{dc} are state variables and are used for the rotor speed control and the reactive power control and the voltage control, respectively. Differential equations of the converter currents are as follows:

$$\begin{aligned}\dot{i}_{qs} &= (i_{qs\text{ref}} - i_{qs})/T_{\epsilon p} \\ \dot{i}_{ds} &= (i_{ds\text{ref}} - i_{ds})/T_{\epsilon q} \\ \dot{i}_{dc} &= (K_V(V_{\text{ref}} - V) - i_{dc})/T_V\end{aligned}\quad (19.49)$$

where

$$\begin{aligned}i_{qs\text{ref}} &= \frac{P_w^*(\omega_m)}{\omega_m(\psi_p - x_d i_{ds})} \\ i_{ds\text{ref}} &= \frac{\psi_p}{x_d} - \sqrt{\frac{\psi_p^2}{x_d^2} - \frac{Q_{\text{ref}}}{\omega_m x_d}}\end{aligned}\quad (19.50)$$

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. 19.5). 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. Current limits are approximated as follows:

$$\begin{aligned}i_{qs\text{max}} &= -P_{\text{min}} \\ i_{qs\text{min}} &= -P_{\text{max}} \\ i_{ds\text{max}} &= i_{dc\text{max}} = -Q_{\text{min}} \\ i_{ds\text{min}} &= i_{dc\text{min}} = -Q_{\text{max}}\end{aligned}\quad (19.51)$$

Finally the pitch angle control is illustrated in Fig. 19.6 and described by the differential equation (19.40).

The wind turbine with direct drive synchronous generator is defined in the `Ddsg` structure, which has the following fields:

1. `con`: direct drive synchronous generator data.
2. `n`: total number of direct drive synchronous generators.
3. `bus`: numbers of buses to which generators are connected.
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 .

Table 19.6: 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_{ep}	Active power control time constant	s
16	T_{eq}	Reactive power control time constant	s
17	R	Rotor radius	m
18	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	u	Connection status	{0,1}

7. **theta_p**: indexes of the state variable θ_p .

8. **ids**: indexes of the state variable i_{ds} .

9. **iqs**: indexes of the state variable i_{qs} .

10. **idc**: indexes of the state variable i_{dc} .

11. **u**: connection status.

Table 19.5 depicts the data format of the wind turbine with direct drive synchronous generator.

Chapter 20

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.

20.1 Dynamic Shaft

A dynamic mass-spring model is used for defining the shaft of the synchronous machine. Figure 20.1 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 20.1 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:

$$\begin{aligned}\dot{\delta}_{\text{HP}} &= \Omega_b(\omega_{\text{HP}} - 1) \\ \dot{\omega}_{\text{HP}} &= (T_m - D_{\text{HP}}(\omega_{\text{HP}} - 1) - D_{12}(\omega_{\text{HP}} - \omega_{\text{IP}}) \\ &\quad + K_{\text{HP}}(\delta_{\text{IP}} - \delta_{\text{HP}}))/M_{\text{HP}} \\ \dot{\delta}_{\text{IP}} &= \Omega_b(\omega_{\text{IP}} - 1) \\ \dot{\omega}_{\text{IP}} &= (-D_{\text{IP}}(\omega_{\text{IP}} - 1) - D_{12}(\omega_{\text{IP}} - \omega_{\text{HP}}) - D_{23}(\omega_{\text{IP}} - \omega_{\text{LP}}) \\ &\quad + K_{\text{HP}}(\delta_{\text{HP}} - \delta_{\text{IP}}) + K_{\text{IP}}(\delta_{\text{LP}} - \delta_{\text{IP}}))/M_{\text{IP}} \\ \dot{\delta}_{\text{LP}} &= \Omega_b(\omega_{\text{LP}} - 1) \\ \dot{\omega}_{\text{LP}} &= (-D_{\text{LP}}(\omega_{\text{LP}} - 1) - D_{23}(\omega_{\text{LP}} - \omega_{\text{IP}}) - D_{34}(\omega_{\text{LP}} - \omega) \\ &\quad + K_{\text{IP}}(\delta_{\text{IP}} - \delta_{\text{LP}}) + K_{\text{LP}}(\delta - \delta_{\text{LP}}))/M_{\text{LP}}\end{aligned}\tag{20.1}$$

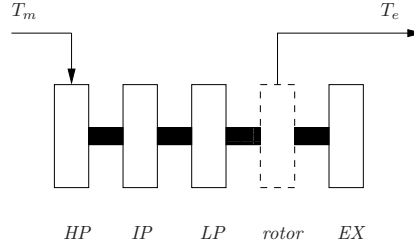


Figure 20.1: Synchronous machine mass-spring shaft model.

$$\begin{aligned}
 \dot{\delta} &= \Omega_b(\omega - 1) \\
 \dot{\omega} &= (-T_e - D(\omega - 1) - D_{34}(\omega - \omega_{LP}) - D_{45}(\omega - \omega_{EX}) \\
 &\quad + K_{LP}(\delta_{LP} - \delta) + K_{EX}(\delta_{EX} - \delta))/M \\
 \dot{\delta}_{EX} &= \Omega_b(\omega_{EX} - 1) \\
 \dot{\omega}_{EX} &= (-D_{EX}(\omega_{EX} - 1) - D_{45}(\omega_{EX} - \omega) \\
 &\quad + K_{EX}(\delta - \delta_{EX}))/M_{EX}
 \end{aligned}$$

Dynamic shafts are defined in the structure **Mass**, as follow:

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. **u**: connection status.

Table 20.1: 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}	Interm.-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	$\{0, 1\}$

20.2 Sub-synchronous Resonance Model

Figure 20.2 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 20.1 and are modeled as high, intermediate and low pressure turbine masses, exciter mass and machine rotor.

This is one of the simplest models [129] 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, 97, 96].

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 [82, 83].

The model used for representing the machine and the line is the same used in [130]. It presents five electrical state variables ($i_d, i_q, i_f, v_{dc}, v_{qc}$) which can be

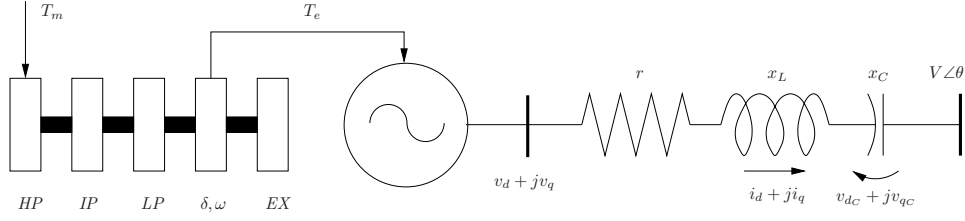


Figure 20.2: Generator with dynamic shaft and compensated line.

determined by the machine differential equations:

$$\begin{aligned}\dot{\psi}_f &= (v_{fd} - i_f)/T'_{d0} \\ \dot{\psi}_d &= \Omega_b(r_a i_d + \omega \psi_q + v_d) \\ \dot{\psi}_q &= \Omega_b(r_a i_q - \omega \psi_d + v_q)\end{aligned}\tag{20.2}$$

the line differential equations:

$$\begin{aligned}\dot{i}_d &= \Omega_b(i_q + (v_d - r i_d - v_{dc} - V \sin(\delta - \theta))/x_L) \\ \dot{i}_q &= \Omega_b(-i_d + (v_q - r i_q - v_{qc} - V \cos(\delta - \theta))/x_L) \\ \dot{v}_{dc} &= \Omega_b(x_C i_d + v_{qc}) \\ \dot{v}_{qc} &= \Omega_b(x_C i_q - v_{dc})\end{aligned}\tag{20.3}$$

along with the algebraic constraints that link the time derivatives of the generator fluxes and of the line currents:

$$\begin{aligned}\dot{\psi}_f &= \dot{i}_f - (x_d - x'_d)\dot{i}_d \\ \dot{\psi}_d &= \dot{i}_f - x_d \dot{i}_d \\ \dot{\psi}_q &= -x_q \dot{i}_q\end{aligned}\tag{20.4}$$

Finally, a five mass system is used for describing the shaft dynamics:

$$\begin{aligned}\dot{\delta}_{HP} &= \Omega_b(\omega_{HP} - 1) \\ \dot{\omega}_{HP} &= (T_m - D_{HP}(\omega_{HP} - 1) + K_{HP}(\delta_{IP} - \delta_{HP}))/M_{HP} \\ \dot{\delta}_{IP} &= \Omega_b(\omega_{IP} - 1) \\ \dot{\omega}_{IP} &= (-D_{IP}(\omega_{IP} - 1) + K_{HP}(\delta_{HP} - \delta_{IP}) \\ &\quad + K_{IP}(\delta_{LP} - \delta_{IP}))/M_{IP} \\ \dot{\delta}_{LP} &= \Omega_b(\omega_{LP} - 1) \\ \dot{\omega}_{LP} &= (-D_{LP}(\omega_{LP} - 1) + K_{IP}(\delta_{IP} - \delta_{LP}) \\ &\quad + K_{LP}(\delta - \delta_{LP}))/M_{LP} \\ \dot{\delta} &= \Omega_b(\omega - 1) \\ \dot{\omega} &= (-T_e - D(\omega - 1) + K_{LP}(\delta_{LP} - \delta))\end{aligned}\tag{20.5}$$

$$\begin{aligned}
& + K_{\text{EX}}(\delta_{\text{EX}} - \delta))/M \\
\dot{\delta}_{\text{EX}} &= \Omega_b(\omega_{\text{EX}} - 1) \\
\dot{\omega}_{\text{EX}} &= (-D_{\text{EX}}(\omega_{\text{EX}} - 1) + K_{\text{EX}}(\delta - \delta_{\text{EX}}))/M_{\text{EX}}
\end{aligned}$$

where the electrical torque is $T_e = \psi_d i_q - \psi_q i_d$. The algebraic equations for the power injections

$$\begin{aligned}
P &= -V i_d \sin(\delta - \theta) - V i_q \cos(\delta - \theta) \\
Q &= -V i_d \cos(\delta - \theta) + V i_q \sin(\delta - \theta)
\end{aligned} \tag{20.6}$$

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:

$$\begin{aligned}
T'_{d0} &= \frac{x_f}{\Omega_b r_f} \\
x'_d &= x_d - x_{ad}
\end{aligned} \tag{20.7}$$

The sub-synchronous resonance generator model is defined in the structure **SSR**, as follows:

1. **con**: SSR data.
2. **bus**: indexes of buses to which SSRs are connected.
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. **E_{dc}**: indexes of the state variable v_{dc} .
8. **E_{qc}**: indexes of the state variable v_{qc} .
9. **T_m**: mechanical torque T_m .
10. **E_{fd}**: 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. **u**: connection status.

The SSR data format is depicted in Table 20.2. 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.

20.3 Solid Oxide Fuel Cell

A Solid Oxide Fuel Cell (SOFC) model is included in PSAT based on what was proposed in [91], [131], [54], and [64].¹ Figure 20.3 depicts the fuel cell scheme, which is based on the following equations:

$$\begin{aligned}
 \dot{p}_{H_2} &= ((q_{H_2} - 2K_r I_k)/K_{H_2} - p_{H_2})/\tau_{H_2} \\
 \dot{p}_{H_2O} &= (2K_r I_k/K_{H_2O} - p_{H_2O})/\tau_{H_2O} \\
 \dot{p}_{O_2} &= ((q_{H_2}/r_{HO} - K_r I_k)/k_{O_2} - p_{O_2})/\tau_{O_2} \\
 \dot{q}_{H_2} &= (2K_r I_k/U_{opt} - q_{H_2})/T_f \\
 \dot{V}_k &= (-V_k - rI_k + N_0(E_0 + \frac{RT}{2F} \ln(p_{H_2}\sqrt{p_{O_2}}/p_{H_2O}))) / T_e
 \end{aligned} \tag{20.8}$$

where R is the gas constant ($R = 8.314$ [J/(mol K)]), F is the Faraday constant ($F = 96487$ [C/mol]), T the absolute gas temperature, and T_e is a “small” time constant which does not affects the fuel cell dynamics. The fuel cell current I_k can be subjected to a constant power control:

$$\dot{I}_k = (P_{ref}/V_k - I_k)/T_e \tag{20.9}$$

or a to constant current control:

$$\dot{I}_k = (P_{ref}/V_{(k_0)} - I_k)/T_e \tag{20.10}$$

where V_{k_0} is the initial fuel cell DC voltage. If the input signal exceeds the dynamic limits proportional to the fuel flow, one has:

$$\dot{I}_k = (\frac{U_{lim} q_{H_2}}{2K_r} - I_k)/T_e \tag{20.11}$$

¹This model was realized in 2002 in collaboration with Valery Knyazkin, Royal Institute of Technology, Sweden.

Table 20.2: 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 ($U_{\text{max}}, U_{\text{min}}$). 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. 20.4. The AC voltage is regulated by means of the inverter modulating amplitude m , as follows:

$$\dot{m} = (K_m(V_{\text{ref}} - V_s) - m)/T_m \quad (20.12)$$

The amplitude control has anti-windup limiters and is depicted in Fig. 20.5.

The DC power of the fuel cell ($P_k = V_k I_k$) is considered to be the real power injected in the network ($P_s = P_k$). Thus the link with the AC network is as follows:

$$\begin{aligned} P_s &= \frac{V_t V_s}{x_T} \sin(\theta_t - \theta_s) = V_k I_k \\ Q_s &= \frac{V_t V_s}{x_T} \cos(\theta_t - \theta_s) - \frac{V_s^2}{x_T} \end{aligned} \quad (20.13)$$

where $V_t = kmV_k$, $k = \sqrt{3/8}$. Thus, one has:

$$\theta_t = \theta_s + \text{asin}\left(\frac{x_T I_k}{kmV_s}\right) \quad (20.14)$$

and, finally:

$$Q_s = -\frac{V_s^2}{x_T} + \frac{V_s kmV_k}{x_T} \sqrt{\left(1 - \left(\frac{x_T I_k}{kmV_s}\right)^2\right)} \quad (20.15)$$

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:

$$\begin{aligned} m_0 &= \frac{x_t}{V_s k V_k} \sqrt{P_g^2 + \left(Q_g + \frac{V_g^2}{x_T}\right)^2} \\ V_{\text{ref}} &= V_g + m/K_m \end{aligned} \quad (20.16)$$

where V_g , P_g and Q_g 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).

The SOFC model is defined in the structure `Sofc`, as follows:

1. `con`: Solid Oxide Fuel Cell data.
2. `bus`: indexes of buses to which SOFCs are connected.
3. `n`: total number of SOFCs.
4. `Ik`: indexes of the state variable I_{dc} .
5. `Vk`: indexes of the state variable V_{dc} .
6. `pH2`: indexes of the state variable p_{H_2} .

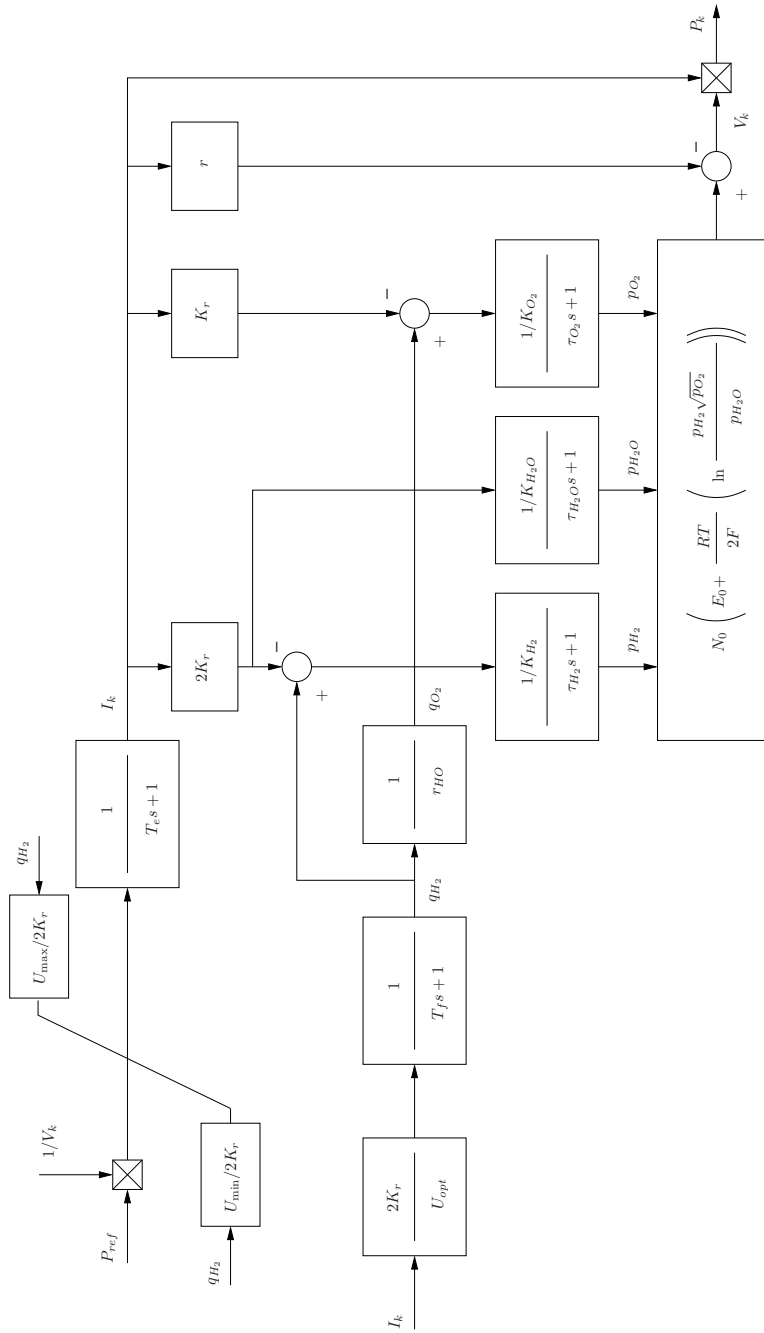


Figure 20.3: Solid Oxide Fuel Cell scheme.

Table 20.3: 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	τ_{H_2}	Response time for hydrogen flow	s
6	K_{H_2}	Valve molar constant for hydrogen	-
7	K_r	Constant	-
8	τ_{H_2O}	Response time for water flow	s
9	K_{H_2O}	Valve molar constant for water	-
10	τ_{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	Ohmic losses	Ω
18	N_0	Number of cells in series in the stack	p.u.
19	E_0	Ideal standard potential	V
20	T	Gas Absolute temperature	K
† 21	P_{ref}	Reference power	p.u.
† 22	V_{ref}	Reference AC voltage	p.u.
† 23	P_B	Base power	MW
† 24	V_B	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.

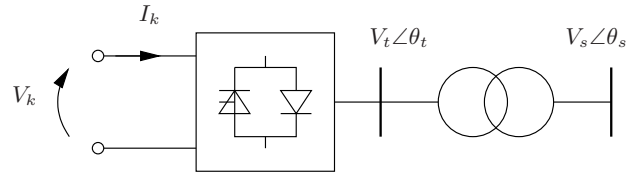


Figure 20.4: Solid Oxide Fuel Cell connection with the AC grid.

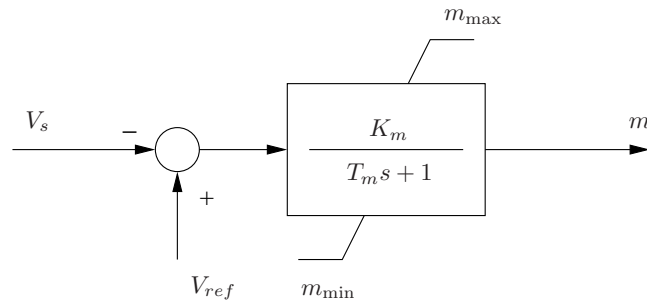
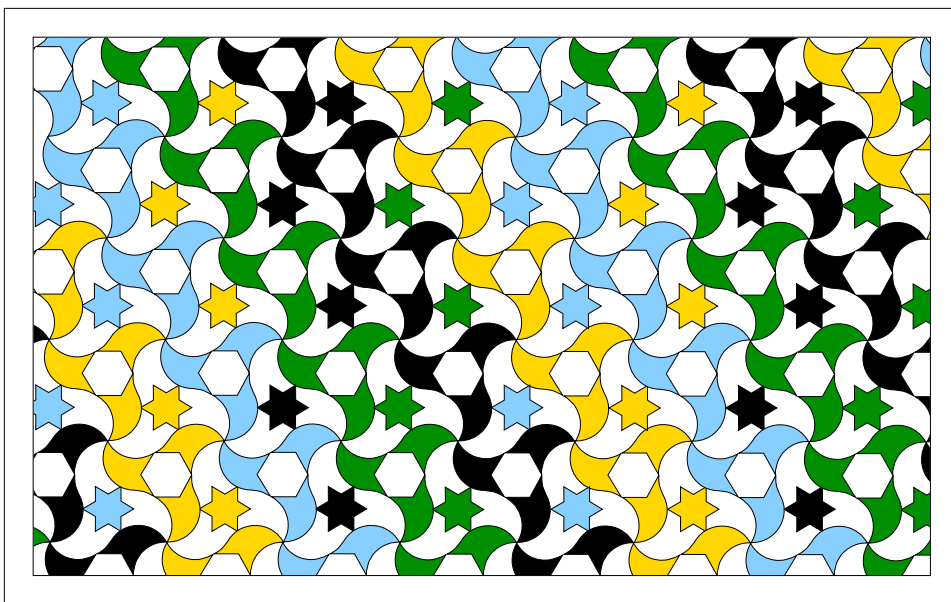


Figure 20.5: AC voltage control for the Solid Oxide Fuel Cell.

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. u: connection status.

Part IV

CAD



Chapter 21

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 21.4 depicts the SIMULINK models of three test systems used in this documentation, i.e. 14-bus, 9-bus and 6-bus test systems.

21.1 Simulink Library

Figure 21.1 depicts the main frame of the PSAT SIMULINK library, which is defined in the file `fm_lib.mdl`, whereas following Figures 21.2, 21.3, 21.4, 21.5, 21.6, 21.7, 21.8, 21.9, 21.10, 21.11, 21.12, and 21.13 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.

Since PSAT 2.0.0, the SIMULINK library makes use of Physical Model Components (PMCs), thus allowing bidirectional connections. Physical connectors are represented by means of circles.

Observe that running time domain simulations from the SIMULINK model menus produces no effect, since no SIMULINK dynamic model is associated with PSAT blocks. Furthermore, only the blocks contained in the PSAT library should be used for building the network.¹

21.2 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

¹The function *Create Subsystem* available in SIMULINK model menu is fully supported.

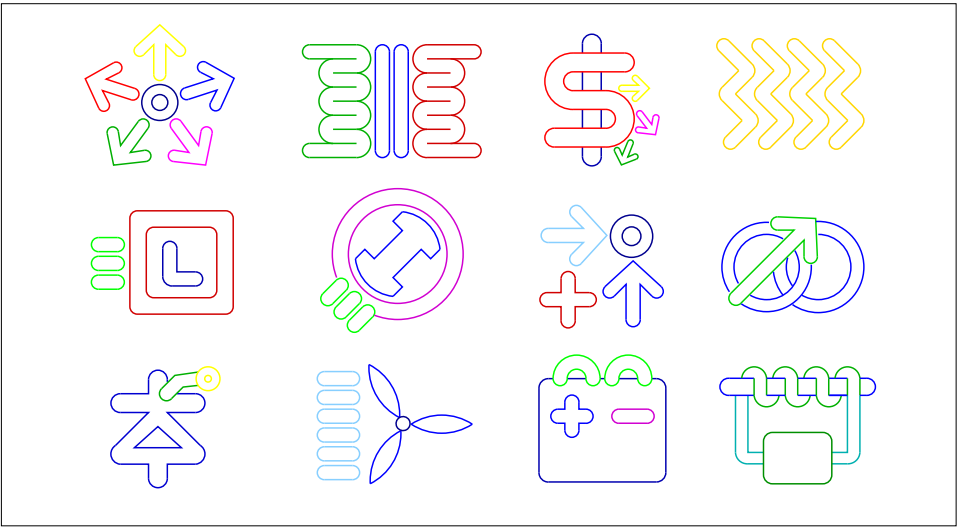


Figure 21.1: Simulink library: Main Window.

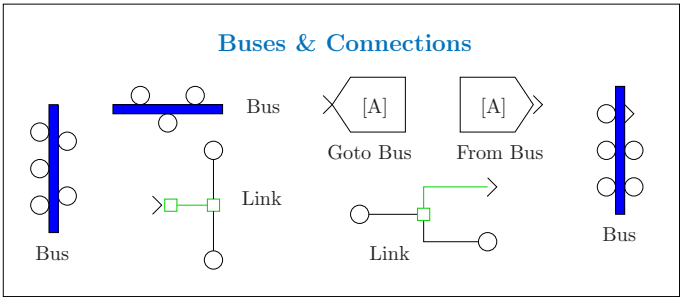


Figure 21.2: Simulink library: Connections.

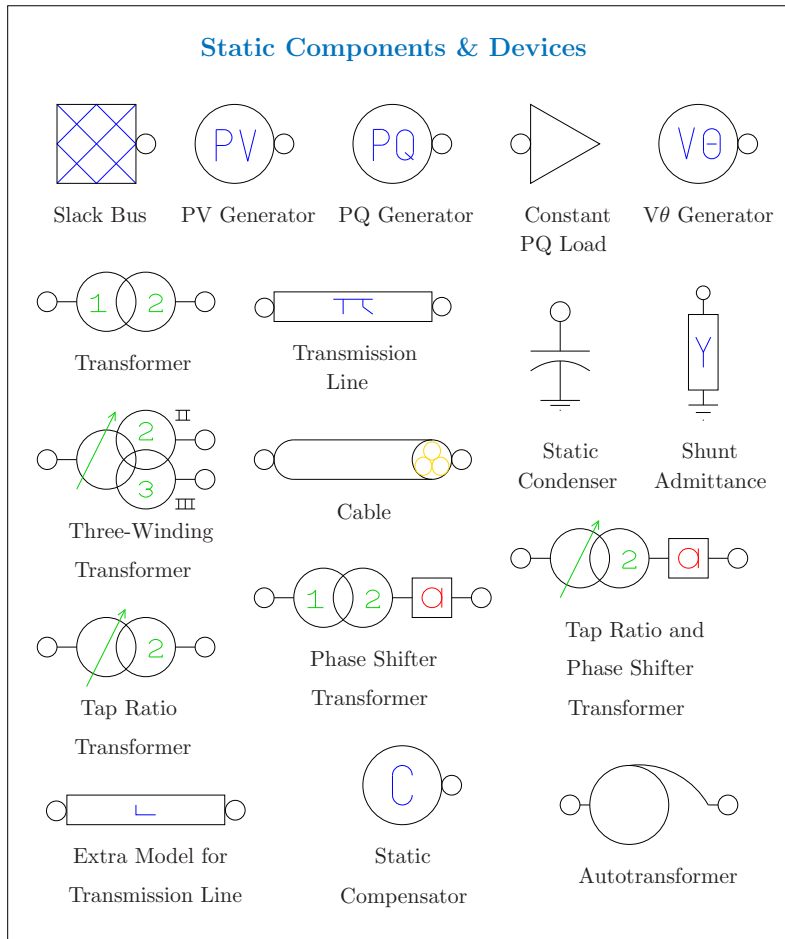


Figure 21.3: Simulink library: Power Flow data.

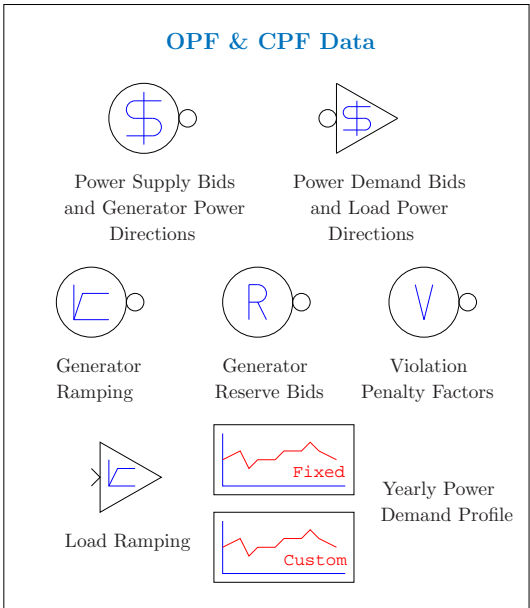


Figure 21.4: Simulink library: OPF & CPF data.

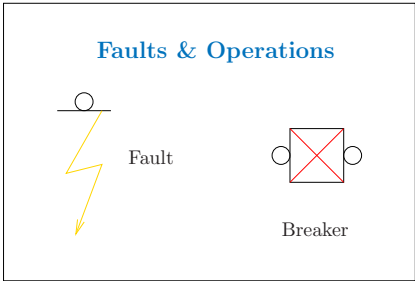


Figure 21.5: Simulink library: Faults & Breakers.

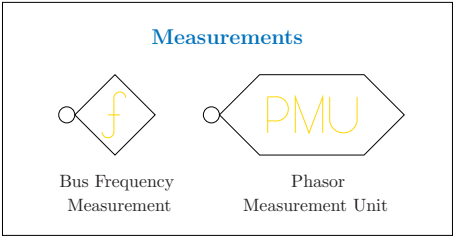


Figure 21.6: Simulink library: Measurements.

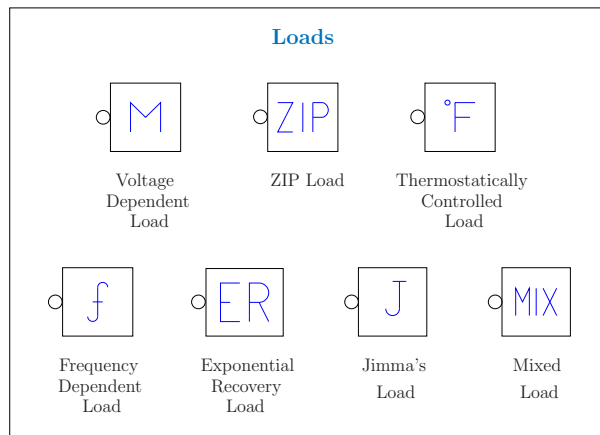


Figure 21.7: Simulink library: Loads.

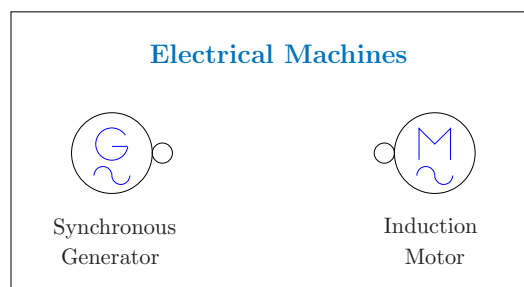


Figure 21.8: Simulink library: Machines.

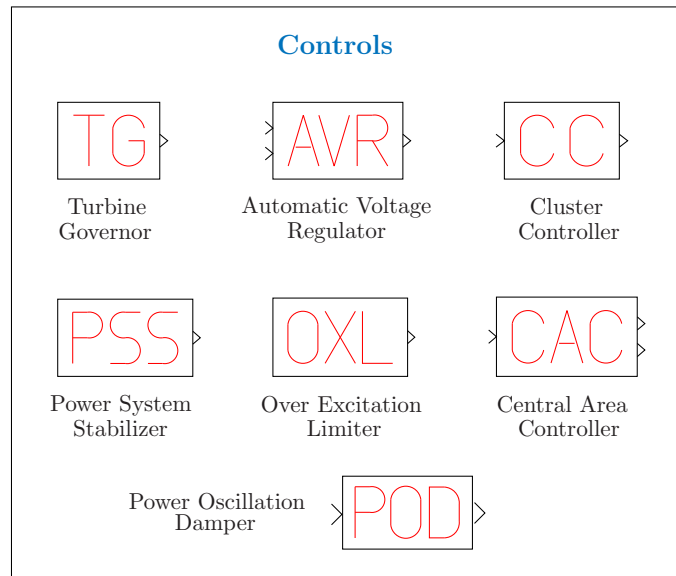


Figure 21.9: Simulink library: Regulators.

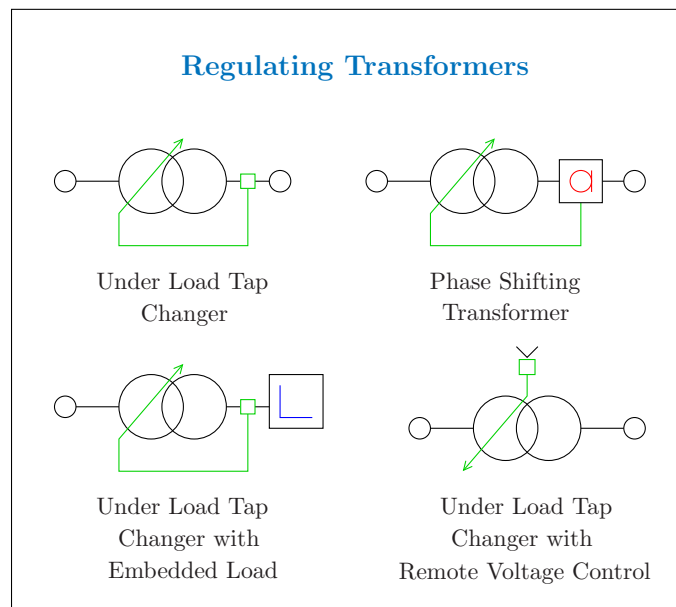


Figure 21.10: Simulink library: Regulating Transformers.

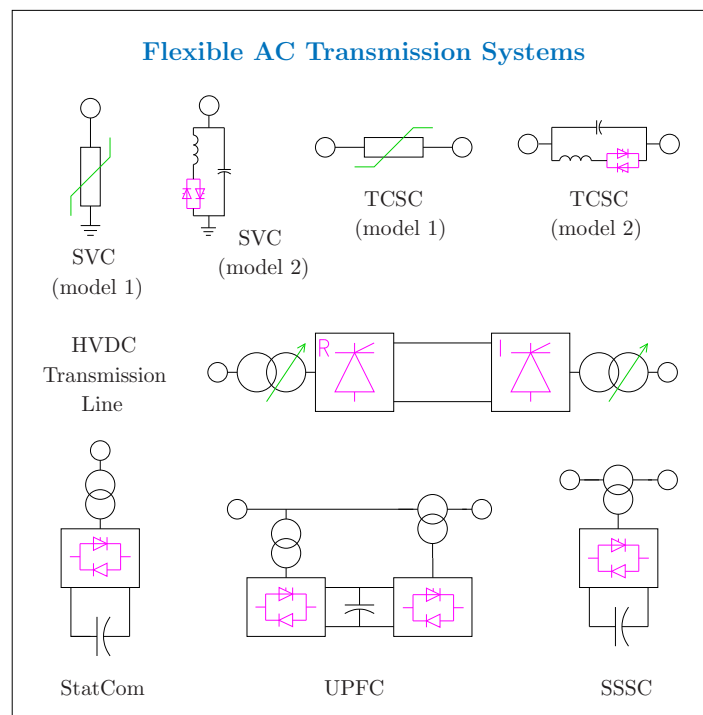


Figure 21.11: Simulink library: FACTS controllers.

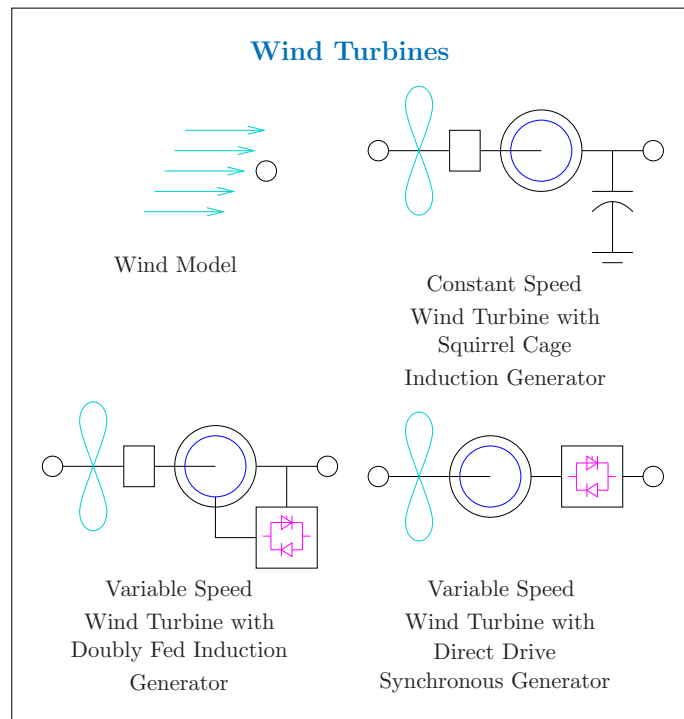


Figure 21.12: Simulink library: Wind Turbines.

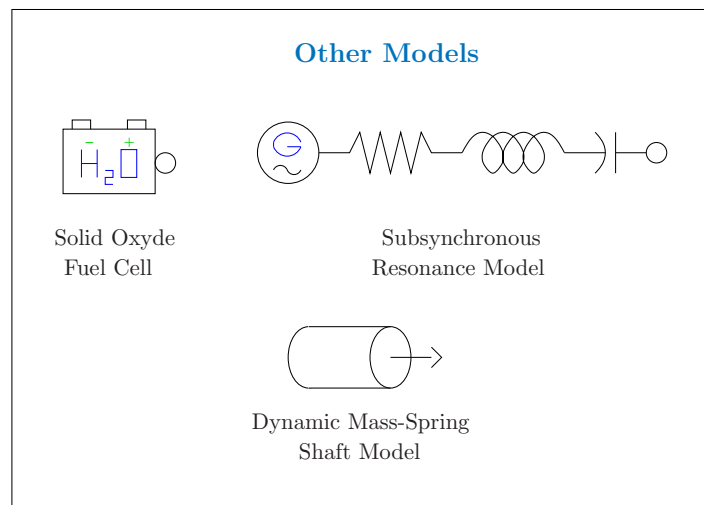


Figure 21.13: Simulink library: Other models.

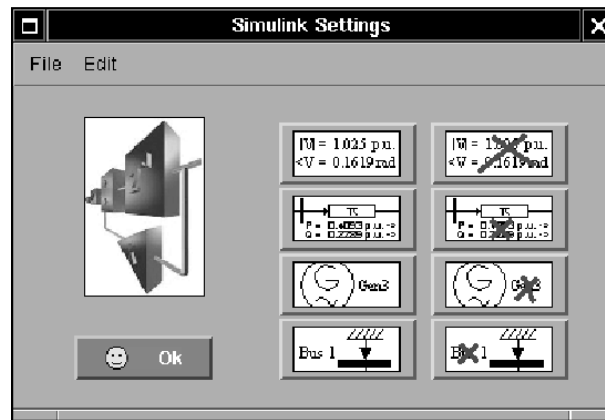


Figure 21.14: GUI for Simulink model settings.

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 SIMULINK models are marked with the flag (mdl).²

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.

21.3 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. 21.14 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 arrows from the resulting *.eps* file (see examples depicted in the next Section 21.4).

21.4 Examples

Figures 21.15, 21.16 and 21.17 depict the SIMULINK models of the 9-bus, 14-bus and 6-bus test systems.³ Figure 21.16 depicts also the bus voltage report generated using the GUI for SIMULINK settings.

²It is possible to convert a SIMULINK model without actually loading the data file, using the *Edit/Simulink Model Conversion* menu in the main window.

³The models are available in the subfolder `tests` of the main PSAT folder.

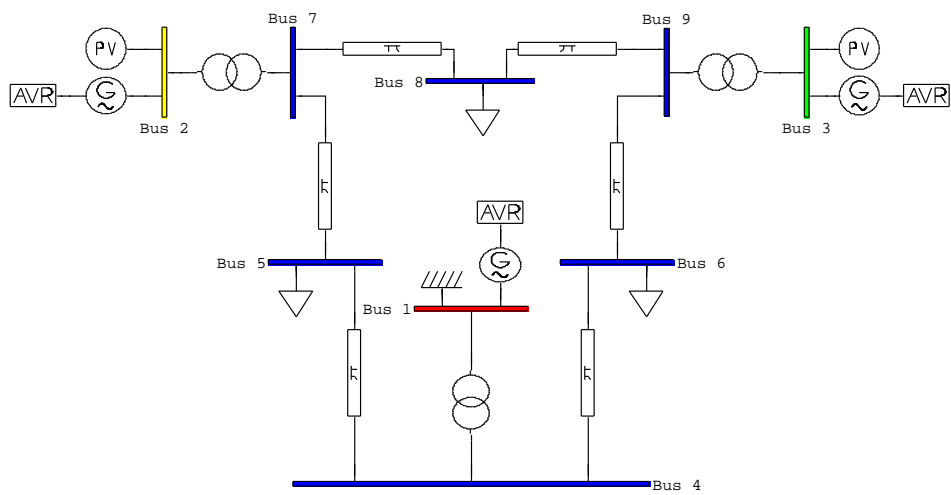


Figure 21.15: Simulink model of the WSCC 3-generator 9-bus test system.

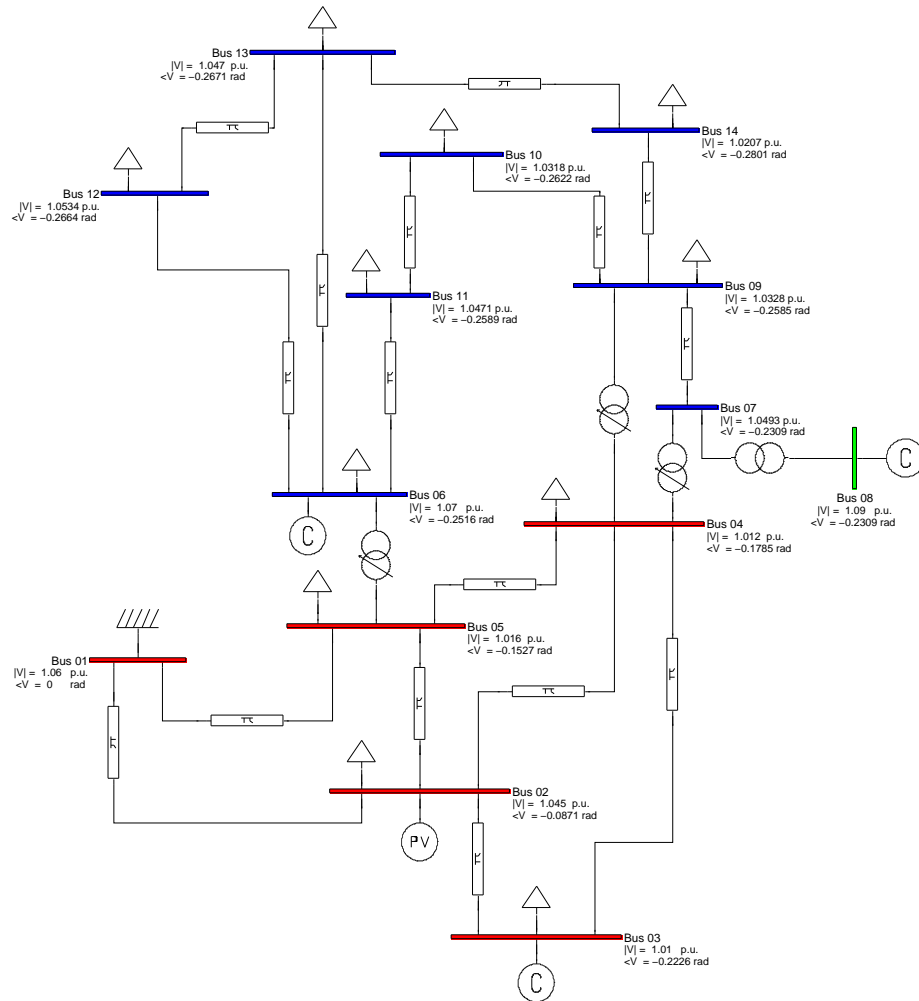


Figure 21.16: Simulink model of the IEEE 14-bus test system.

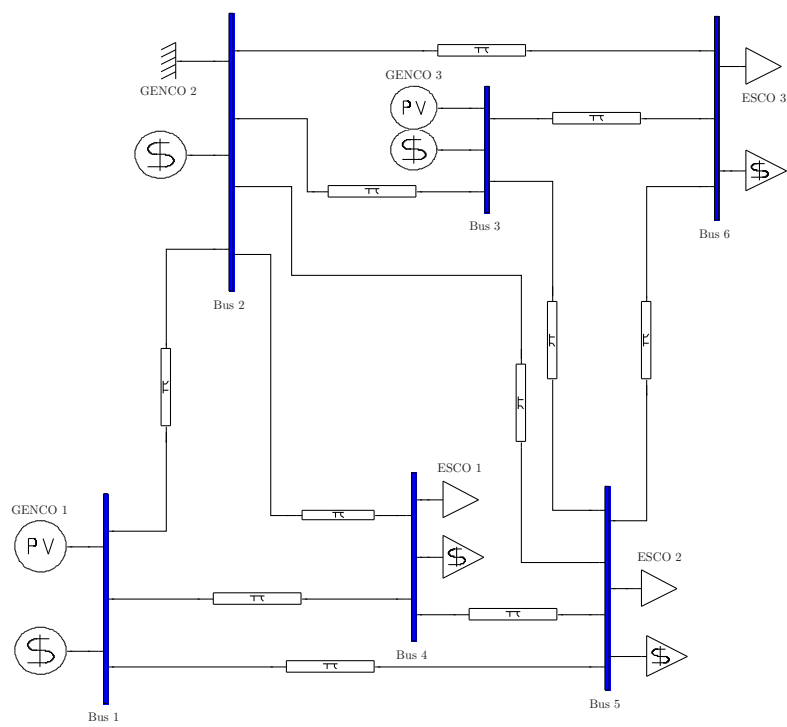


Figure 21.17: Simulink model of the 6-bus test system.

Chapter 22

Block Usage

This chapter describes how to use and connect blocks of the SIMULINK library provided with PSAT.

22.1 Block Connections

Generally speaking, 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 unpredictable 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).

As a general rule, PSAT should take care of all not allowed connections,¹ while the user should check for possible infeasible conditions. Following Sections 22.2 and 22.3 mostly 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 the following, blocks are subdivided in two main groups: *standard* and *non-standard*. Standard blocks must be connected only to buses, while nonstandard blocks can be connected to other blocks or may need another block at the same bus. Observe that well formed models must contain **only** blocks which are provided with the PSAT SIMULINK library.

¹There is still some work to do on this issue.

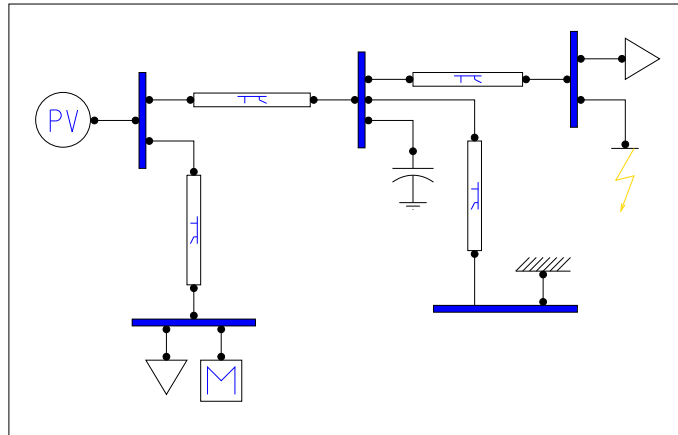


Figure 22.1: Examples of standard blocks of the PSAT SIMULINK Library.

22.2 Standard Blocks

Standard blocks only need to be connected to one bus for each input/output port. Blocks which do not follow this rule are described in the following Section 22.3. Some examples of standard blocks are depicted in Fig. 22.1.

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. 22.2 and 22.3). 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. 22.4). This kind of inconsistency cannot be easily checked automatically. A particular care should be devoted to avoid infeasible constraints.

²Note that the number of slack generators may be greater than one. This may occur if one defines two or more disconnected 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.

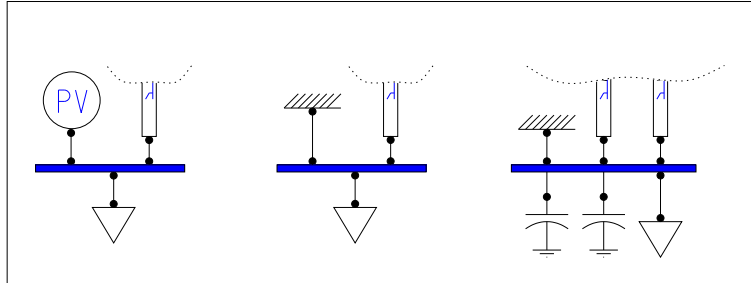


Figure 22.2: Examples of allowed connections of slack generators, PV generators and PQ loads.

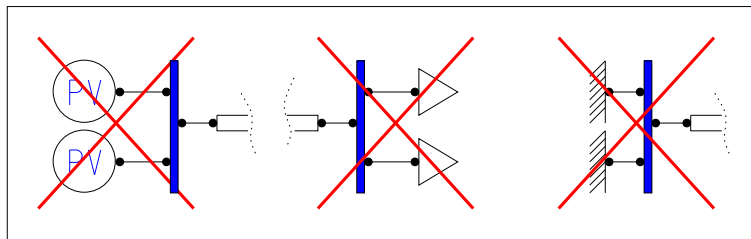


Figure 22.3: Not allowed connection of slack generators, PV generators and PQ loads.

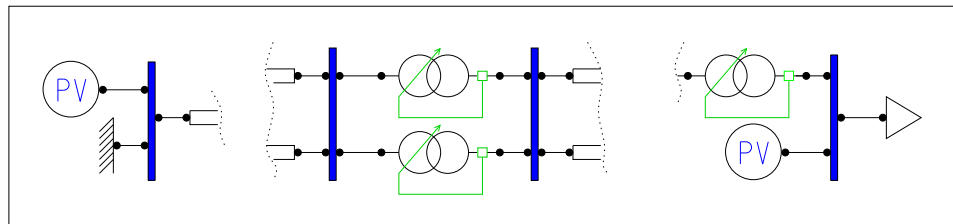


Figure 22.4: Infeasible or “likely” infeasible block connections.

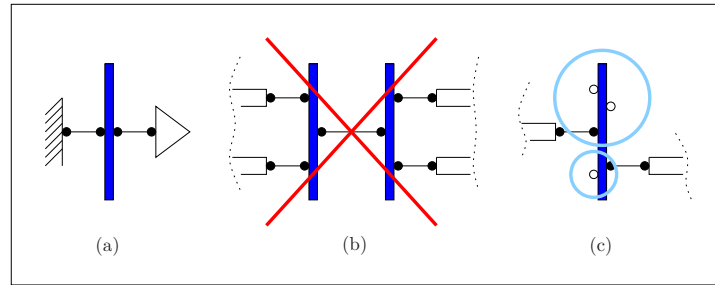


Figure 22.5: *Bus* block usage. (a) Minimal working network; (b) Not allowed bus connections; (c) Unused bus ports are allowed but not recommended.

22.3 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.

22.3.1 Buses

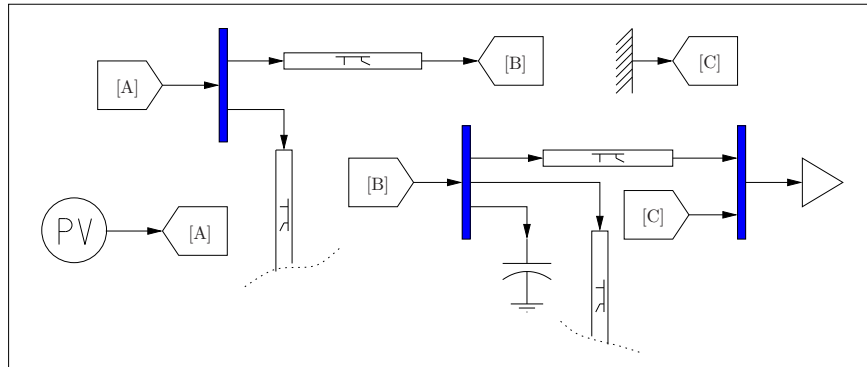
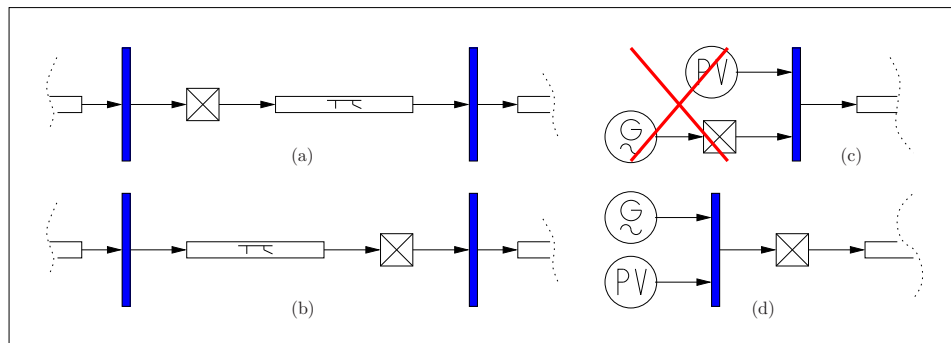
Bus blocks are the basic elements of each model. A PSAT network has to contain at least one bus. Observe that Bus blocks cannot be connected directly one to another. The number of input and output ports is variable. It is not mandatory to use all ports, but the habit of leaving unused bus ports is not 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 22.5 illustrates the bus block usage.

22.3.2 Goto and From Blocks

Goto and *From* blocks are inherited from the SIMULINK standard library and can connect any two blocks of the PSAT library. In practice they can be useful to draw neater schemes. Figure 22.6 illustrates the usage of *Goto* and *From* blocks.

22.3.3 Links

The *Link* block is a special kind of connection which is used **only** within a Secondary Voltage Regulation control system. See Section 22.3.11 for details.

Figure 22.6: *Goto* and *From* block usage.Figure 22.7: *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)** Correct usage of a breaker to disconnect a synchronous machine.

22.3.4 Breakers

Breaker blocks works only when connected to one line and one bus. The relative position with respect to the line does not matter. It is not allowed to use breaker to disconnect other components than lines; thus in order to simulate a switch for a generator, a load or any other component, one has to insert a new bus and a “dummy” line (low impedance). Future versions of PSAT could include switches which avoid including new buses and new lines. Figure 22.7 illustrates the usage of breaker blocks.

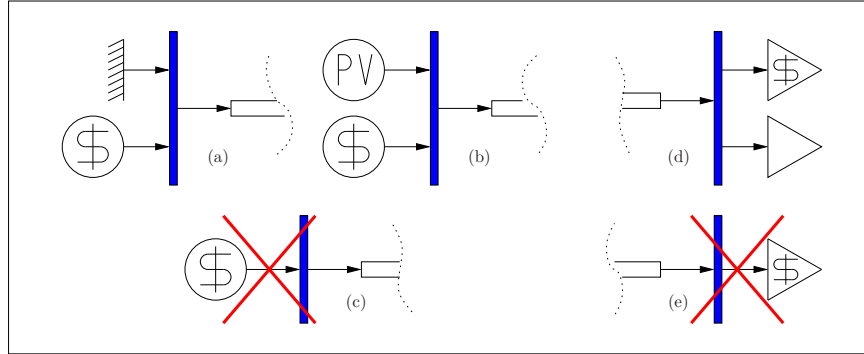


Figure 22.8: *Supply* and *Demand* block usage. (a) and (b) Correct usage of Supply blocks; (c) Incorrect usage of Supply blocks; (d) Correct usage of Demand blocks; (e) Incorrect usage of Demand blocks.

22.3.5 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 22.8 illustrates Supply and Demand block usage.

22.3.6 Generator Ramping

Generator Ramping 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 Ramping block; it is not recommended to leave unused input ports in Supply blocks. Figure 22.9 illustrates the Ramping block usage. Observe that Generator Ramping data only have effects when used with the PSAT-GAMS interface (multi-period and unit commitment methods).

22.3.7 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 22.10 illustrates the Reserve block usage. Observe that Generator Reserve data only have effects when used with the PSAT OPF routine.

22.3.8 Non-conventional Loads

Non-conventional load blocks are those described in Chapter 14, 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

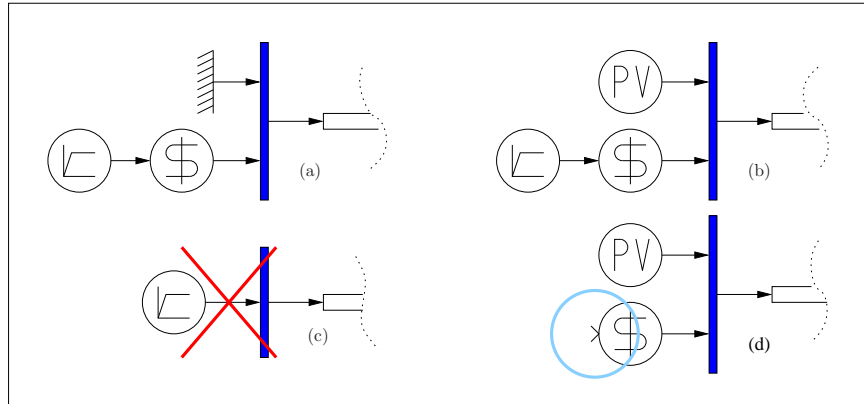


Figure 22.9: *Generator Ramping* block usage. (a) and (b) Correct usage of Ramping blocks; (c) Incorrect usage of Ramping blocks; (d) Not recommended usage of Supply blocks.

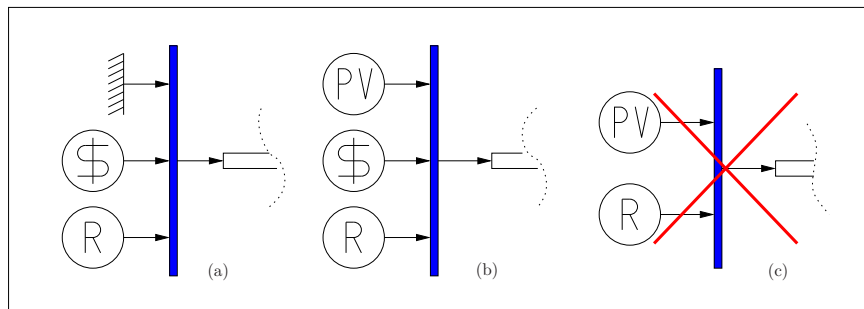


Figure 22.10: *Generator Reserve* block usage. (a) and (b) Correct usage of Reserve blocks; (c) Incorrect usage of Reserve blocks.

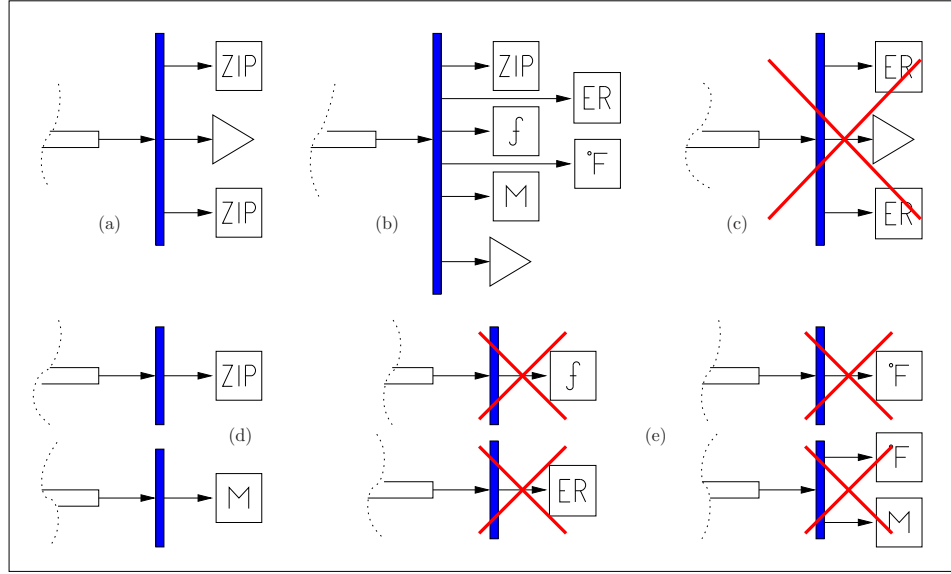


Figure 22.11: *Non-conventional Load* block usage. (a) and (b) Correct usage of non-conventional load blocks; (c) Incorrect usage of exponential recovery load. (d) Correct usage of voltage dependent and ZIP load blocks when the “Initialize after power flow” parameter is set to 0. (e) Incorrect usage of non-conventional load blocks.

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 the same bus. It is allowed to connect multiple non-conventional loads at the same bus, however, observe that it does not make sense to connect two exponential recovery loads at the same bus. See Section 14.8 for a few remarks on the usage of non-conventional loads. Figure 22.11 illustrates non-conventional loads usage within SIMULINK models.

22.3.9 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 zero to two input ports. The input ports are needed only when using regulators; it is not recommended to leave unused input ports in Synchronous Machine blocks. Observe that 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

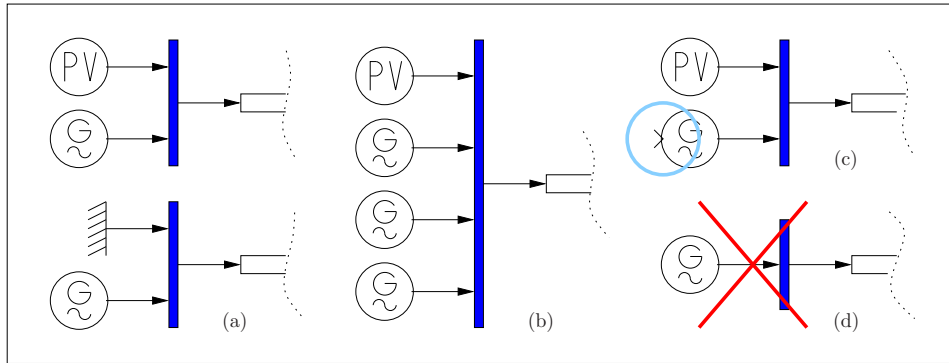


Figure 22.12: *Synchronous Machine* block usage. (a) and (b) Correct usage of synchronous machine blocks; (c) Not recommended usage of synchronous machine blocks. (d) Incorrect usage of synchronous machine blocks.

computing the Synchronous machine power injections are those of PV or slack bus generators. Figure 22.12 illustrates the Synchronous Machine block usage.

22.3.10 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 zero to three input ports. The input ports are needed only when using PSSs, OXLs, or Secondary Voltage Regulator blocks; it is not recommended to leave unused input ports in AVR blocks. Only one kind of regulator is allowed for each machine. Figure 22.13 illustrates the usage of the primary regulator blocks.

22.3.11 Secondary Voltage Regulation

Secondary Voltage Regulation (SVR) blocks are the Central Area Controller (CAC) and the Cluster Controller (CC) blocks. It is not allowed to use CAC or CC blocks alone. 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 22.14 illustrates the usage of CAC, CC

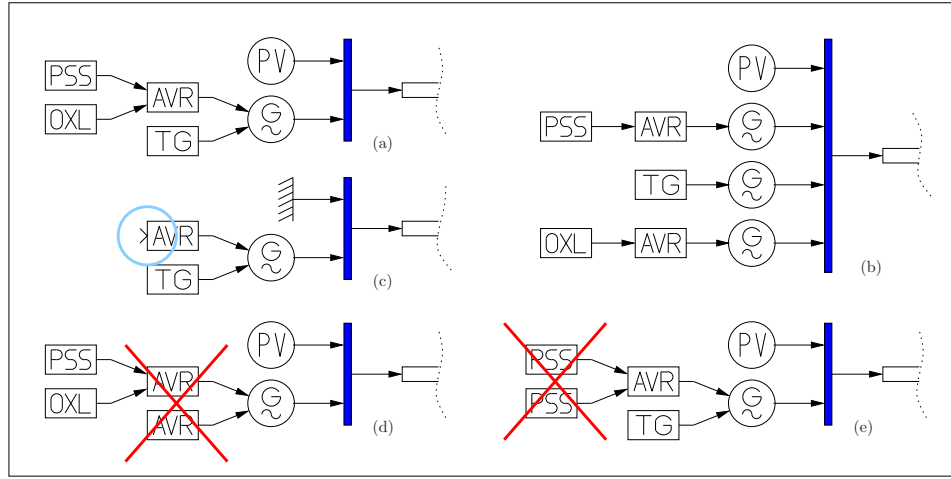


Figure 22.13: *Primary Regulator* block usage. (a) and (b) Correct usage of regulator blocks; (c) Not recommended usage of automatic voltage regulator blocks; (d) and (e) Incorrect usage of regulator blocks.

and Link blocks. Any other usage of these blocks is not allowed and would lead to unpredictable results or to error messages.

22.3.12 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 22.15 illustrates the usage of ULTC blocks. Observe that the ULTC is a standard block when using control types 1 and 2. In the case of control type 3, it is a nonstandard one since the remote bus provides a signal, not a topological connection.

22.3.13 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 22.16 illustrates the usage of SVC and STATCOM blocks.

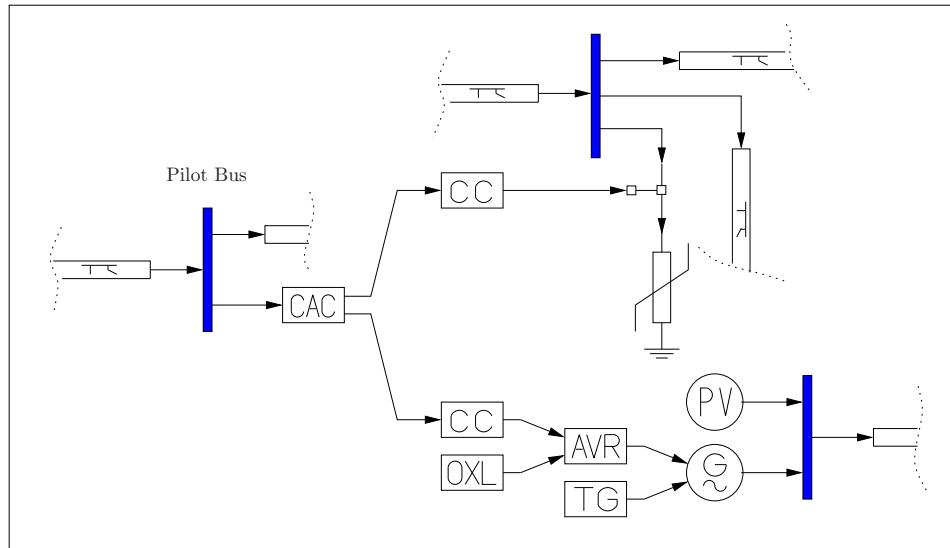


Figure 22.14: *Secondary Voltage Regulation* block usage.

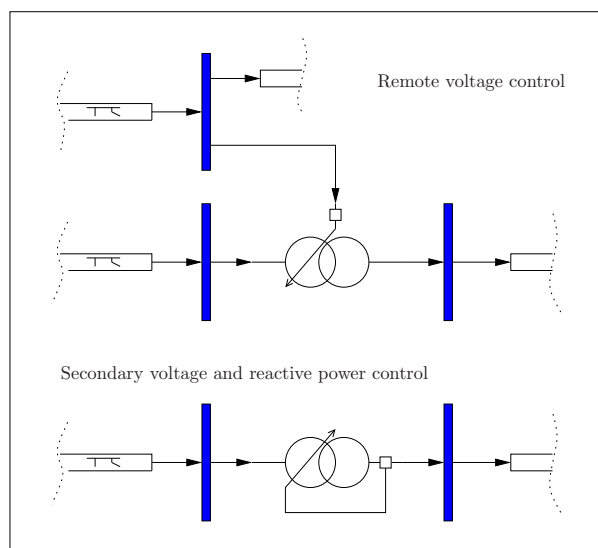


Figure 22.15: *Under Load Tap Changer* block usage.

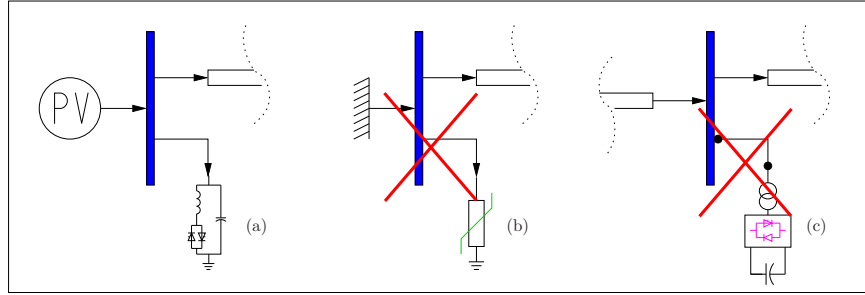


Figure 22.16: *SVC* block usage. (a) Correct usage of *SVC* blocks; (b) and (c) Incorrect usage of *SVC* blocks.

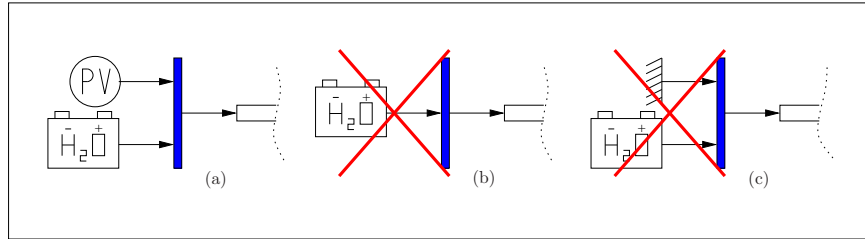


Figure 22.17: *Solid Oxide Fuel Cell* block usage. (a) Correct usage of Fuel Cell blocks; (b) and (c) Incorrect usage of Fuel Cell blocks.

22.3.14 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 22.17 illustrates the fuel cell block usage.

22.3.15 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. 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 22.18 illustrates the dynamic shaft block usage.

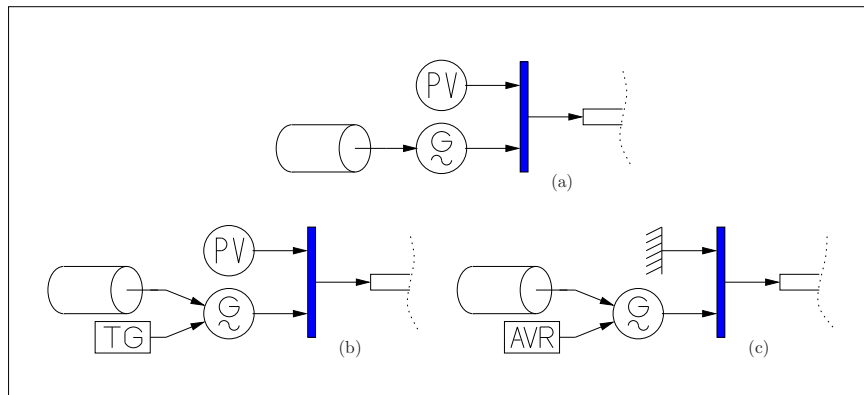


Figure 22.18: *Dynamic Shaft* block usage. **(a-c)** Correct usages of Dynamic Shaft blocks.

Chapter 23

Block Masks

As already mentioned in Chapter 21, 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 associated with blocks of the PSAT-SIMULINK library work and how to create a new mask for a custom block.

23.1 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 component 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. 23.1);
2. In theory, any other CAD tool could be used for drawing PSAT models, given a filter able to translate data in a format readable by PSAT.

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.

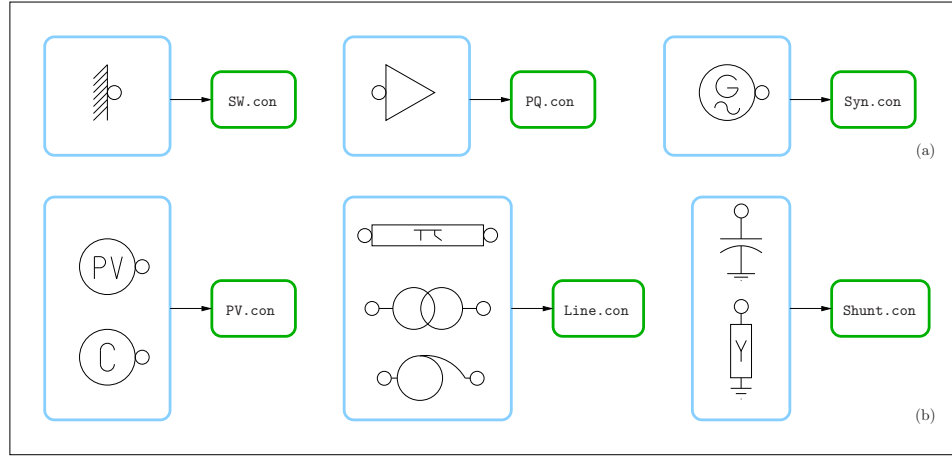


Figure 23.1: Correspondence between SIMULINK blocks and PSAT global structures. (a) examples of structures associated with only one SIMULINK block; (b) examples of structures associated with more than one SIMULINK block.

23.2 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. 23.2). 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 23.4 in this chapter.

23.2.1 Mask Initialization

Each SIMULINK block mask must be initialized, i.e. 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 is

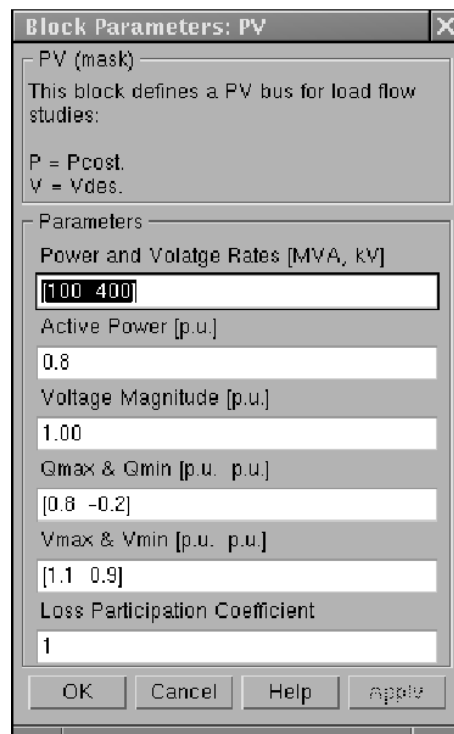


Figure 23.2: Mask GUI of a PSAT-SIMULINK block.

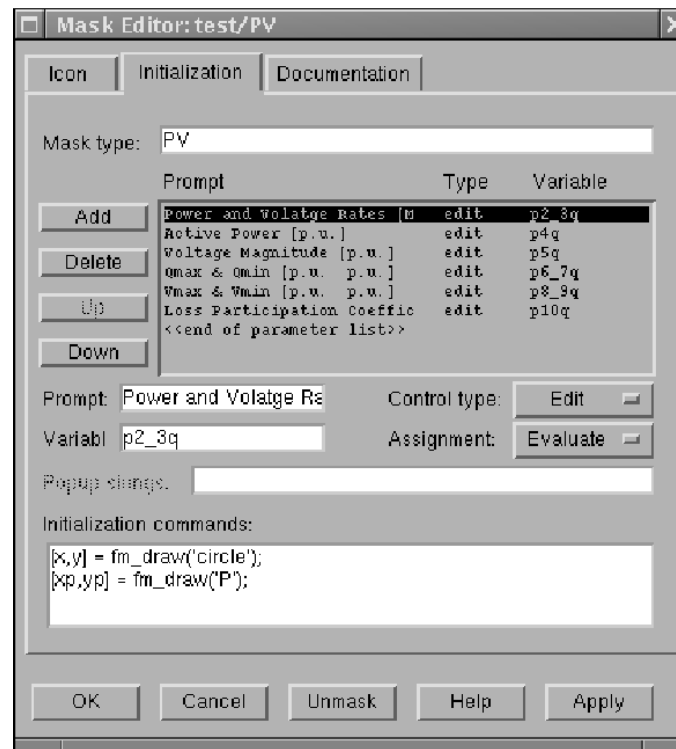


Figure 23.3: Mask initialization GUI for a PSAT-SIMULINK block.

depicted in Fig. 23.3.

Each PSAT block uses the initialization GUI to define block parameters. Parameter names have a special syntax, which is fully described in Section 23.3.

23.2.2 Mask Icon

Mask icons are defined in the edit mask GUI as well. An example of block icon is depicted in Fig. 23.4. 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. 23.3).

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.

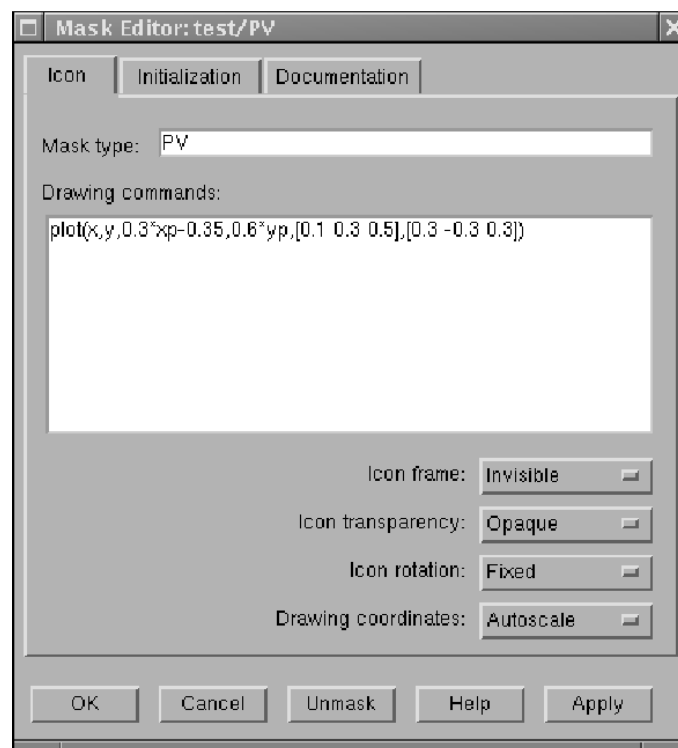


Figure 23.4: Mask icon GUI of a PSAT-SIMULINK block.

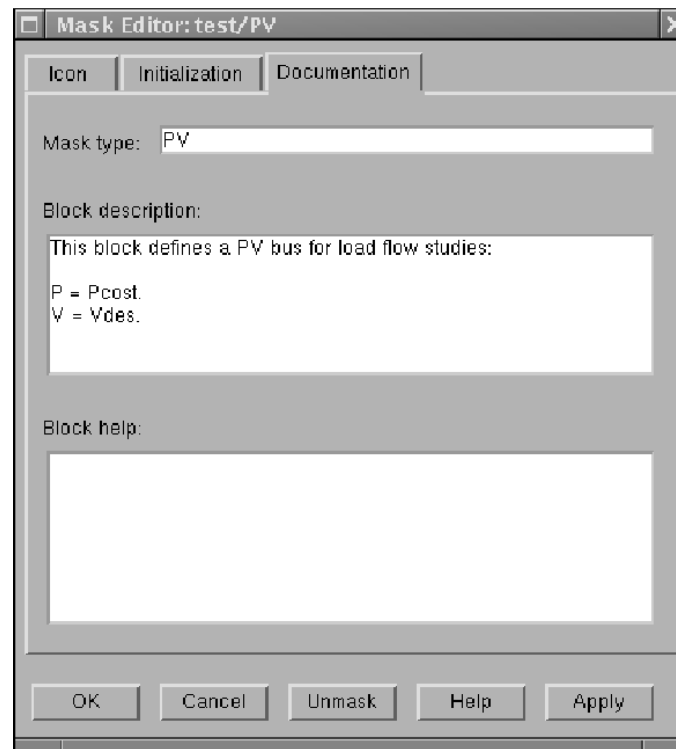


Figure 23.5: Mask documentation GUI of a PSAT-SIMULINK block.

23.2.3 Mask Documentation

Each PSAT block comes with a brief documentation. An example of block documentation is depicted in Fig. 23.5.

23.3 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 23.1 whereas a few

Table 23.1: Mask parameter symbols

Pseudo Symbol	Meaning	Matlab Symbol
p	open array	[
q	close array]
- ²	comma	,
x	list	:

Table 23.2: 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]

examples of well formed mask variable names are depicted in Table 23.2.

There are also some keywords, which are associated with constant values, as depicted in Table 23.3. 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 23.1.

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 23.2

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.

23.4 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:

²That is underscore, not dash.

Table 23.3: 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	<i>none</i>
friday	5	out	<i>none</i>

1. Each PSAT block is a hollow subsystem. Thus in order to add or remove input/output ports, all it needs is opening the underneath subsystem model and adding or removing input/output blocks (see Fig. 23.6.a). If the block has an odd number of input/output ports add a sink or a source block (see Fig. 23.6.b). Block connections are needed to avoid SIMULINK error messages.
2. The *Mask Type* property has always to be defined and has to be an existing PSAT structure.
3. Mask variables must follow the conventional syntax defined in the previous Section 23.3.
4. 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.

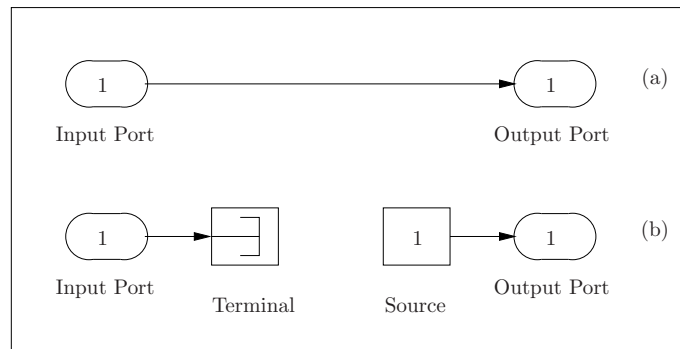
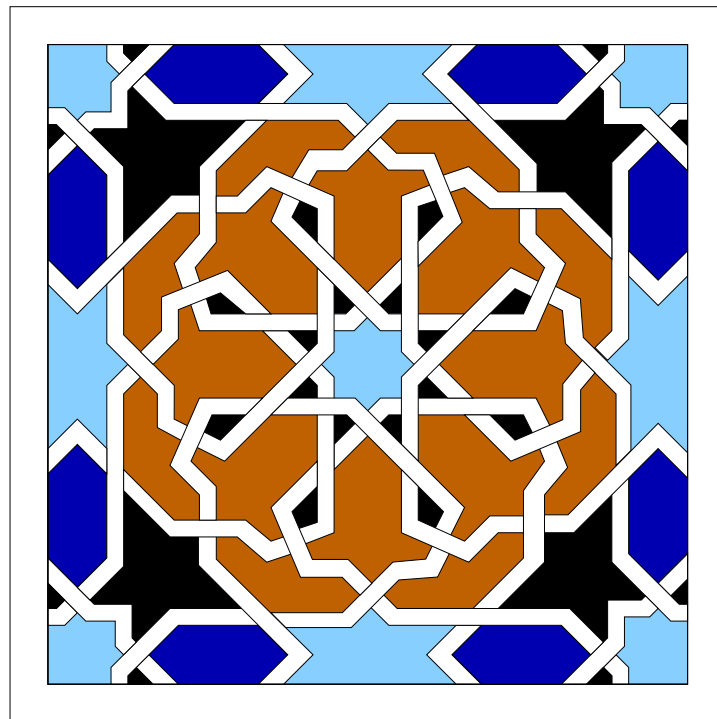


Figure 23.6: SIMULINK model underneath a mask of a PSAT block. **(a)** Usage of input and output ports connected by a line; **(b)** usage of sinks and terminal blocks.

Part V

Tools



Chapter 24

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 WECC and EPRI ETMSP format.²

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).

Observe that the conversions 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 24.1 depicts the this GUI.

The following filters have been implemented so far:³

`chapman2psat`: conversion from Chapman's data format [30];

`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;

`eurostag2psat`: conversion from Eurostag data format;

`flowdemo2psat`: conversion from FlowDemo.net data format;

`ge2psat`: conversion from General Electric data format;

¹Most 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 found in [126]. 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`.

`ieee2psat`: conversion from IEEE common data format;
`inptc12psat`: conversion from CESI INPTC1 data format;
`matpower2psat.m`: conversion from MATPOWER data format;
`neplan2psat`: conversion from NEPLAN data format;⁴
`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;
`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;
`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.

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.

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/

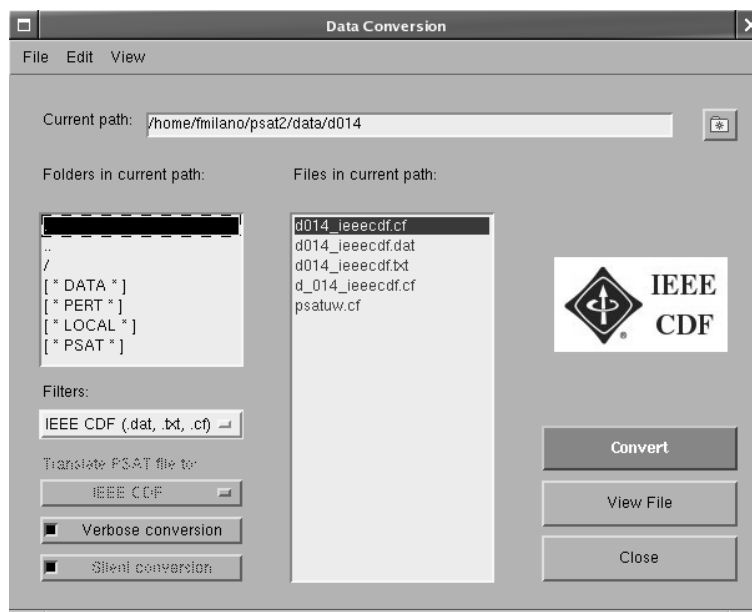


Figure 24.1: GUI for data format conversion.

-a : define additional file. 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 25

User Defined Models

This chapter describes routines and GUIs for creating and installing User Defined Models (UDMs) in PSAT. The routine which compiles UDM functions is not complete yet and several limitations apply. However it can be used for creating templates and to easily install/remove custom components to and from PSAT.

25.1 Installing and Removing Models

Figure 25.1 depicts the browser of the UDM archive. UDM files are placed in the `build` folder within the PSAT main folder. The browser allows to install and uninstall UDMs and to compile the functions which describe the UDMs. After installing a UDM, PSAT should be restarted to work properly. Table 25.1 depicts all functions and files which are modified in order to install a new component. When the component is installed the first time, a GUI will display a list of these files and allow the user to inspect changes.

Table 25.1: Functions and files to be modified for installing a UDM

MATLAB functions	Other files
Contents.m	comp.ini
closepsat.m	namevarx.ini
fm_dynidx.m	namevary.ini
fm_dynlf.m	service.ini
fm_inilf.m	
fm_ncomp.m	
fm_var.m	
fm_xfirst.m	
psat.m	



Figure 25.1: Browser of user defined models.

25.2 Creating a User Defined Model

To create a new UDM, use the button “New Component” of the browser depicted in Fig. 25.1. The GUI depicted in Fig. 25.2 will appear.

Using this interface should be quite intuitive. General settings of the component can be defined using the interface depicted in Fig. 25.3, which can be launched from the button “Setting” of the toolbar on top of the main UDM window GUI. In order to add variables and equations simply type them in the respective edit texts and then use the buttons “Add” for updating the model. Variable names and equations must be valid MATLAB variables and expressions. State variables and parameters settings are handled by the masks depicted in Figs. 25.4 and 25.5. The complete list of properties that can be defined is reported in Appendix A.

When the model definition is completed, the model can be saved and compiled.¹ If the routine encounters inconsistencies during the compilation process, it will display a report of errors, which should simplify the debugging process.

Brief descriptions of the GUIs for setting parameters and variables follow.

25.2.1 Component Settings

Component properties can be viewed and changed by means of the GUI depicted in Fig. 25.3. Properties are as follows:

Name: the name of the component which is used as the name of the “build” file for defining the component itself, as the resulting function name to be included in PSAT, and as the structure name of the component. For example if the component name is `testudm`, one has:

<code>testudm.m</code>	component “build” file
<code>fm.testudm.m</code>	component function
<code>Testudm</code>	component global structure

Description: brief component description which will be included in the help of the component function and in the file `Contents.m`. The description should be one line long.

Initialization: this field tells the compiler if the component has to be included in the power flow or must be initialized when the power flow is completed. Select the checkbox to initialize after the power flow computation.

Shunt: this field is needed at the installation time and tells the installer if the component should be treated as a shunt or not. In the latter case, the component is not considered a load and its power absorption will not be included in the vectors `Bus.P1` and `Bus.Q1`.

¹The Symbolic Toolbox is required for compiling UDM functions.

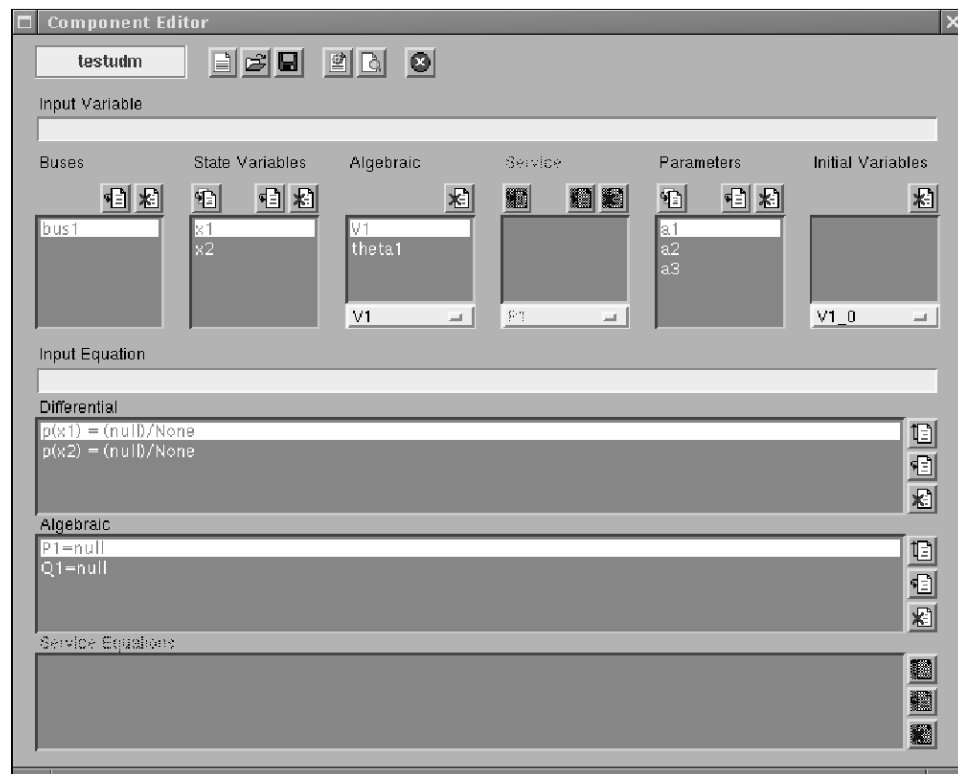


Figure 25.2: GUI for creating user defined models.

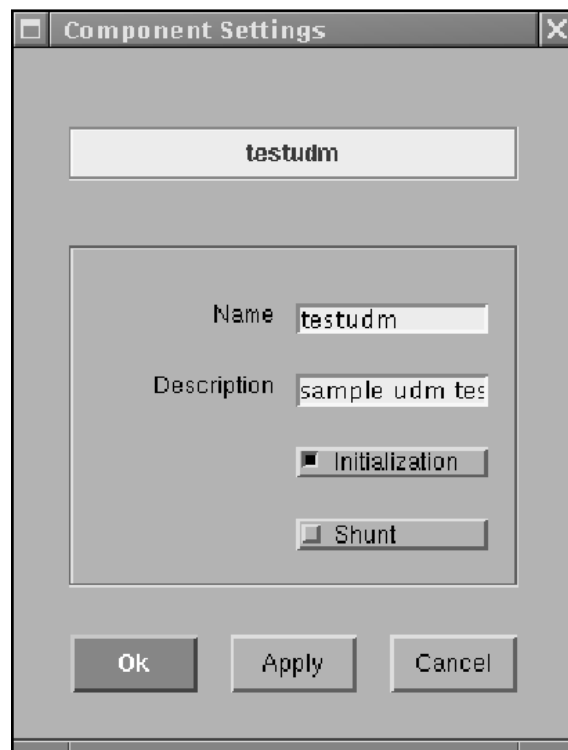


Figure 25.3: GUI for setting component properties.

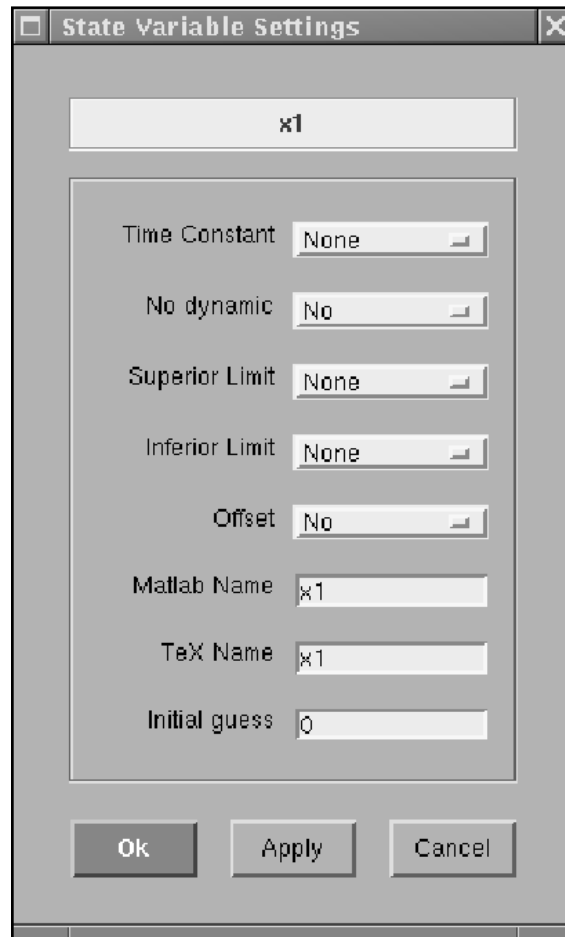


Figure 25.4: GUI for setting state variable properties.



Figure 25.5: GUI for setting parameters properties.

25.2.2 State Variable Settings

State variable properties can be viewed and changed by means of the GUI depicted in Fig. 25.4. Properties are as follows:

Time Constant: time constant associated with the state variable differential equation. The user can chose among all time constants defined in the parameter lists. If *none* is selected, no time constant will be used for the differential equation.

No Dynamic: if “yes” is selected the time constant is allowed to be zero, which corresponds to set an algebraic equation.

Superior Limit: if a value is selected other than *none*, the state variable undergoes an anti-windup limit for its maximum value.

Inferior Limit: if a value is selected other than *none*, the state variable undergoes an anti-windup limit for its minimum value.

Offset: if “yes” is selected the state variable initial value is set to zero and an offset value is used for setting a zero first derivative at the end of the initialization step.

Matlab Name: name of the variable which is used as a field of the component structure. Must be a well formed Matlab variable.

TeX Name: \TeX formatted name which is used in the legend when plotting the state variable. Must be a well formed \TeX math expression.

Initial Guess: value to be used for components which are initialized after power flow solutions.

25.2.3 Parameter Settings

Parameter properties can be viewed and changed by means of the GUI depicted in Fig. 25.5. Properties are as follows:

Unit: unit of the parameter. This field is used only in the “on-line” help of the component function.

Type: type of the parameter. This field is generally used only in the “on-line” help of the component function. However, when *Time Constant* is selected, the parameter will be included in the list of time constants for differential equations.

Description: brief parameter description. This field is used only in the “on-line” help of the component function. The description should be one line long.

25.3 Limitations

The routine that compiles UDM functions is currently incomplete, and several limitations applies, e.g.:

1. Only models which are connected to at least one bus can be defined;
2. Algebraic equations should be always defined;
3. UDMs cannot share state variables with other models;
4. Bus voltages are the only allowed algebraic variables;
5. All defined variables should be used either in the algebraic or in the differential equations.

Chapter 26

Utilities

This chapter describes the GUI and the properties of the PSAT command history, the GUI for sparse matrix visualization, and theme and text viewer settings.

26.1 Command History

The command history of all operations performed in PSAT is contained in the structure `History`, which can be displayed and saved by means of the GUI depicted in Fig. 26.1. The GUI can be launched using the *Options/History* menu in the main window.

26.2 Sparse Matrix Visualization

Figure 26.2 depicts the GUI for sparse matrix visualization, available in the *View/Sparse Matrix Visualization* menu in the main window.¹

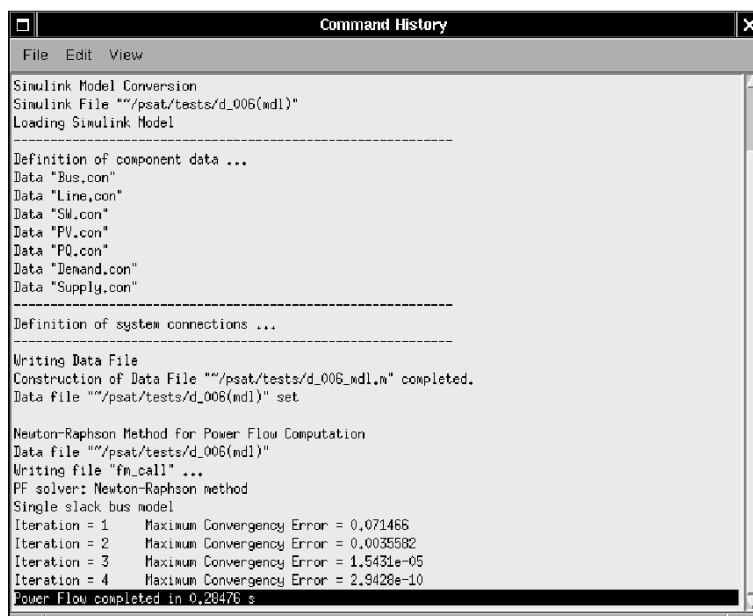
26.3 Themes

The graphical appearance of PSAT GUIs can be changed using the theme browser GUI, available in the *Option/Themes* in the main window. Figure 26.3 depicts this GUI displaying a preview of a theme provided with the toolbox. For adding new themes, just open and modify one of the sample files contained in the sub-folder `themes` within the main PSAT folder.

26.4 Text Viewer

Figure 26.4 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

¹The figure illustrates the complete power flow Jacobian matrix of a 1228-bus test system.



```

Command History
File Edit View
Simulink Model Conversion
Simulink File ""/psat/tests/d_006.mdl"
Loading Simulink Model
-----
Definition of component data ...
Data "Bus.con"
Data "Line.con"
Data "SW.con"
Data "PV.con"
Data "PQ.con"
Data "Demand.con"
Data "Supply.con"
-----
Definition of system connections ...
-----
Writing Data File
Construction of Data File ""/psat/tests/d_006.mdl.m" completed.
Data file ""/psat/tests/d_006.mdl" set
-----
Newton-Raphson Method for Power Flow Computation
Data file ""/psat/tests/d_006.mdl"
Writing file "fm_call" ...
PF solver: Newton-Raphson method
Single slack bus model
Iteration = 1      Maximum Convergency Error = 0.071466
Iteration = 2      Maximum Convergency Error = 0.0035582
Iteration = 3      Maximum Convergency Error = 1.5431e-05
Iteration = 4      Maximum Convergency Error = 2.9428e-10
Power Flow completed in 0.28476 s

```

Figure 26.1: Command history GUI.

operating system, i.e. Unix (Solaris), Linux and Windows. Power flow results can be saved in three different formats, i.e. Microsoft Excel², 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.

26.5 Building p-code Archive

MATLAB allows pre-compiling (p-code) the plain text *m*-files. Pre-compiled files run faster on some platforms, although they cannot be modified as are in binary format. Figure 26.5 depicts the GUI for creating a p-code archive. The GUI can be launched from the menu *Tools/p-code archive* in the main window. Before running the utility, be sure you have writable permission in the PSAT folder.

²ActiveX[®] is used for exporting results to Microsoft Excel.

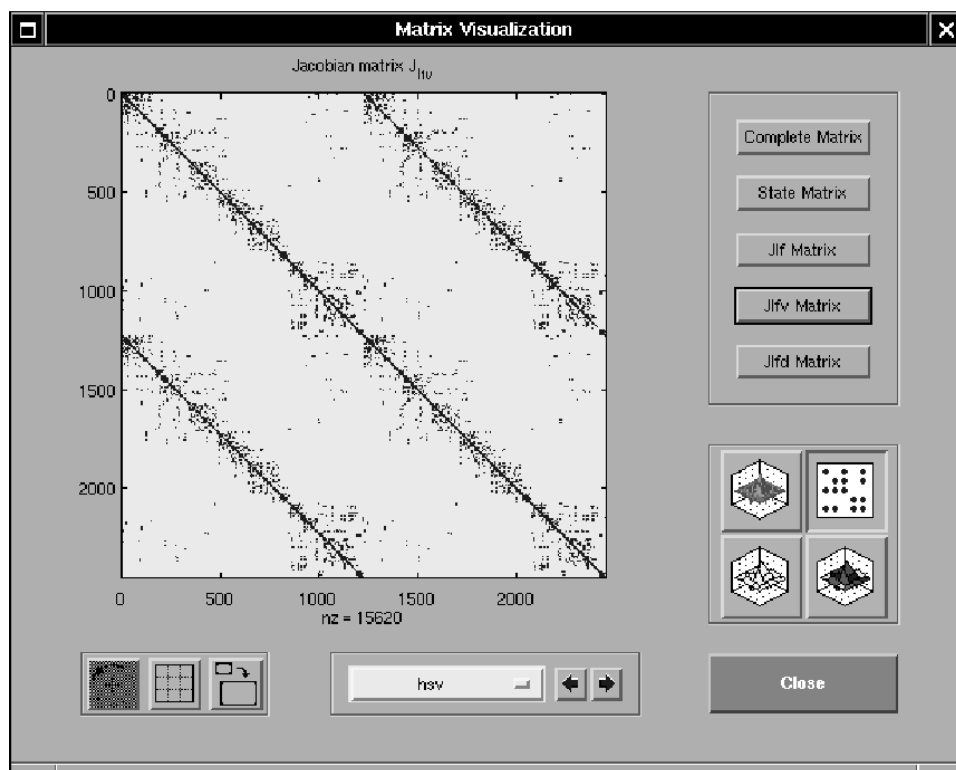


Figure 26.2: GUI for sparse matrix visualization.

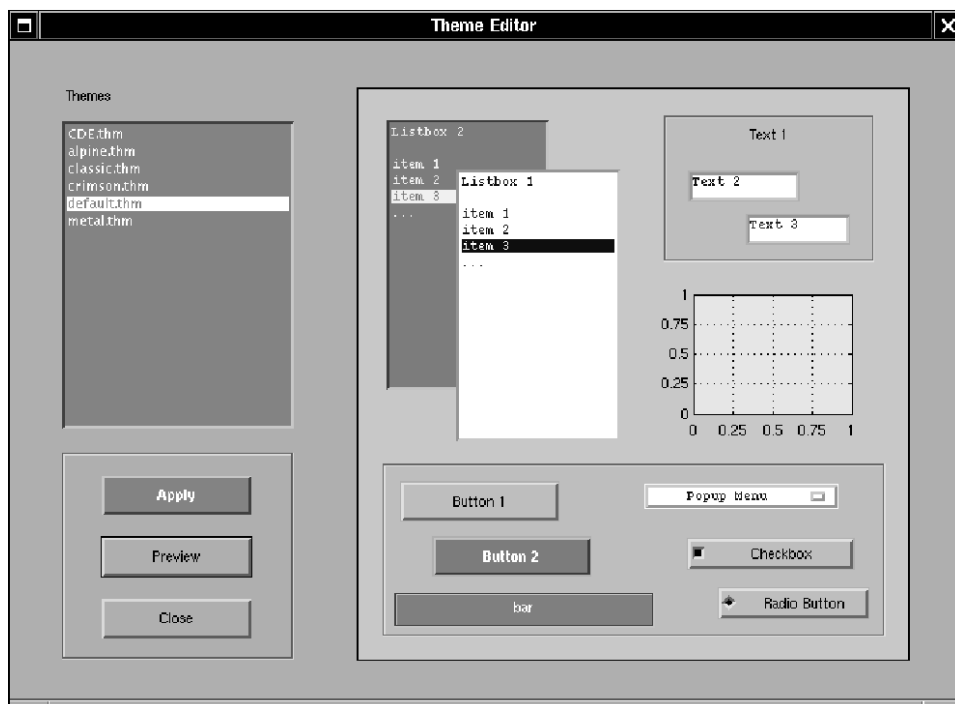


Figure 26.3: GUI for PSAT theme selection.

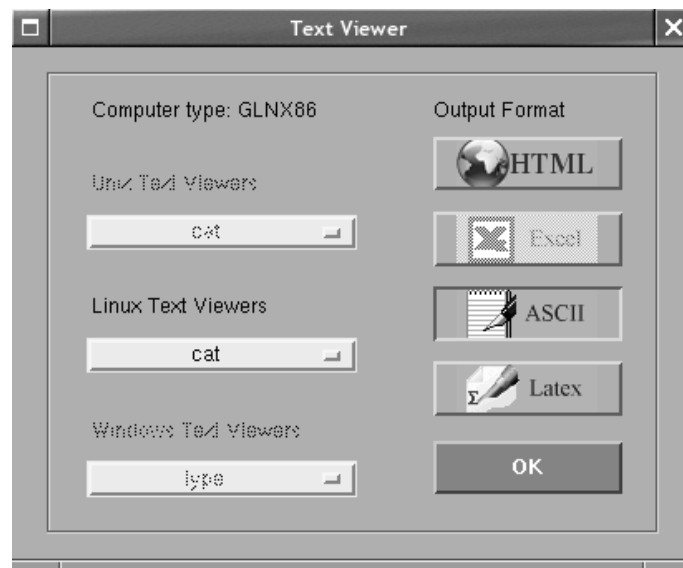


Figure 26.4: GUI for text viewer selection.



Figure 26.5: GUI for p-code archive builder.

Chapter 27

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 speeds up operations.

27.1 Basics

Firstly, one needs to set up PSAT environment. Launching the script file `initpsat`, as follows:

```
>> initpsat
```

will initialize PSAT and display on the MATLAB workspace:

```
< P S A T >
Copyright (C) 2002-2004 Federico Milano
Version 1.3.2
November 2, 2004
```

```
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 7.0.0.19901 (R14)
Session:       02-Nov-2004 17:30:23
Usage:         Command Line
Path:          /home/fmilano/psatd
```

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 27.1: 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
cpfatc	evaluate ATC using CPF analysis
sensatc	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 27.1. 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 27.2. 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 27.2: General Options for Command Line Usage.

String	Associated routine
<code>data</code>	set data file
<code>pert</code>	set perturbation file
<code>opensys</code>	open solved case
<code>savesys</code>	save current system
<code>log</code>	write log file of the current session
<code>pfrep</code>	write current power flow solution
<code>eigrep</code>	write eigenvalue report file
<code>pmurep</code>	write PMU placement report file

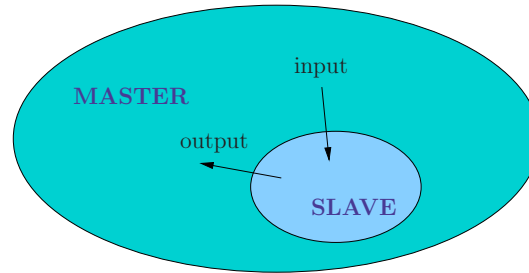


Figure 27.1: Master-slave architecture.

27.2 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 27.1). 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 continu-

Table 27.3: 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

ation curves or the Pareto 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. Observe that, 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 27.3 depicts PSAT routines and the associated global structures which define routine options. A full description of these structures is presented in Appendix A.

27.3 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.

27.4 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 28

Running 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 [44]. GNU Octave is also freely redistributable software. You may redistribute it and/or modify it under the terms of the GNU General Public License (GPL) as published by the Free Software Foundation.

PSAT 1.3.4 can be run on GNU Octave. PSAT has been tested with Octave version 2.1.72 and on a few older distributions² for Linux and the `octave-forge` package (dated 2005.06.13).³ The following restrictions apply:

1. PSAT version 2.0.0 **cannot** run on Octave;
2. only the command line usage of PSAT is allowed;
3. sparse matrix methods are not used;
4. there is no support for SIMULINK models;
5. only full eigenvalue analysis can be performed;
6. there is no plotting utility support;
7. PSAT cannot run on the stand-alone Windows release of Octave. I guess it is just a matter of waiting for the next Octave release for Windows (current version is `octave-2.1.50-windows`). There should not be any problem if using Windows, CygWin and the last Linux version of Octave.

¹GNU Octave is available at www.octave.org.

²PSAT should work properly also on Octave versions back to 2.1.53.

³Octave-forge is available at <http://octave.sourceforge.net/>.

28.1 Basic Commands

All commands provided by the command line usage (see Chapter 27) work well on GNU Octave. However observe that, on Octave, the syntax

```
>> runpsat command
```

is not allowed and one of the following functional forms

```
>> runpsat('command')
```

```
>> runpsat("command")
```

must be used. Furthermore, on 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) 2002-2004 Federico Milano
      Version 1.3.2
      November 2, 2004
```

```
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 2.1.53
Session:        02-Nov-2004 15:49:48
Usage:          Command Line
Path:           /home/fmilano/psat
```

28.2 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] Voltages
[ 3] Active Powers
[ 4] Reactive Powers
[ 5] Generator speeds
[ 6] Generator angles
```

pick a number, any number:

Figure 28.1 depicts an example of plot obtained using GNU/Octave and `gplot`. The graphs refers to the generator speeds of the 9-bus example described in Chapter 8 (see Fig. 8.6).

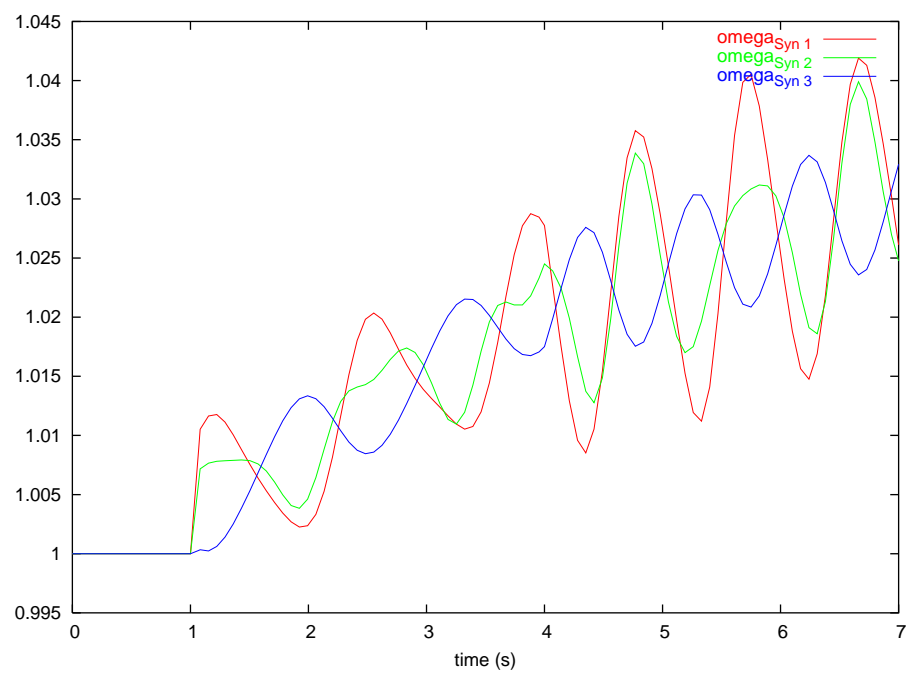


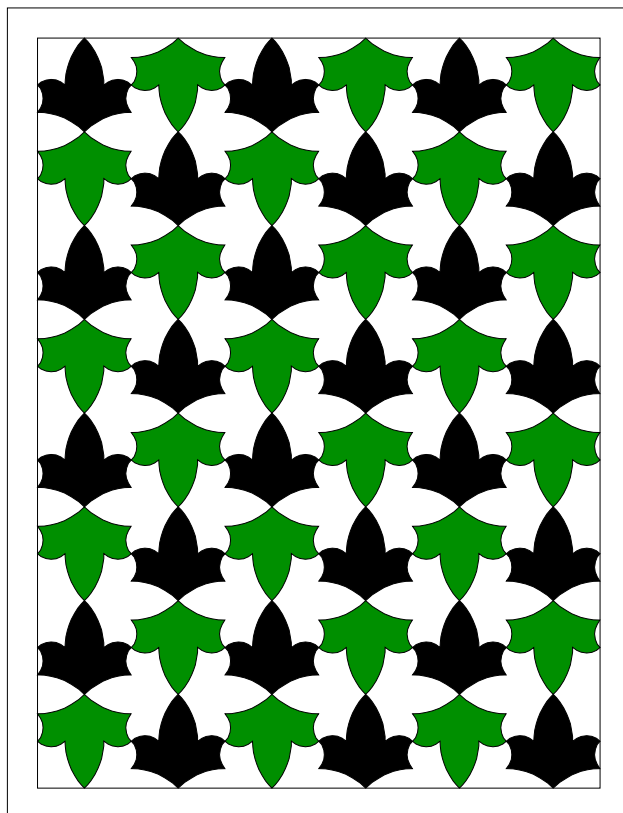
Figure 28.1: Example of graph obtained using GNU/Octave and gplot.

28.3 ToDos

The usage of PSAT on Octave is currently in an early stage of development. Any help, bug squash and contribution is very welcome. PSAT can be “free software” only if it is fully compatible with Octave.

Part VI

Interfaces



Chapter 29

GAMS Interface

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]. 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.

29.1 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 which 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*).

29.2 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 [43]) 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 [87]), 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 always led to reasonable results.

29.3 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 startup 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 29.1 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 29.2 depicts the PSAT-GAMS interface main window.

29.4 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 weighing factor ω in order to produce a Pareto set. Refer to [75] 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.

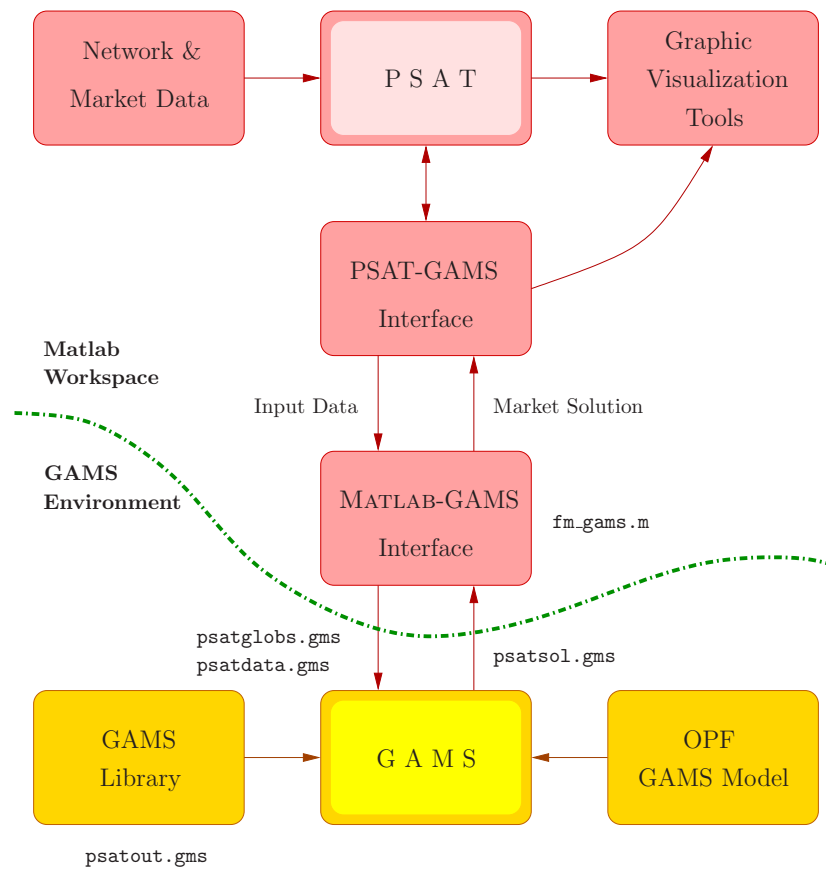


Figure 29.1: Structure of the PSAT-GAMS interface.

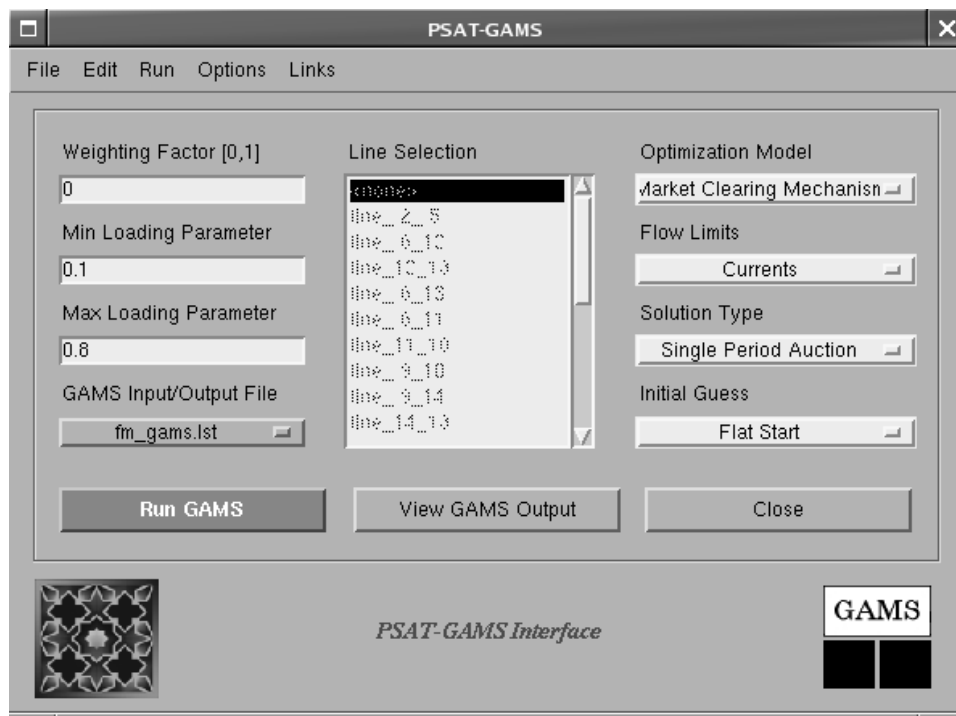


Figure 29.2: GUI of the PSAT-GAMS interface.

29.5 Multiperiod Market Clearing Model

29.5.1 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 [40, 86, 39].

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 ;

29.5.2 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 [40, 86].

The PSAT-GAMS interface includes ramping constraints as those that were described in [40], where the authors presents a detailed model of a multi-period auction for pool-based electricity markets. Model presented in [40] is linear, and take into account congestions 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:

$$\sum_{i \in \mathcal{I}} P_{S_i}(t) - \sum_{j \in \mathcal{J}} P_{D_j}(t) = 0 \quad \forall t \in \mathcal{T} \quad (29.1)$$

Market clearing model:

$$\sum_{i \in \mathcal{I}_b} P_{S_i}(t) - \sum_{j \in \mathcal{J}_b} P_{D_j}(t) - P_{m,n}(t) = 0 \quad \forall b \in \mathcal{B}, \quad \forall t \in \mathcal{T}, \quad (29.2)$$

$$\forall m, n \in \mathcal{N}, \quad \forall t \in \mathcal{T} \quad (29.3)$$

$$P_{m,n}(t) = \frac{S_b}{x_{mn}}(\theta_m - \theta_n) \quad \forall m, n \in \mathcal{N}, \quad \forall t \in \mathcal{T} \quad (29.4)$$

$$-P_{m,n}^{\max} \leq P_{m,n}(t) \leq P_{m,n}^{\max} \quad \forall m, n \in \mathcal{N}, \quad \forall t \in \mathcal{T} \quad (29.5)$$

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:

$$\begin{aligned} \text{Max. } G = & \sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{J}} C_{Dj} P_{Dj}(t) \\ & - \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{I}} (C_{Si} P_{Si}(t) + C_{SU_i} w_i(t) + C_{SD_i} z_i(t)) \end{aligned} \quad (29.6)$$

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 :

$$P_{Si}^{\min} u_i(t) \leq P_{Si}(t) \leq \bar{P}_{Si}(t) \quad \forall i \in \mathcal{I}, \quad \forall t \in \mathcal{T} \quad (29.7)$$

$$Q_{Gi}^{\min} u_i(t) \leq Q_{Gi}(t) \leq Q_{Gi}^{\max} u_i(t) \quad \forall i \in \mathcal{I}, \quad \forall t \in \mathcal{T} \quad (29.8)$$

where maximum available power output limits $\bar{P}_{Si}(t)$ 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:

$$\bar{P}_{Si}(t) \leq P_{Si}^{\max} [u_i(t) - z_i(t+1)] + z_i(t+1) SD_i \quad \forall i \in \mathcal{I}, \quad \forall t \in \mathcal{T} \quad (29.9)$$

$$\bar{P}_{Si}(t) \leq P_{Si}(t-1) + RU_i u_i(t-1) + SU_i w_i(t) \quad \forall i \in \mathcal{I}, \quad \forall t \in \mathcal{T}$$

The ramp-down rate limit and the shut-down ramp rate limit are modeled as follows:

$$P_{Si}(t-1) \leq P_{Si}(t) + RD_i u_i(t) + SD_i z_i(t) \quad \forall i \in \mathcal{I}, \quad \forall t \in \mathcal{T} \quad (29.10)$$

Equations (29.9) and (29.10) model start-up and shut-down constraints in a more detailed way than the one commonly used in the literature [70, 123], i.e.

$$P_{Si}(t) - P_{Si}(t-1) \leq RU_i \quad \forall i \in \mathcal{I}, \quad \forall t \in \mathcal{T} \quad (29.11)$$

$$P_{Si}(t-1) - P_{Si}(t) \leq RD_i \quad \forall i \in \mathcal{I}, \quad \forall t \in \mathcal{T}$$

since in (29.11) start-up and shut-down variables are used instead of ramp-up and ramp-down limits as in (29.9) and (29.10). Minimum on-line and off-line time constraints are formulated as presented in [40] and in [39]. These are as follows:

Minimum up time:

$$\begin{aligned} \sum_{t=1}^{\Gamma_i} (1 - u_i(t)) &= 0 \quad \forall i \in \mathcal{I} \\ \sum_{\tau=t}^{k+UT_i-1} u_i(\tau) &\geq UT_i w_i(t) \quad \forall i \in \mathcal{I}, \\ &\forall t = \Gamma_i + 1 \dots T - UT_i + 1 \\ \sum_{\tau=t}^T (u_i(\tau) - w_i(t)) &\geq 0 \quad \forall i \in \mathcal{I}, \\ &\forall t = T - UT_i + 2 \dots T \end{aligned} \quad (29.12)$$

Minimum down time:

$$\begin{aligned} \sum_{t=1}^{\Pi_i} u_i(t) &= 0 \quad \forall i \in \mathcal{I} & (29.13) \\ \sum_{\tau=t}^{t+DT_i-1} (1 - u_i(\tau)) &\geq DT_i z_i(t) \quad \forall i \in \mathcal{I}, \\ &\forall t = \Pi_i + 1 \dots T - DT_i + 1 \\ \sum_{\tau=t}^T (1 - u_i(\tau) - z_i(t)) &\geq 0 \quad \forall i \in \mathcal{I}, \\ &\forall t = T - DT_i + 2 \dots T \end{aligned}$$

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:

$$\Gamma_i = \min\{T, (UT_i - \alpha_i^0)u_i(0)\} \quad (29.14)$$

$$\Pi_i = \min\{T, (DT_i - \beta_i^0)(1 - u_i(0))\} \quad (29.15)$$

Finally, the start-up and the shut-down status of the units are managed as follows:

$$\begin{aligned} w_i(t) - z_i(t) &= u_i(t) - u_i(t-1) \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T} \\ w_i(t) + z_i(t) &\leq 1 \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T} \end{aligned} \quad (29.16)$$

Equations (29.16) 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 (29.1)-(29.16) by imposing a scheduling time $T = 1$ h.

29.6 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 29.3 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

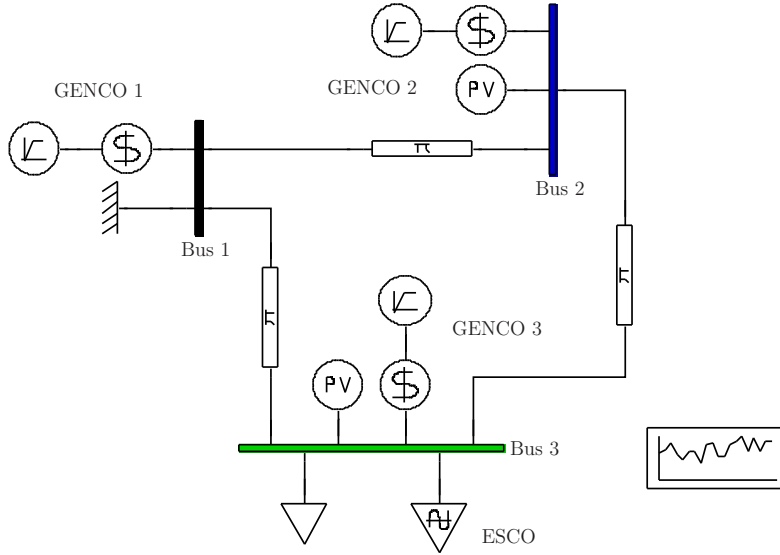


Figure 29.3: PSAT-Simulink model of the three-bus test system.

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 29.1 and 29.2 depict the solution of the single period OPF problem obtained with the IPM MATLAB routine and the PSAT-GAMS interface respectively. Tables 29.3, 29.4, and 29.5 depict the input and output files which are used for exchanging data between PSAT and GAMS.

Figure 29.4 illustrates the demand profile for a 5 hour time horizon, while Figures 29.5 and 29.6 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 29.1: 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	Ps	Ps max	mu max
		[MW]	[MW]	[MW]	
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	Pd	Pd max	mu max
		[MW]	[MW]	[MW]	
Bus3	1294562.538	100	100	100	1294539.717

REACTIVE POWERS

Bus	mu min	Qg min	Qg	Qg max	mu max
		[MVar]	[MVar]	[MVar]	
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	V	V max	mu max	phase
		[p.u.]	[p.u.]	[p.u.]		[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	Q	rho P	rho Q	NCP	Pay
	[MW]	[MVar]	[\$ /MWh]	[\$ /MVarh]	[\$ /MWh]	[\$ /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

FLOWS IN TRANSMISSION LINES

From bus	To bus	I _{ij}	I _{ij} max	mu I _{ij}
		[p.u.]	[p.u.]	
Bus1	Bus2	0.07666	0.4	0
Bus1	Bus3	0.4	0.4	2.6146
Bus2	Bus3	0.32335	0.4	0

FLOWS IN TRANSMISSION LINES

From bus	To bus	I _{ji}	I _{ji} max	mu I _{ji}
		[p.u.]	[p.u.]	
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 29.2: PSAT-GAMS OPF report for the three-bus test system.

PSAT-GAMS Interface							

Standard OPF							
Single-Period Auction							
GAMS routine completed in 0.11565 s							
Power Supplies							

Bus	Ps	Ps max	Ps min				
<i>	[MW]	[MW]	[MW]				
1	52.4252	60	10				
2	27.1322	60	10				
3	20.4425	60	10				
Power Demands							

Bus	Pd	Pd max	Pd min				
<i>	[MW]	[MW]	[MW]				
3	100	100	100				
Generator Reactive Powers							

Bus	Qg	Qg max	Qg min				
<i>	[MVar]	[MVar]	[MVar]				
1	0.8294	150	-150				
2	0.5522	80	-20				
3	61.3228	80	-20				
Power Flow Solution							

Bus	V	theta	PG	PL	QG	QL	
<i>	[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	
Prices and Pays							

Bus	LMP	NCP	Pay S	Pay D			
<i>	[\$/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			
Flows on Transmission Lines							

From Bus	To Bus	Iij	Iijmax	Iij margin	Iji	Ijimax	Iji margin
<i>	<j>	[p.u.]	[p.u.]	[p.u.]	[p.u.]	[p.u.]	[p.u.]
1	2	0.0767	0.4000	0.3233	0.0767	0.4000	0.3233
1	3	0.4000	0.4000	0.0000	0.4000	0.4000	0.0000
2	3	0.3234	0.4000	0.0766	0.3234	0.4000	0.0766
Totals							

Total Losses = 0 [MW]							
Bid Losses = 0 [MW]							
Total demand = 100 [MW]							
Total Transaction Level = 100 [MW]							
IMD Pay = 167.3758 [\$/h]							

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 29.3: Input file psatglobals.gms for the three-bus test system.

```

$setglobal nBus '3'
$setglobal nLine '3'
$setglobal nPs '3'
$setglobal nPd '1'
$setglobal nSW '1'
$setglobal nPV '2'
$setglobal nBusref '1'
$setglobal control '3'
$setglobal flow '1'

```

Table 29.4: Input file `psatdata.gms` for the three-bus test system.

```

$onempty
$skill Gh
parameter Gh /
/;
$skill Bh
parameter Bh /
1.1 -20.000000
2.1 10.000000
3.1 10.000000
1.2 10.000000
2.2 -20.000000
3.2 10.000000
1.3 10.000000
2.3 10.000000
3.3 -20.000000
/;
$skill Li
parameter Li /
1.1 1.000000
2.1 1.000000
3.2 1.000000
/;
$skill Lj
parameter Lj /
1.2 1.000000
2.3 1.000000
3.3 1.000000
/;
$skill Ps_idx
parameter Ps_idx /
1.1 1.000000
2.2 1.000000
3.3 1.000000
/;

$skill Pd_idx
parameter Pd_idx /
3.1 1.000000
/;
$skill S
parameter S /
1.Psmax 0.600000
2.Psmax 0.600000
3.Psmax 0.600000
1.Psmin 0.100000
2.Psmin 0.100000
3.Psmin 0.100000
1.Csa 0.060000
2.Csa 0.040000
3.Csa 0.080000
1.Csb 9.800000
2.Csb 10.700000
3.Csb 12.600000
1.Csc 10.000000
2.Csc 20.000000
3.Csc 25.000000
1.ksu 1.000000
2.ksu 1.000000
3.ksu 1.000000
/;
$skill D
parameter D /
1.Pd0 1.000000
1.Pdmax 1.000000
1.Pdmin 1.000000
1.tgphi 0.600000
/;
$skill X
parameter X /

1.V0 1.000000
2.V0 1.000000
3.V0 1.000000
1.Qgmax 1.500000
2.Qgmax 0.800000
3.Qgmax 0.800000
1.Qgmin -1.500000
2.Qgmin -0.200000
3.Qgmin -0.200000
1.Vmax 1.100000
2.Vmax 1.100000
3.Vmax 1.100000
1.Vmin 0.900000
2.Vmin 0.900000
3.Vmin 0.900000
1.ksw 1.000000
2.kpv 1.000000
3.kpv 1.000000
/;
$skill N
parameter N /
1.b -10.000000
2.b -10.000000
3.b -10.000000
1.Pijmax 0.400000
2.Pijmax 0.400000
3.Pijmax 0.400000
1.Pjimax 0.400000
2.Pjimax 0.400000
3.Pjimax 0.400000
/;
$offempty

```

Table 29.5: Output file `psatsol.m` for the three-bus test system.

```

nout = nout + 1;
varargout{nout} = zeros(3,1);
varargout{nout}(1) = 5.2425221822350E-01;
varargout{nout}(2) = 2.7132243262640E-01;
varargout{nout}(3) = 2.0442534915010E-01;
nout = nout + 1;
varargout{nout} = zeros(1,1);
varargout{nout}(1) = 1.0000000000000E+00;
nout = nout + 1;
varargout{nout} = zeros(3,1);
varargout{nout}(1) = 1.1000000000000E+00;
varargout{nout}(2) = 1.1000000000000E+00;
varargout{nout}(3) = 1.1000000000000E+00;
nout = nout + 1;
varargout{nout} = zeros(3,1);
varargout{nout}(2) = -6.9690606686979E-03;
varargout{nout}(3) = -3.6365640167873E-02;
nout = nout + 1;
varargout{nout} = zeros(3,1);
varargout{nout}(1) = 8.2938340399436E-03;
varargout{nout}(2) = 5.5216188162457E-03;
varargout{nout}(3) = 6.1322778477527E-01;
nout = nout + 1;
varargout{nout} = zeros(3,1);
varargout{nout}(1) = 2.0285044364470E+01;
varargout{nout}(2) = 2.1552897306532E+01;

varargout{nout}(3) = 2.2821267457505E+01;
nout = nout + 1;
varargout{nout} = zeros(3,1);
varargout{nout}(1) = 7.6659512222775E-02;
varargout{nout}(2) = 4.0000000000000E-01;
varargout{nout}(3) = 3.2335073143047E-01;
nout = nout + 1;
varargout{nout} = zeros(3,1);
varargout{nout}(1) = 7.6659512222775E-02;
varargout{nout}(2) = 4.0000000000000E-01;
varargout{nout}(3) = 3.2335073143047E-01;
nout = nout + 1;
varargout{nout} = zeros(3,1);
varargout{nout}(1) = -3.5091452691537E-01;
varargout{nout}(2) = -5.0727651287779E-01;
varargout{nout}(3) = -6.6380624368687E-01;
nout = nout + 1;
varargout{nout} = zeros(3,1);
varargout{nout}(2) = -4.1832967567593E+00;
nout = nout + 1;
varargout{nout} = zeros(3,1);
nout = nout + 1;
varargout{nout} = 2.0000000000000E+00;
nout = nout + 1;
varargout{nout} = 1.0000000000000E+00;

```

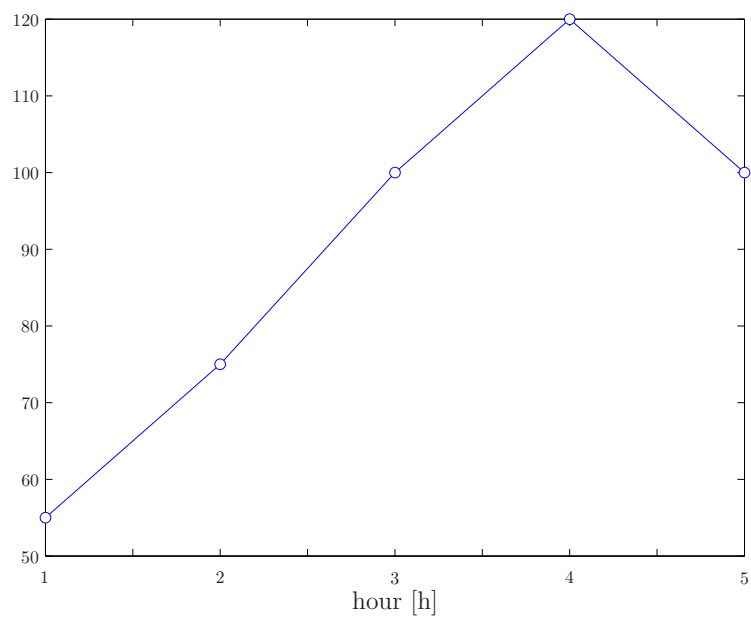


Figure 29.4: Demand profile for the multiperiod auction.

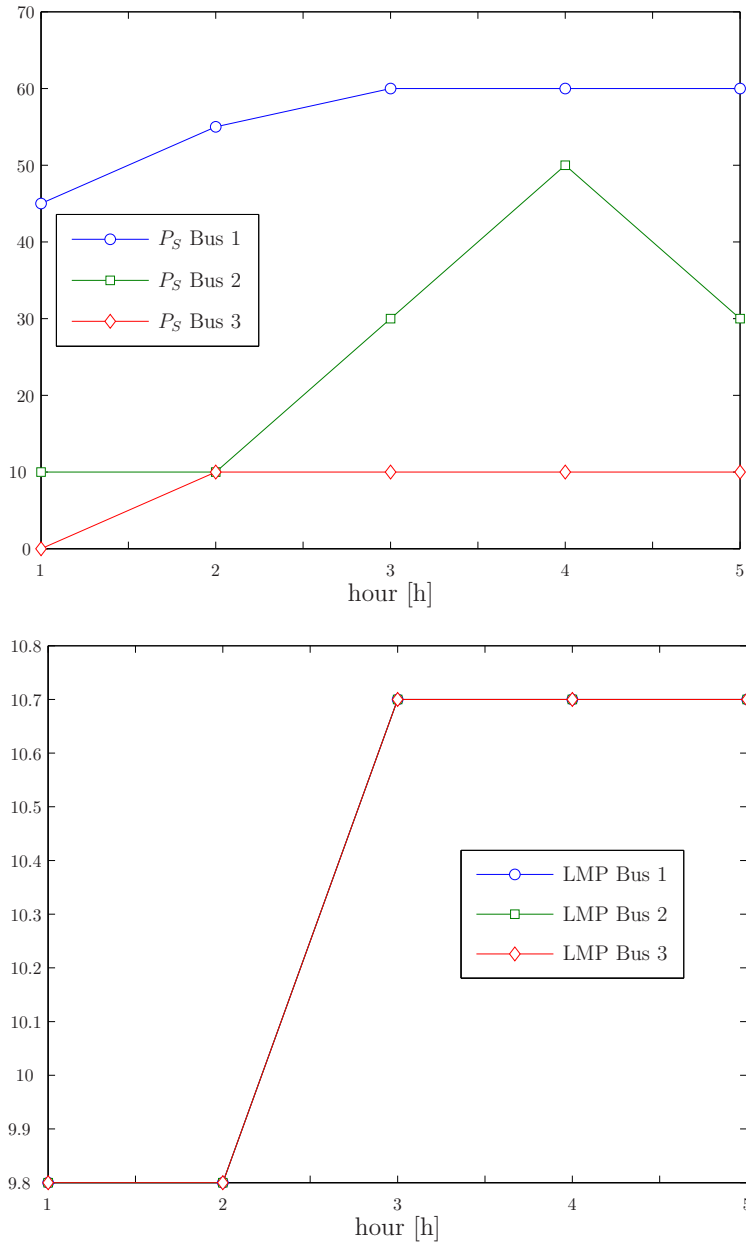


Figure 29.5: Supply and LMP profiles for the multiperiod auction without P_{mn}^{\max} limits.

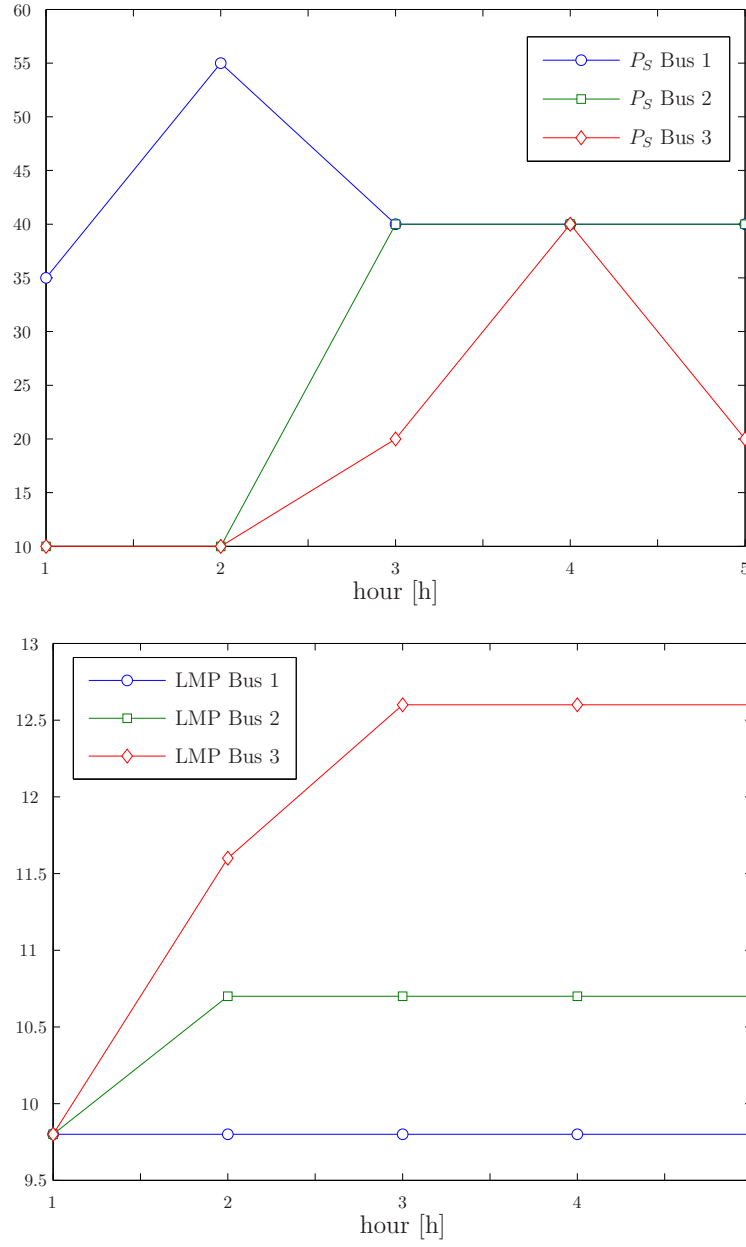


Figure 29.6: Supply and LMP profiles for the multiperiod auction with P_{mn}^{\max} limits.

Chapter 30

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 30.3 for limitations and ToDos.

30.1 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, instead, 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 `uwpflow.exe` (e.g. `uwpflow_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`.

30.2 Graphical User Interface

Figure 30.1 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.

30.3 Limitations and ToDos

The PSAT-UWPFLOW interface is an 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. Furthermore, 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 10).

Furthermore the interface allows exporting PSAT models to UWPFLOW and running UWPFLOW with the proper options, but not viceversa, i.e. UWPFLOW results and the power flow solutions are not loaded in PSAT. Furthermore, the user is always asked if UWPFLOW power flow solution should be loaded in PSAT. Finally, continuation power flow solutions, i.e. nose curves, are plotted in a separate 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, I will send you the executable files. However, be aware that the reference UWPFLOW version will remain the original one, which is freely distributed by Prof. C. Cañizares.

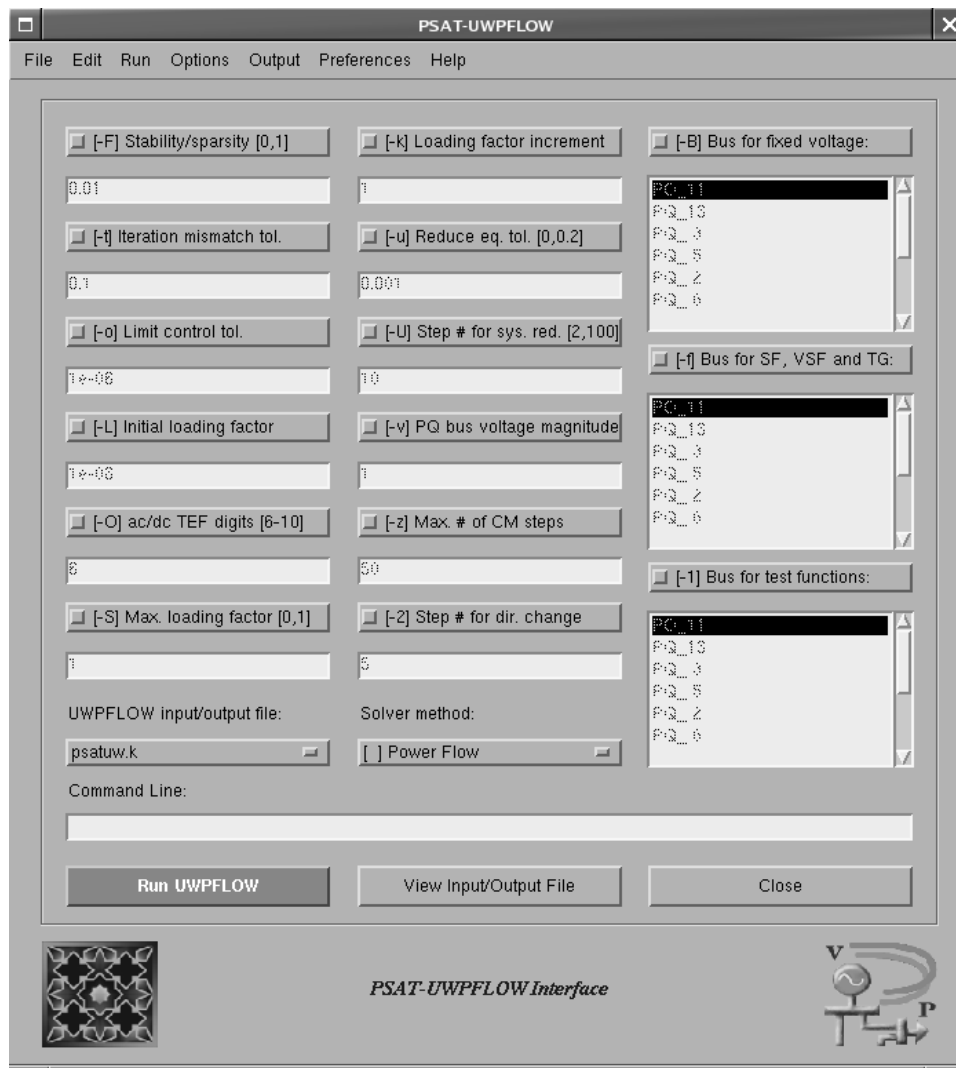


Figure 30.1: GUI of the PSAT-UWPFLOW interface.

30.4 Example

This section illustrates the usage of the PSAT-UWPFLOW interface by means of the 6-bus text system, whose data are reported in Appendix F.2.

Firstly, 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 30.1.⁴

Results are stored in the file `psatuw.pf`,⁵ which is located in the same folder as the PSAT data file (see Table 30.2). 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 30.3 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 30.4. When this file is created, the PSAT-UWPFLOW interface will load and display the data in a MATLAB figure, as depicted in Fig. 30.2. 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 30.1: 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	0.25000000	0.0600000	0	0	0	0	0	0	0.0000	0.0000	0.3082
3	6	1	1	1	0	0.0200000	0.10000000	0.0200000	0	0	0	0	0	0	0.0000	0.0000	1.3973
4	5	1	1	1	0	0.2000000	0.40000000	0.0800000	0	0	0	0	0	0	0.0000	0.0000	0.1796
3	5	1	1	1	0	0.1200000	0.26000000	0.0500000	0	0	0	0	0	0	0.0000	0.0000	0.6585
5	6	1	1	1	0	0.1000000	0.30000000	0.0600000	0	0	0	0	0	0	0.0000	0.0000	0.2000
2	4	1	1	1	0	0.0500000	0.10000000	0.0200000	0	0	0	0	0	0	0.0000	0.0000	1.3740
1	2	1	1	1	0	0.1000000	0.20000000	0.0400000	0	0	0	0	0	0	0.0000	0.0000	0.2591
1	4	1	1	1	0	0.0500000	0.20000000	0.0400000	0	0	0	0	0	0	0.0000	0.0000	0.9193
1	5	1	1	1	0	0.0800000	0.30000000	0.0600000	0	0	0	0	0	0	0.0000	0.0000	0.8478
2	6	1	1	1	0	0.0700000	0.20000000	0.0500000	0	0	0	0	0	0	0.0000	0.0000	0.9147
2	5	1	1	1	0	0.1000000	0.30000000	0.0400000	0	0	0	0	0	0	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 30.2: 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

TSCs -> 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	Pshunt	j	Bus	C	Pij	Plosses	Iij (A)	kVi/kVj	T	Controlled Bus	
n		Name	d(deg)	d(rad)	Qg(MVAR)	Qload	Qshunt		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	25.63					
								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 30.3: 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 30.4: 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
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

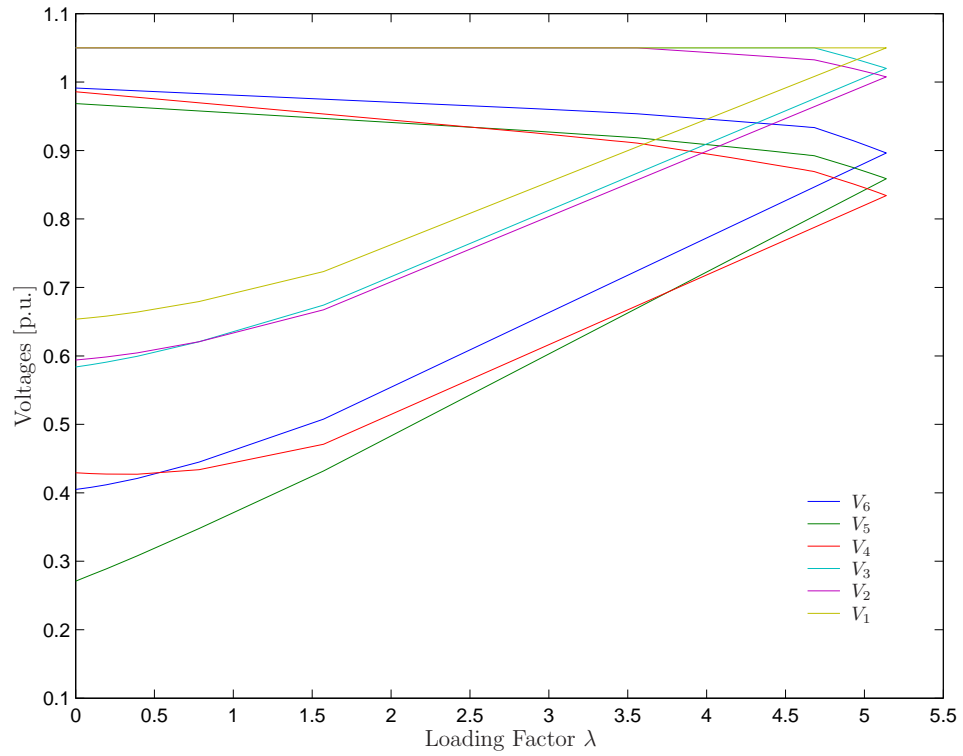
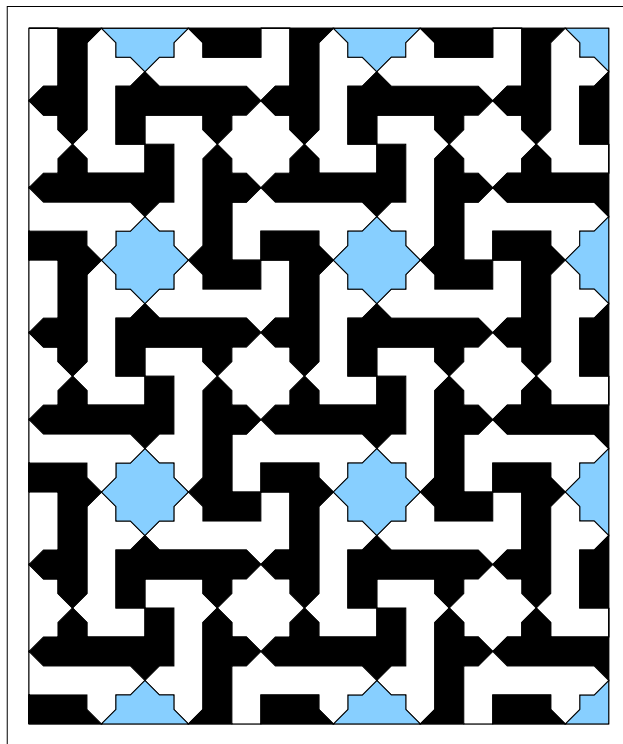


Figure 30.2: UWPFLOW nose curves for the 6-bus test systems. Results are obtained enforcing reactive power limits of generators. Compare these results with the PSAT CPF results depicted in Fig. 5.8.

Part VII

Libraries



Chapter 31

Numeric Linear Analysis

by **Alberto Del Rosso**¹

The library for numeric linear analysis computes output and input matrices A , B , C and D for small signal stability analysis, as follows:

$$\begin{aligned}\Delta\dot{x} &= A\Delta x + B\Delta u \\ \Delta y &= C\Delta x + D\Delta u\end{aligned}\tag{31.1}$$

where $x \in \mathbb{R}^n$ are the state variables, $u \in \mathbb{R}^m$ the input variables and $y \in \mathbb{R}^p$ the output variables. Currently supported input variables are reference voltages of AVRs, reference speeds of turbine governors, and additional input signal of SVC controllers.

31.1 Description

The numeric linear analysis library is composed of three functions, as follows:

fex_abcd.m: computes output and input matrices for linear analysis. This is the main function which internally calls the other library functions.

fex_nrloop.m: solves Newton-Raphson loop. Used for B matrix computations.

fex_lineflows.m: computes power flows in transmission lines.

The function **fex_abcd.m** is to be run from the MATLAB prompt after solving power flow and computes output and input matrices A , B , C and D for linear analysis. Available output and input variables are described in the function source

¹Dr. Alberto Del Rosso is with Mercados Energeticos, Buenos Aires, Madrid, Washington D.C. and with National University of Technology, Buenos Aires.
E-mail: adelrosso@mercadosenergeticos.com

code, while matrices for other variables can be easily added using this code as a template. The function evaluates Jacobian matrices via numerical differentiation. Functions for numeric linear analysis have been written for the command line version of PSAT, but can also run if using PSAT GUIs.

Jacobian matrices and settings for the numeric linear analysis library are contained in the structure `NLA`, which has the following fields:

1. `tol`: minimum state variation Δx for numeric Jacobian computations. Default value is 10^{-5} .
2. `a_sys`: numeric state matrix A .
3. `b_Vr`: numeric matrix B for exciter reference voltages V_{ref} .
4. `b_Tr`: numeric matrix B for governor reference speeds ω_{ref} .²
5. `b_svc`: numeric matrix B for SVC additional signals.
6. `c_y`: numeric matrix C for algebraic variables.
7. `c_ps`: numeric matrix C for active power flows P_{ij} .
8. `c_qs`: numeric matrix C for reactive power flows Q_{ij} .
9. `c_pr`: numeric matrix C for active power flows P_{ji} .
10. `c_qr`: numeric matrix C for reactive power flows Q_{ji} .
11. `c_Is`: numeric matrix C for current flows I_{ij} .
12. `c_Ir`: numeric matrix C for current flows I_{ji} .

Note: the current version of the numeric linear analysis functions is preliminary and only a few tests have been performed so far.

31.2 Test cases

The WSCC 9-bus test system described in [101] (see also the Appendix F.3) is used in this section to illustrate results obtained with the numeric linear analysis functions. In the following examples, the command line version of PSAT will be used and it will be assumed that the data files `d_009.mdl.m` is in the current path. The state matrix A which is obtained by means of `fex_abcd` can be readily tested by means of the analytical state matrix calculated by PSAT. The consistency and the accuracy of input/output matrices B , C and D is checked by comparing time domain response of the linearized and the full non-linear systems.

²Observe that PSAT does not currently support additional signal for SVC controllers.

Table 31.1: State matrix eigenvalues for the 9-bus test system

Numerical State Matrix	Analytical State Matrix
-1000	-1000
-1000	-1000
-1000	-1000
$-0.7075 \pm 11.60651i$	$-0.7075 \pm 11.60652i$
$-0.18645 \pm 7.6324i$	$-0.18645 \pm 7.63242i$
$-5.4838 \pm 7.94648i$	$-5.48381 \pm 7.94648i$
$-5.21801 \pm 7.81343i$	$-5.21801 \pm 7.81343i$
$-5.3211 \pm 7.91899i$	$-5.3211 \pm 7.91899i$
-5.19713	-5.19712
-3.40392	-3.4039
$-0.44279 \pm 1.21198i$	$-0.44279 \pm 1.21199i$
$-0.43829 \pm 0.74015i$	$-0.43829 \pm 0.74015i$
$-0.42483 \pm 0.49685i$	$-0.42483 \pm 0.49685i$
$0 \pm 0.00652i$	$0 \pm 0.0106i$
-3.22581	-3.22581

31.2.1 Comparison of state matrices

Table 31.1 depicts the eigenvalues obtained by means of the numerical and the analytical differentiation, respectively. To obtain these results, the power flow has to be solved first; then one has to compute eigenvalues by means of the numerical and the analytical differentiation, as follows:

```
>> runpsat('d_009_md1.m','data')
>> runpsat('pf')
>> fex_abcd;
>> mu1 = eig(NLA.a_sys);
>> As = DAE.Fx - DAE.Fy*inv(DAE.Jlfv)*DAE.Gx;
>> mu2 = eig(full(As));
```

31.2.2 Results for a change of an exciter reference voltage

Following time domain simulations assume a 2% step in the exciter reference voltage V_{ref} of synchronous machine 1. A sample code which uses the numeric linear analysis function is as follows:

```
>> runpsat('d_009_md1.m','data')
>> runpsat('pf')
>> [Ps0,Qs0,Pr0,Qr0,Is0,Ir0] = fex_lineflows;
>> t = 0:0.01:20;
>> u(1:length(t)) = 0.02;
>> d = zeros(Bus.n,1);
```

```

>> x0 = zeros(DAE.n,1);
>> [Vlinear,x1V] = lsim(NLA.a_sys,NLA.b_Vr(:,1),NLA.c_V,d,u,t,x0);
>> d = zeros(length(Line.n),1);
>> x0 = zeros(DAE.n,1);
>> [Qs,x1V] = lsim(NLA.a_sys,NLA.b_Vr(:,1),NLA.c_qs,d,u,t,x0);
>> [Ps,x1V] = lsim(NLA.a_sys,NLA.b_Vr(:,1),NLA.c_ps,d,u,t,x0);
>> vmat = ones(size(Vlinear));
>> v1 = Vlinear + vmat*diag(Snapshot.V);
>> qmat = ones(size(Qs));
>> q1 = Qs+qmat*diag(Qs0);
>> pmat = ones(size(Ps));
>> p1 = Ps+pmat*diag(Ps0);

```

Once computed the linear model using `fex_abcd`, the time response is obtained by means of the MATLAB function `lsim`. Variations of bus voltages, reactive powers and active power flows are contained in the matrices `Vlinear`, `Qs` and `Ps`, respectively. Finally, initial values of bus voltages and power flows are added to variations (see `v1`, `q1` and `p1`) to ease comparisons with non-linear system results.

The time domain simulation of the full non-linear system is obtained as follows:

```

>> runpsat('d_009_mdl.m','data')
>> runpsat('pf')
>> Settings.fixt = 1;
>> Settings.tstep = 0.01;
>> Exc.vrif0(1) = Exc.vrif0(1)*1.02;
>> runpsat('td')

```

Observe that the instructions `Settings.fixt = 1;` and `Settings.tstep = 0.01;` fix the simulation time step to $\Delta t = 0.1$ s in order to reproduce the same conditions as the linearized system. Figures 31.1, 31.2 and 31.3 depict results for the time domain simulations obtained with the linearized and the full non-linear system. Figure 31.1 depicts voltage magnitudes at buses 6 and 7, while Figs. 31.2 and 31.3 illustrate reactive power flow through transformer 2-7 and transmission line 6-4, and the active power flow through transformer 2-7, respectively. Observe that the maximum difference within simulation time range is less than 0.2%.

31.2.3 Results for a change of governor reference speeds

Following time domain simulations assume a 0.5% step in the reference speeds ω_{ref} of turbine governors of the synchronous machines connected at buses 1 and 3. Observe that one has to change the original 9-bus test system file by adding the governors data, as follows:

```

Tg.con = [ ...
2      2      1      0.05      1.0      0.1      0.1      0.3;
3      2      1      0.05      1.0      0.1      0.1      0.3];

```

Code for the numeric linear analysis:

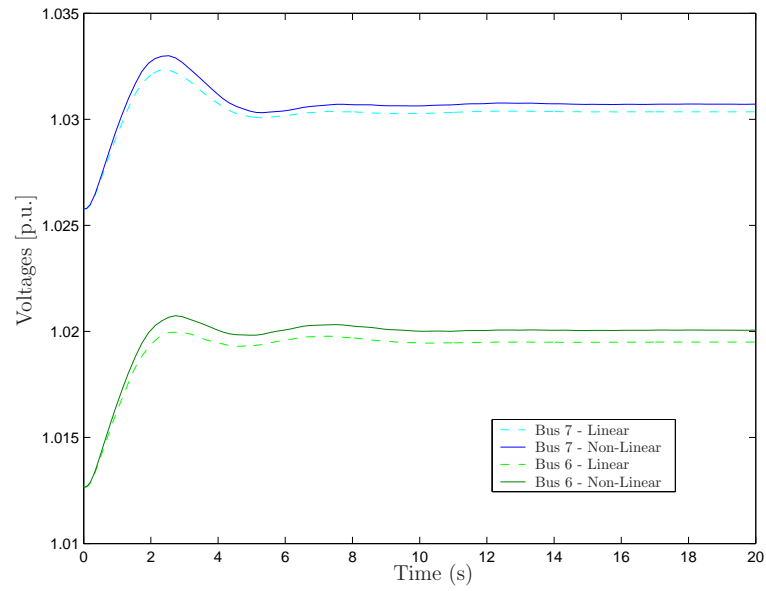


Figure 31.1: Comparison of voltages at buses 6 and 7 for a 2% step in the reference voltage of machine 2.

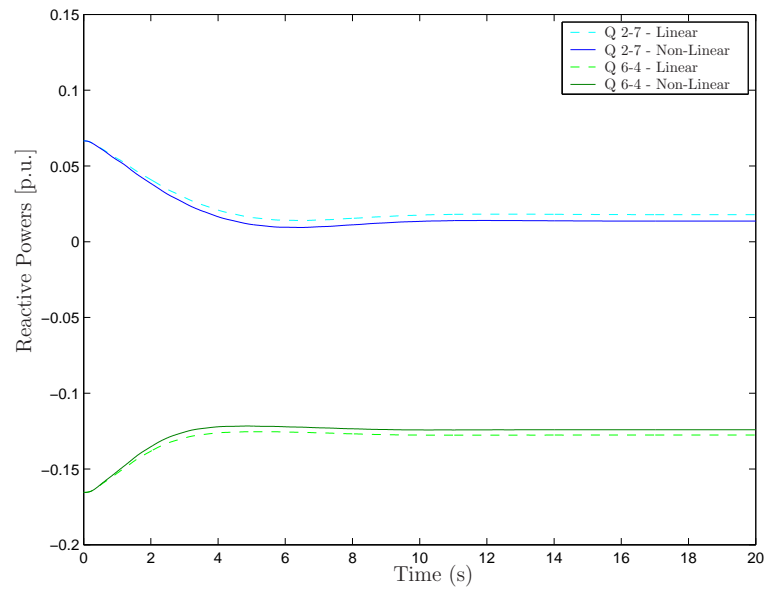


Figure 31.2: Comparison of reactive powers flows in lines 2-7 and 6-4 for a 2% step in the reference voltage of machine 2.

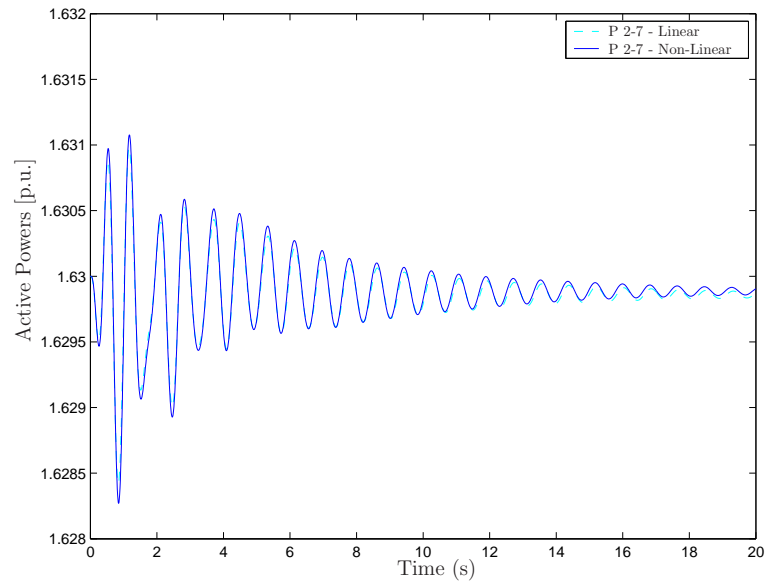


Figure 31.3: Comparison of active powers flows in line 2-7 for a 2% step in the reference voltage of machine 2.


```

>> runpsat('d_009_mdl.m','data')
>> runpsat('pf')
>> [Ps0,Qs0,Pr0,Qr0,Is0,Ir0] = fex_lineflows;
>> fex_abcd;
>> t = 0:0.01:20;
>> u1(1:length(t)) = 0.005;
>> u = [u1',u1'];
>> d = zeros(Line.n,2);
>> x0 = zeros(DAE.n,1);
>> [Ps,x1V] = lsim(NLA.a_sys,NLA.b_Tr,NLA.c_ps,d,u,t,x0);
>> pmat = ones(size(Ps));
>> pl = Ps+pmat*diag(Ps0);

```

Code for the non-linear time domain simulation:

```

>> runpsat('d_009_mdl.m','data')
>> runpsat('pf')
>> Settings.fixt = 1;
>> Settings.tstep = 0.01;
>> Tg.con(:,3) = Tg.con(:,3)*1.005;
>> runpsat('td')

```

Figures 31.4 and 31.5 depict the rotor speed of generator connected at bus 3 while Fig. 31.6 illustrates the active power flow through transformer 2-7.

31.2.4 Results for a change of a SVC reference voltage

Following time domain simulations assume a 2% step in the reference voltage V_{ref} of a SVC connected at bus 8 for the 9-bus test system. Observe that one has to change the original file by adding the SVC and the dummy PV generator data, as follows:

```

PV.con = [ ...
    2 100 18 1.63 1.025 99 -99 1.1 0.9 1;
    3 100 13.8 0.85 1.025 99 -99 1.1 0.9 1;
    8 100 13.8 0.00 1.025 99 -99 1.1 0.9 1];

Svc.con = [8 100 230 60 1 10 100 1 1 -1 ...
    0.001 0 1 0.01 0.2 0.1 ];

```

Code for the numeric linear analysis:

```

>> runpsat('d_009_mdl.m','data')
>> runpsat('pf')
>> fex_abcd;
>> t = 0:0.01:20;
>> u(1:length(t)) = 0.02;
>> d = zeros(Bus.n,1);

```

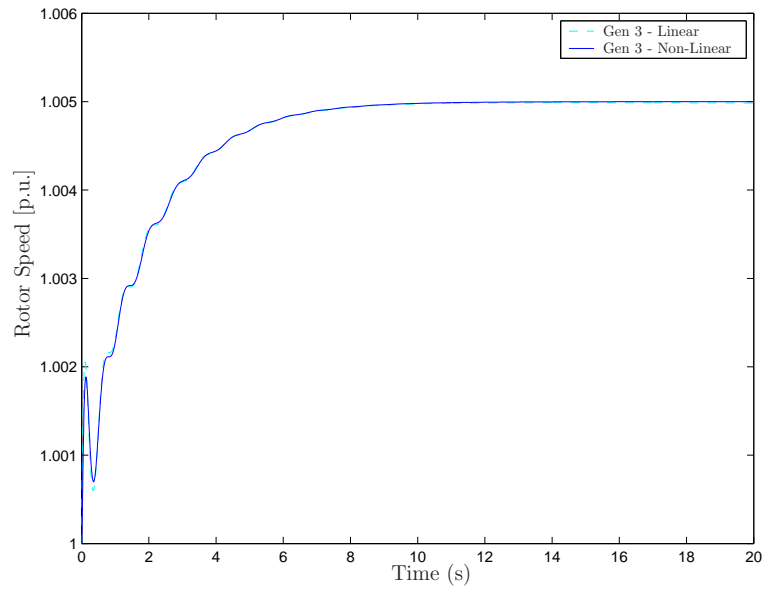


Figure 31.4: Comparison of rotor speeds for a 0.5% step in the reference speed of all machine governors.

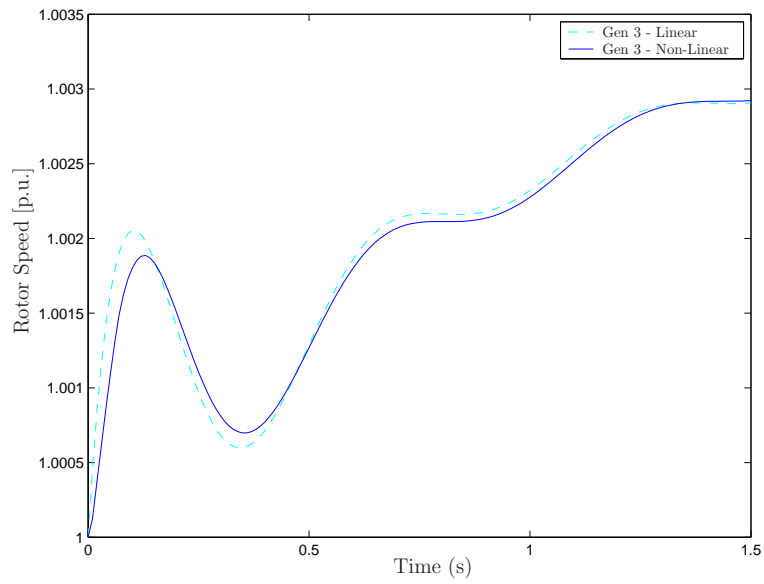


Figure 31.5: Detail of the comparison of rotor speeds for a 0.5% step in the reference speed of all machine governors.

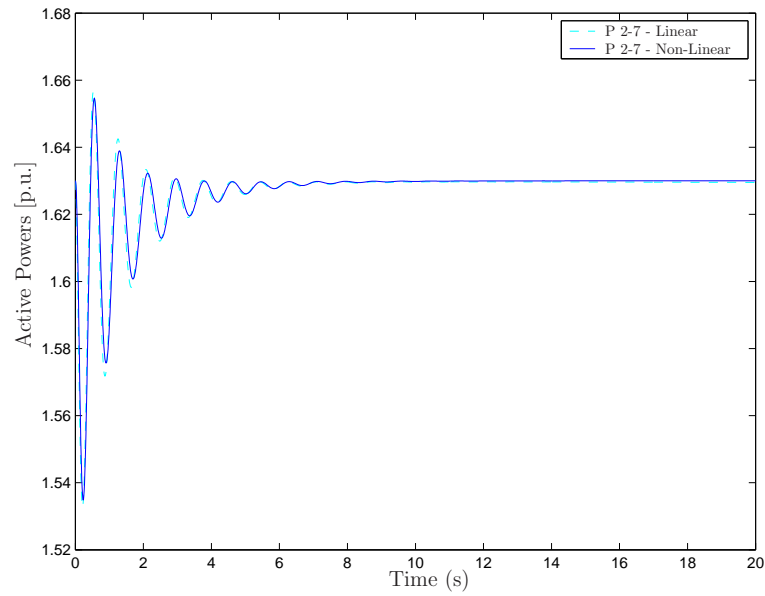


Figure 31.6: Comparison of active powers flows in line 2-7 for a 0.5% step in the reference speed of all machine governors.

```
>> xi = DAE.x;
>> x0 = zeros(DAE.n,1);
>> [dV,dx] = lsim(NLA.a_sys,NLA.b_svc,NLA.c_V,d,u,t,x0);
>> d = zeros(length(Line.n),1);
>> x1 = [dx' + diag(xi)*ones(size(dx'))]';
>> vmat = ones(size(dV));
>> v1 = dV + vmat*diag(Snapshot.V);
```

Code for the non-linear time domain simulation:

```
>> runpsat('d_009_md1.m','data')
>> runpsat('pf')
>> Settings.fixt = 1;
>> Settings.tstep = 0.01;
>> Svc.Vref = Svc.Vref+0.02;
>> runpsat('td')
```

Figures 31.7 and 31.8 depict the SVC state variable and the voltage magnitude at bus 8, respectively.

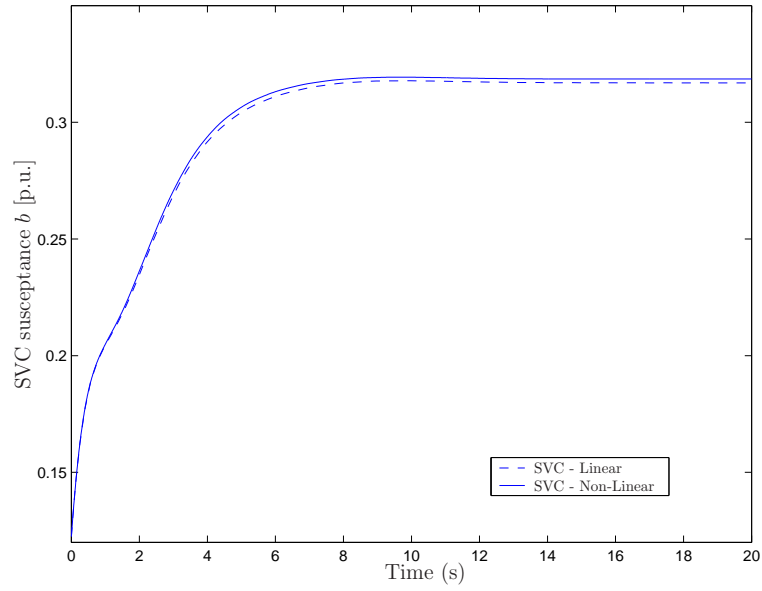


Figure 31.7: Comparison of SVC state variables for a 2% step in the reference voltage of SVC regulators.

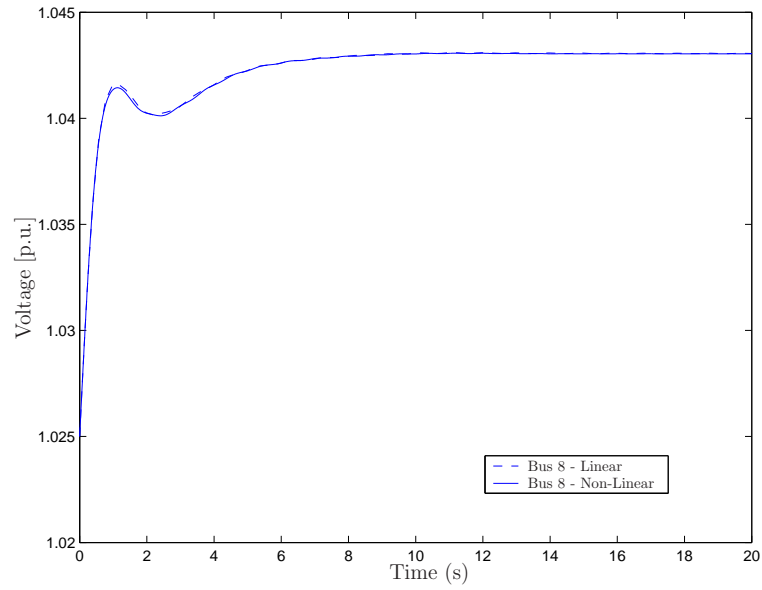
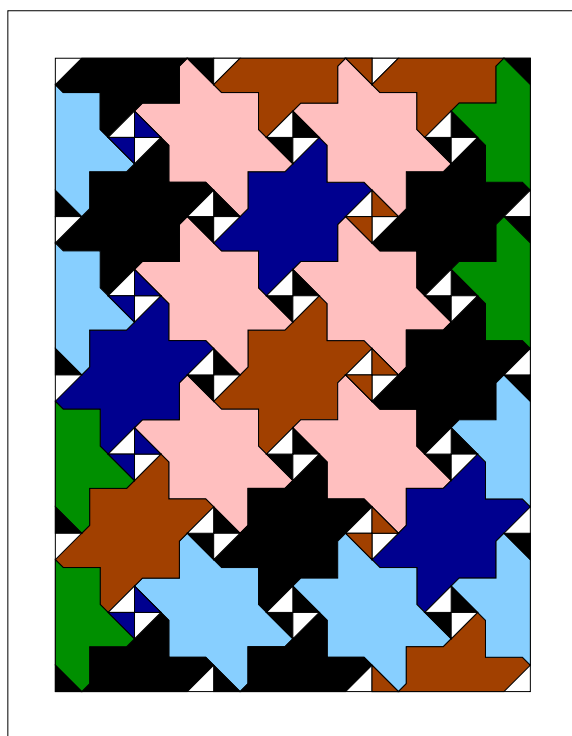


Figure 31.8: Comparison of voltages at bus 8 for a 2% step in the reference voltage of SVC regulators.

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

`beep` beep control

`0` disabled

`1` enabled

`chunk` initial dimension of output arrays

`color` default GUI colors

`conv` system base conversion and checks

`0` disabled

`1` enabled

`date` release date of the current PSAT version

`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

1 enabled

dynmit maximum number of iteration for dynamic analyses

dyntol error tolerance for dynamic analyses

fixt set fixed time step

0 disabled

1 enabled

format Data file format number (default 1)

freq system frequency rating [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

lftol error tolerance for static analyses

lfmit maximum number of iteration for static analyses

lftime elapsed time for power flow computations

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 initialization of the **Snapshot** structure after power flow computation

0 disabled

1 enabled

method integration method

- 1 forward Euler method
- 2 trapezoidal method

mv model version of the currently loaded SIMULINK model

mva system power rating [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 defines if the current PSAT session is running on Octave

- 0 PSAT is running on MATLAB
- 1 PSAT is running on Octave

ok output of the `fm_choice` dialog box

- 0 yes
- 1 no

pfsolver select power flow solver

- 1 Newton-Raphson method
- 2 XB variation of fast decoupled power flow
- 3 BX variation of fast decoupled power flow

plot plot during time domain simulations

- 0 disabled
- 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

0 disabled
1 enabled

pv2pq generator reactive power limit control during power flow computation

0 disabled
1 enabled

rad system frequency rating [rad]

show display iteration status and messages

0 disabled
1 enabled

showlf display report GUI after power flow solution

0 disabled
1 enabled

shuntvalues include shunt power absorptions in transmission line balances when writing the report file of the current case solution

on include shunts in transmission lines
off do not include shunts in transmission lines

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
1 enabled

status display convergence error of the current iteration on the main window

0 disabled
1 enabled

t0 initial simulation time [s]

tf final simulation time [s]

tstep fixed time step value [s]

`tviewer` current text viewer

`version` current PSAT version

`violations` enforce limit violation checks when writing the report file of the current case solution

on disabled
off enabled

`xlabel` label for plotting variables

`zoom` zoom plotting variables

0 disabled
1 enabled

A.2 Other Settings

Fig: handles of the GUI windows. The handle value is 0 if the associated window is not open. The handle names are as follows:

<code>about</code>	PSAT information GUI
<code>author</code>	author's pic
<code>clock</code>	analogical watch window
<code>comp</code>	user defined component browser
<code>cpf</code>	continuation power flow GUI
<code>cset</code>	mask for user defined component properties
<code>dir</code>	file browser and data format conversion GUI
<code>eigen</code>	small signal stability analysis GUI
<code>gams</code>	GUI for the PSAT-GAMS interface
<code>hist</code>	command history GUI
<code>laprint</code>	GUI for the \LaTeX settings
<code>lib</code>	GUI for limit-induced bifurcations
<code>license</code>	GUI that displays the program licence
<code>line</code>	GUI for editing the plotted line properties
<code>main</code>	PSAT main window
<code>make</code>	GUI for building user defined components
<code>matrx</code>	GUI for Jacobian matrix visualization
<code>opf</code>	optimal power flow GUI
<code>plot</code>	GUI for plotting variables
<code>plotsel</code>	GUI for selecting output variables for TDs
<code>pmu</code>	PMU placement GUI
<code>pset</code>	mask for parameter properties
<code>simset</code>	GUI for setting SIMULINK model properties
<code>setting</code>	general setting GUI

<code>snap</code>	GUI for setting snapshots
<code>snb</code>	direct method for SNB GUI
<code>sset</code>	mask for auxiliary variable properties (<i>not used</i>)
<code>stat</code>	power flow report GUI
<code>theme</code>	theme browser
<code>threed</code>	3D system visualization
<code>tvviewer</code>	GUI for selecting the text viewer
<code>update</code>	GUI for installing and uninstalling user defined components
<code>uwpflow</code>	GUI for the PSAT-UWPFLOW interface
<code>xset</code>	mask for state variable properties
<code>warranty</code>	GUI that displays the warranty conditions

File: data and disturbance file names, as follows:

<code>data</code>	current data file name
<code>pert</code>	current disturbance file name

Path: path strings of the most commonly used folders, as follows:

<code>local</code>	current workspace path
<code>data</code>	current data file path
<code>pert</code>	current disturbance file path
<code>psat</code>	PSAT path
<code>images</code>	absolute path of the secondary folder <code>images</code>
<code>build</code>	absolute path of the secondary folder <code>build</code>
<code>themes</code>	absolute path of the secondary folder <code>themes</code>
<code>filters</code>	absolute path of the secondary folder <code>filters</code>

Hdl: handles of the most used graphic objects.

<code>hist</code>	command history listbox in the command history GUI
<code>text</code>	message static text in the main window
<code>status</code>	axis for convergence status in the main window
<code>frame</code>	frame of message text in the main window
<code>bar</code>	axis for the progress bar in the main window
<code>axes</code>	PSAT logo axis in the main window

Snapshot: snapshot data.

<code>name</code>	cell array of snapshot names
<code>time</code>	array of times associated to the defined snapshots
<code>y</code>	vector of algebraic variables
<code>x</code>	vector of state variables
<code>Ybus</code>	network admittance matrix
<code>Pg</code>	vector of generator real powers injected at buses
<code>Qg</code>	vector of generator reactive powers injected at buses

P1	vector of load real powers absorbed from buses
Q1	vector of load reactive powers absorbed from buses;
Fx	Jacobian matrix of differential equations $F_x = \nabla_x f$
Fy	Jacobian matrix of differential equations $F_y = \nabla_y f$
Gx	Jacobian matrix of algebraic equations $G_x = \nabla_x g$
Gy	Jacobian matrix of algebraic equations $G_y = \nabla_y g$
Ploss	total real losses of the current power flow solution
Qloss	total reactive losses of the current power flow solution

History: command history text and settings.

text	cell array of the last $n = \text{Max}$ commands
string	string for text search within the command history
index	number of the last row where string was found
workspace	enable displaying messages on the MATLAB workspace
Max	maximum number of rows of the text cell array
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
BackgroundColor	background color of the command history GUI
ForegroundColor	foreground color of the command history GUI

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
color07	text color 3
color08	progress bar color
color09	text color for special buttons
color10	text color for special list boxes
color11	axis color
font01	font name for edit texts, list boxes and axes
hdl	handles of graphical objects in the theme manager GUI

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
pert	disturbance file cell array
description	case description (<i>not used</i>)

A.3 System Properties and Settings

DAE differential and algebraic equations, functions and Jacobians matrices. Fields are as follows:

y	algebraic variables y
kg	variable for distributing losses among generators
x	state variables x
n	number of state variables n
m	number of algebraic variables m
npf	dynamic order during power flow n_{PF}
f	differential equations f
g	algebraic equations g
Fx	Jacobian matrix of differential equations $F_x = \nabla_x f$
Fy	Jacobian matrix of differential equations $F_y = \nabla_y f$
Gx	Jacobian matrix of algebraic equations $G_x = \nabla_x g$
Gy	Jacobian matrix of algebraic equations $G_y = \nabla_y g$
Gλ	Jacobian matrix of algebraic equations $G_\lambda = \nabla_\lambda g$
Gk	Jacobian matrix of algebraic equations $G_k = \nabla_k g$
Ac	complete DAE Jacobian matrix
tn	vector of DAE for time domain simulations
t	current simulation time (-1 for static analysis)

SSSA Settings for small signal stability analysis.

matrix matrix type

- 1 reduced dynamic power flow Jacobian J_{LFD_r}
- 2 reduced complete power flow Jacobian J_{LFF_r}
- 3 reduced standard power flow Jacobian J_{LF_r}
- 4 state matrix A_S

map map type

- 1 S -map
- 2 participation factor map
- 3 Z -map

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

report structure containing the small signal stability analysis report

neig number of eigenvalues to be computed (applies only if **method** \neq 1)
eigs vector of eigenvalues
pf matrix of participation factors

SNB Settings for saddle-node bifurcation analysis (direct method).

slack enable distributed slack bus
 0 single slack bus
 1 distributed slack bus
lambda loading parameter λ value
dmdl sensitivity coefficient $\partial P / \partial \lambda$ values
bus generation and load direction buses

LIB Settings for limit-induced bifurcation (direct method).

type LIB type
 1 V_{\max}
 2 V_{\min}
 3 Q_{\max}
 4 Q_{\min}
selbus bus number where applying the limit
slack enable distributed slack bus
 0 single slack bus
 1 distributed slack bus
lambda loading parameter λ value
dmdl sensitivity coefficient $\partial P / \partial \lambda$ values
bus generation and load direction buses

CPF Continuation power flow settings.

method method for corrector step
 1 perpendicular intersection
 2 local parametrization
flow select transmission line flow
 1 current I_{ij}
 2 active power P_{ij}
 3 apparent power S_{ij}
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

sbus slack bus model

    0    distributed slack bus
    1    single slack bus

vlim check voltage limits

    0    disabled
    1    enabled

ilim check transmission line flow limits

    0    disabled
    1    enabled

qlim check generator reactive power 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)

tolc corrector step tolerance

tolf error tolerance for transmission line flows

tolv error tolerance for bus voltages

step step size control

nump maximum number of points to be computed

show show iteration status on main window

    0    disabled
    1    enabled

linit initial value of the loading parameter  $\lambda$ 

lambda loading parameter

kg distributed slack bus variable

pmax maximum power flow limits. This field is filled up by the function
    fm_n1cont as a result of the (N-1) contingency criterion.

hopf check for change of sign of pair of complex conjugate eigenvalues (Hopf
    bifurcation points)

```


- 0 disabled (default)
- 1 enabled

stepcut step size control

- 0 disabled
- 1 enabled (default)

negload include negative active power loads in CPF analysis

- 0 disabled (default)
- 1 enabled

onlynegload use only negative active power loads in CPF analysis

- 0 disabled (default)
- 1 enabled

OPF Optimal power flow settings and outputs.

method method used for computing the variable directions and increments

- 1 Newton directions
- 2 Merhotra Predictor/Corrector

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} (*not tested*)

type type of OPF problem to be solved

- 1 Single OPF (if ω is a vector, the first value is used)
- 2 Pareto 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*)

deltat time step in minutes ofr the daily forecast (*not used*)

lmin minimum value of the loading parameter λ_c

lmax maximum value of the loading parameter λ_c

sigma centering parameter σ

gamma safety factor γ

eps_mu error tolerance of the barrier parameter μ_s

eps1 error tolerance of the power flow equations

eps2 error tolerance of the objective function

omega weighting factor ω (can be a vector)

flatstart set initial guess of system variables

```

1      Flat start ( $V = 1$  and  $\theta = 0$ )
2      Actual power flow solution

conv OPF method convergence status

0      OPF routine did not converge
1      OPF routine converged

guess vector of values for initializing the OPF routine
report cell array of the OPF solution
show display the convergence error of the OPF routine

0      disabled
1      enabled

init OPF solution status

0      to be solved yet
1      standard OPF has been solved
2      multiobjective OPF has been solved
3      Pareto set OPF has been solved

w      actual value of the weighting factor
atc    maximum loading condition for the current OPF solution
line  number of the line to be deleted for N-1 contingency evaluations in the
      maximum loading condition system
tiebreak tiebreak term in the objective function

0      disabled
1      enabled

basepg include base case generation powers

0      disabled
1      enabled

basepl include base case load powers

0      disabled
1      enabled

enflow enforce flow limit inequalities

0      disabled
1      enabled

envolt enforce voltage limit inequalities

0      disabled
1      enabled

enreac enforce generator reactive power inequalities

```

0 disabled
1 enabled

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.)

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.)

obj value of the objective function

ms barrier parameter

dy algebraic variable mismatch

dF equality constraint mismatch

dG objective function mismatch

LMP Locational Marginal Prices of the current solution

NCP Nodal Congestion Prices of the current solution

iter number of iterations to obtain the current solution

gpc active power injections for the critical loading condition

gqc reactive power injections for the critical loading condition

PMU Settings for PMU placement algorithms

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

number current number of PMU

measv number of measured voltages

measc number of measured currents

pseudo number of pseudo-measured currents

noobs current number of non-observable buses

voltage cell array of estimated voltages

angle cell array of estimated angles

location cell array of PMU placement

A.4 Outputs and Variable Names

Varout: output of time domain simulations. Fields are as follows:

<code>t</code>	time vector
<code>vars</code>	output variables

Varname: system variable `TpX` and plain names. Formatted `TpX` names are used for creating legends in the plotting variable GUI. Fields are as follows:

<code>compx</code>	names of components with state variables
<code>fname_x, unam_x</code>	names of all state variables
<code>compy</code>	names of components with algebraic variables
<code>fname_y, unam_y</code>	names of all algebraic variables
<code>fvars</code>	formatted names of output variables
<code>uvars</code>	unformatted names of output variables
<code>nvars</code>	total number of output variables
<code>idx</code>	indexes of selected plot variables
<code>custom</code>	1 if custom selection of plot variables
<code>fixed</code>	1 if fixed selection of plot variables
<code>x</code>	1 if selecting all state variables
<code>V</code>	1 if selecting all bus voltages
<code>PQ</code>	1 if selecting all P and Q injections at buses
<code>Pij</code>	1 if selecting all power flows in transmission lines

A.5 User Defined Models

Comp: component general settings

<code>funct</code>	cell array of all component functions
<code>number</code>	cell array of all component <code>.n</code> fields
<code>prop</code>	component properties
<code>n</code>	total number of installed components
<code>init</code>	enable initialization $\{0, 1\}$
<code>descr</code>	current component description
<code>name</code>	current component name
<code>shunt</code>	shunt component $\{0, 1\}$

Buses Bus connection variables

<code>name</code>	cell array of bus names
<code>n</code>	number of buses

Algeb Algebraic equations and variables

<code>name</code>	cell array of algebraic variables
<code>n</code>	number of algebraic variables
<code>idx</code>	indexes of algebraic variables
<code>eq</code>	cell array of algebraic equations
<code>eqidx</code>	indexes of algebraic equations
<code>neq</code>	number of algebraic equations

State Differential equations and state variables

<code>name</code>	cell array of state variables
<code>n</code>	number of state variables
<code>eq</code>	cell array of differential equations
<code>eqidx</code>	indexes of differential equations
<code>neq</code>	number of differential equations
<code>init</code>	state variable initialization
<code>limit</code>	enable anti-windup limiters
<code>fn</code>	TEX name of the state variable
<code>un</code>	MATLAB name of the state variable
<code>time</code>	time constant name
<code>offset</code>	offset value
<code>nodyn</code>	allow time constant being $T = 0$

Servc Service equations and variables (*not used...*)

<code>name</code>	cell array of service variables
<code>n</code>	number of service variables
<code>eq</code>	cell array of service equations
<code>eqidx</code>	indexes of service equations
<code>neq</code>	number of service equations
<code>init</code>	service variable initialization
<code>limit</code>	enable anti-windup limiters
<code>fn</code>	TEX name of the service variable
<code>un</code>	MATLAB name of the service variable
<code>type</code>	service variable type
<code>offset</code>	offset value
<code>oldidx</code>	cell array of current “external” service variable

Param Parameter variables

<code>name</code>	cell array of parameter names
<code>n</code>	number of parameters
<code>descr</code>	parameter description
<code>type</code>	parameter type
<code>unit</code>	parameter unit

Init1 Variables for initialization

name	cell array of initial variables
n	number of initial variables
idx	indexes of initial variables

A.6 Models

Power Flow Data

Bus	Bus numbers and voltage ratings	Section 10.1
Line	Transmission line and transformer	Section 10.2-10.3.1
Lines	Alternative transmission line	Section 10.2
Twt	Three-winding transformer	Section 10.3.2
SW	Slack bus	Section 10.4
PV	PV generator	Section 10.5
PQ	Constant power load	Section 10.6
PQgen	Constant power generator	Section 10.7
Shunt	Shunt admittance	Section 10.8
Areas	Interchange area	Section 10.9

CPF and OPF Data

Supply	Power supply	Section 11.1
Rsrv	Generator power reserve	Section 11.2
Rmpg	Generator ramping	Section 11.3
Demand	Power demand	Section 11.4
Ydpd	Demand profile	Section 11.5
Rmpl	Power demand ramping	Section 11.6
Vltn	Violation parameters	<i>not used...</i>

Faults & Breakers

Fault	Transmission line fault	Section 12.1
Breaker	Transmission line breaker	Section 12.2

Measurements

Busfreq	Bus frequency measurement	Section 13.1
Pmu	Phasor measurement units	Section 13.2

Loads

Mn	Voltage dependent load	Section 14.1
F1	Frequency dependent load	Section 14.3
P1	ZIP (polynomial) load	Section 14.2
Exload	Exponential recovery load	Section 14.4

Thload	Thermostatically controlled load	Section 14.5
Jimma	Jimma's load	Section 14.6
Mixload	Mixed load	Section 14.7

Machines

Syn	Synchronous machine	Section 15.1
COI	Center of inertia	Section 15.1.9
Mot	Induction motor	Section 15.2

Controls

Tg	Turbine Governor	Section 16.1
Exc	Automatic Voltage Regulator	Section 16.2
Pss	Power System Stabilizer	Section 16.3
Oxl	Overexcitation Limiter	Section 16.4
CAC	Central Area Controller	Section 16.5
Cluster	Cluster Controller	Section 16.5
Pod	Power Oscillation Damper	Section 16.6

Regulating Transformers

Ltc	Load tap changer	Section 17.1
Tap	Tap changer with embedded load	Section 17.2
Phs	Phase shifting transformer	Section 17.3

FACTS

Svc	Static Var Compensator	Section 18.1
Tcsc	Thyristor Controlled Series Capacitor	Section 18.2
Statcom	Static Var Compensator	Section 18.3
Sssc	Static Synchronous Source Series Compensator	Section 18.4
Upfc	Unified Power Flow Controller	Section 18.5
Hvdc	High Voltage DC transmission system	Section 18.6

Wind Turbines

Wind	Wind models	Section 19.1
Cswt	Constant speed wind turbine	Section 19.2.1
Dfig	Doubly fed induction generator	Section 19.2.2
Ddsg	Direct drive synchronous generator	Section 19.2.3

Other Models

Mass	Synchronous machine dynamic shaft	Section 20.1
SSR	Subsynchronous resonance model	Section 20.2
Sofc	Solid Oxide Fuel Cell	Section 20.3

A.7 Command Line Usage

clpsat structure for command line usage of PSAT (defaults refers to the the command line standard behavior):

init command line initialization status

- 0 PSAT is running with the standard GUIs
- 1 command line PSAT is active (default)

mesg status of PSAT messages

- 0 no message
- 1 messages will be displayed in the current output (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

readfile if true, force to read data file before running power flow

- 0 false
- 1 true (default)

showopf if true, force to display OPF result on the standard output running power flow

- 0 false (default)
- 1 true

pq2z if true, force to switch PQ loads to constant impedances before running time domain simulations

- 0 false
- 1 true (default)

viewrep if true, force to visualize report files when created

- 0 false (default)
- 1 true

A.8 Interfaces

GAMS parameters and settings for the PSAT-GAMS interface:

```

method select OPF method
    1    simple auction
    2    market clearing mechanism
    3    standard OPF
    4    VSC-OPF
    5    maximum loading condition
    6    continuation OPF

type solution type
    1    single period auction
    2    multiperiod auction
    3    pareto set auction
    4    unit commitment auction

flow flow type in transmission lines
    0    none
    1    currents
    2    active powers
    3    apparent powers

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

lmin minimum value of  $\lambda$  (float)
lmin_s minimum value of  $\lambda$  (string)
omega weighting factor  $\omega$  values (float)
omega_s weighting factor  $\omega$  values (string)
lmax maximum value of  $\lambda$  (float)
ldir command line options for GAMS calls
libinclude use command line options
    0    disabled
    1    enabled

loaddir use load direction when solving maximum loading condition OPF
    0    disabled
    1    enabled

basepl use base load powers in OPF
    0    disabled
    1    enabled (default)

```

basepg use base generator powers in OPF

- 0 disabled
- 1 enabled (default)

line number of line to be taken out in N-1 contingency analysis

show display results and logs

- 0 disabled
- 1 enabled

UWPFLOW parameters, option and settings for the PSAT-UWPFLOW interface:.

opt list of UWPFLOW options. Refer to UWPFLOW documentation for details [22].

method loading parameter λ value

- 1 power flow
- 2 continuation power flow
- 3 direct method
- 4 parametrized continuation method

file name of output files (default **psatuw**)

command generation and load direction buses

status generation and load direction buses

A.9 Classes

@ARclass	Class for Area components
@AVclass	Class for Exc components
@BFclass	Class for Busfreq components
@BKclass	Class for Breaker components
@BUclass	Class for Bus components
@CCclass	Class for Cac components
@CIclass	Class for COI components
@CLclass	Class for Cluster components
@CSclass	Class for Cswt components
@DDclass	Class for Ddsg components
@DFclass	Class for Dfig components
@DMclass	Class for Demand components
@DSclass	Class for Mass components
@ELclass	Class for Exload components
@FCclass	Class for Sofc components
@FLclass	Class for F1 components
@FTclass	Class for Fault components
@HVclass	Class for Hvdc components
@IMclass	Class for Mot components

@Jiclass	Class for Jimma components
@LNclass	Class for Line components
@LSclass	Class for Lines components
@LTclass	Class for Ltc components
@MNclass	Class for Mn components
@MXclass	Class for Mixload components
@OXclass	Class for Oxl components
@PHclass	Class for Phs components
@PLclass	Class for Pl components
@PMclass	Class for Pmu components
@POclass	Class for Pod components
@PQclass	Class for PQ components
@PSclass	Class for Pss components
@PVclass	Class for PV components
@RGclass	Class for Rmpg components
@RLclass	Class for Rmpl components
@RSclass	Class for Rsrv components
@SHclass	Class for Shunt components
@SRclass	Class for Ssr components
@SSclass	Class for Sssc components
@STclass	Class for Statcom components
@SUclass	Class for Supply components
@SVclass	Class for Svc components
@SWclass	Class for SW components
@SYclass	Class for Syn components
@TCclass	Class for Tcsc components
@TGclass	Class for Tg components
@THclass	Class for Thload components
@TPclass	Class for Tap components
@TWclass	Class for Twt components
@UPclass	Class for Upfc components
@VLclass	Class for Vltm components
@WNclass	Class for Wind components
@YPclass	Class for Ypdp components

Appendix B

Matlab Functions

This appendix lists the MATLAB script files and functions of the PSAT folder. The list is also available on-line (`Contents.m`) by typing
`>> help psat`

General Functions and GUIs

<code>psat</code>	start the program
<code>fm_set</code>	general settings and utilities
<code>fm_var</code>	definition of global variables
<code>fm_main</code>	main GUI

Power Flow

<code>fm_spf</code>	standard power flow routine
<code>fm_nrlf</code>	power flow with fixed state variables
<code>fm_flows</code>	network power flows
<code>fm_dynlf</code>	indexes of state variables (before power flow)
<code>fm_dynidx</code>	indexes of state variables (after power flow)
<code>fm_xfirst</code>	initial guess of state variables
<code>fm_ncomp</code>	indexes of components
<code>fm_inilf</code>	reset variables for power flow computations
<code>fm_stat</code>	GUI for displaying power flow results
<code>fm_base</code>	report of component quantities on system bases
<code>fm_report</code>	writes power flow report files

Direct Methods

<code>fm_snb</code>	Saddle-node bifurcation routine
<code>fm_snbfig</code>	GUI for saddle-node bifurcations
<code>fm_limit</code>	Limit-induced bifurcation routine
<code>fm_snbfig</code>	GUI for limit-induced bifurcations

Continuation Power Flow (CPF)

fm_cpf	continuation power flow
fm_n1cont	N-1 contingency computations
fm_cpffig	GUI for continuation power flow

Optimal Power Flow (OPF)

fm_opfm	optimal power flow
fm_opfsdr	VS constrained optimal power flow
fm_pareto	Pareto set computations
fm_atc	Available transfer capability computations
fm_opffig	GUI for optimal power flow
fm_opfrep	writes optimal power flow report files

Small Signal Stability Analysis

fm_eigen	eigenvalue computations
fm_eigfig	GUI for eigenvalue computations

Time Domain Simulation

fm_int	time domain simulation
fm_tstep	definition of time step for transient computations
fm_out	time domain simulation output
fm_snap	GUI for snapshot settings

User Defined Model Construction

fm_build	compile user defined components
fm_comp	general settings and utilities for component definition
fm_open	open user defined models
fm_save	save user defined models
fm_new	reset user defined component variables
fm_add	add user defined model variable
fm_del	delete user defined model variable
fm_install	install user defined component
fm_uninstall	uninstall user defined component
fm_component	GUI for user defined models
fm_make	GUI for user defined component definition
fm_update	GUI for displaying user defined model installation results
fm_cset	GUI for component settings
fm_xset	GUI for state variable settings
fm_sset	GUI for service variable settings
fm_pset	GUI for parameter variable settings

Utilities Functions

<code>autorun</code>	secure routine launch
<code>fm_idx</code>	definition of variable names
<code>fm_iidx</code>	find bus interconnetcions
<code>fm_errv</code>	check component voltage rating
<code>fm_filenum</code>	enumeration of output files
<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_rmgen</code>	find and remove static generators connected to a bus
<code>fm_setgy</code>	delete row and columns in power flow Jacobian G_y
<code>fm_status</code>	display convergence error status on main GUI
<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 <i>.m</i> 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_lib</code>	SIMULINK library
<code>fm_simrep</code>	power flow report for SIMULINK models
<code>fm_simset</code>	GUI for SIMULINK model settings
<code>fm_simsave</code>	save a SIMULINK 5, 4.1 or 4 model as a SIMULINK 3 model
<code>fm_block</code>	set SIMULINK block parameters
<code>fm_inout</code>	create and delete SIMULINK block input/output ports
<code>fm_draw</code>	draw SIMULINK block icons

Data File Conversion

<code>fm_dir</code>	browser for data conversion
<code>fm_dirset</code>	utilities for data conversion
<code>filters/chapman2psat</code>	Chapman to PSAT filter (perl file)
<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/matpower2psat</code>	Matpower to PSAT filter (m-file)
<code>filters/neplan2psat</code>	NEPLAN to PSAT filter (perl file)
<code>filters/pcflo2psat</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/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>	Tsinghua 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_plot</code>	general function for plotting results
<code>fm_plotfig</code>	GUI for plotting results
<code>fm_axesdlg</code>	GUI for axes properties settings
<code>fm_linedlg</code>	GUI for line properties settings
<code>fm_linelist</code>	GUI for line list browser
<code>fm_view</code>	general function for sparse matrix visualization
<code>fm_matrx</code>	GUI for sparse matrix visualization
<code>fm_bar</code>	plots status bar on main window

Command History

<code>fm_text</code>	command history general functions and utilities
<code>fm_hist</code>	GUI for command history visualization
<code>fm_disp</code>	command, message and error display
<code>fval</code>	message line for variable manipulation

Output

<code>fm_write</code>	call function for writing output results
<code>fm_writexls</code>	write output results in HTML format
<code>fm_writetex</code>	write output results in \LaTeX format
<code>fm_writetxt</code>	write output results in plain text
<code>fm_writexls</code>	write output results in Excel format

Themes

<code>fm_theme</code>	theme manager
<code>fm_themefig</code>	GUI of theme manager
<code>fm_mat</code>	background for GUI images

Other GUI Utilities

<code>fm_setting</code>	GUI for general settings
<code>fm_enter</code>	welcome GUI
<code>fm_tviewer</code>	GUI for text viewer selection
<code>fm_about</code>	about PSAT
<code>fm_iview</code>	image viewer
<code>fm_author</code>	author's pic
<code>fm_clock</code>	analogic watch
<code>fm_choice</code>	dialog box

GNU License Functions

<code>gnulicense</code>	type the GNU-GPL
<code>fm_license</code>	GUI for the GNU-GPL
<code>gnuwarranty</code>	type the “no warranty” conditions
<code>fm_warranty</code>	GUI for the “no warranty” conditions

PMU Placement Functions

<code>fm_pmiloc</code>	PMU placement manager
<code>fm_pmunl</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_lssest</code>	linear static state estimation
<code>fm_spantree</code>	spanning tree of existing PMUs
<code>fm_mintree</code>	minimum tree search
<code>fm_annealing</code>	annealing method for PMU placement
<code>fm_pmufig</code>	GUI for PMU placement

Command Line Usage

<code>initpsat</code>	initialize PSAT global variables
<code>closepsat</code>	clear all PSAT global variables from workspace
<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_uwpflow</code>	UWPFLOW interface
<code>fm_uwfig</code>	GUI of the UWPFLOW interface

Numeric Linear Analysis Functions

<code>fex_abcd</code>	compute numeric matrices A , B , C , D
-----------------------	--

Appendix C

Other Files and Folders

This appendix lists the files other than MATLAB functions and scripts which are contained in the PSAT folder and the auxiliary folders needed by PSAT to work properly. The names and the positions of these folders can be changed only if the path defined in the `psat` script file is accordingly changed. In the distribution tarball these folders are placed within the PSAT main folder.

.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:

function name	cols. 1-23
structure name	cols. 25-44
call algebraic equations	col. 46
call algebraic Jacobians	col. 48
call differential equations	col. 50
call state Jacobians	col. 52
call hard limits	col. 54
call during power flow	col. 56
call initialization	col. 58
call if computing shunt powers	col. 60
call if computing series flows	col. 62

history settings for the command history. The file is updated each time the command history settings are saved.

namevarx definition of state variables names, formatted names in a \LaTeX synthax 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.

.mat Files

finger matrix defining a custom mouse pointer.

.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/neplan2psat` filter for the Neplan data format.

`filters/pcflo2psat` filter for the PCFLOH 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/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.

`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

There are a few files I modified from the original version provided within the MATLAB package:

`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`

`isvarname.m`

The free utility `uigetfolder`¹ written by Neil Rutland is no longer used, since it has been substituted by the built-in function `uigetdir`. When using MATLAB releases < R13, calls to `uigetdir` are disabled.

¹`uigetfolder` is available at www.mathworks.com in the File Exchange section.

Appendix E

Power System Softwares

This appendix lists a selection of websites of power system software packages. The list can be incomplete and some links can be broken. Any help in maintaining this list as complete and updated as possible is greatly appreciated.

ANA's Softwares	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.cyme.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_pslf/index.htm
INTELICON	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
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.scscscc-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 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

Appendix F

Test System Data

This appendix depicts schemes and 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 (ESCO) 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 1 1;
3 400 1 0 1 1];

Line.con = [ ...
1 2 100 400 60 0 0 0 0.1 0 0 0 0.4 0.4 0;
1 3 100 400 60 0 0 0 0.1 0 0 0 0.4 0.4 0;
2 3 100 400 60 0 0 0 0.1 0 0 0 0.4 0.4 0];

SW.con = [ ...
1 100 400 1 0 1.5 -1.5 1.1 0.9 0.4 1];

PV.con = [ ...
2 100 400 0.4 1 0.8 -0.2 1.1 0.9 1;
3 100 400 0.4 1 0.8 -0.2 1.1 0.9 1];

PQ.con = [ ...
3 100 400 1 0.6 1.2 0.8 1];

Demand.con = [ ...
3 100 1 0.6 1 1 0 0 0 0 0 0 0 0 0];
```

¹The SIMULINK models are placed in the folder `tests` within the PSAT main folder.

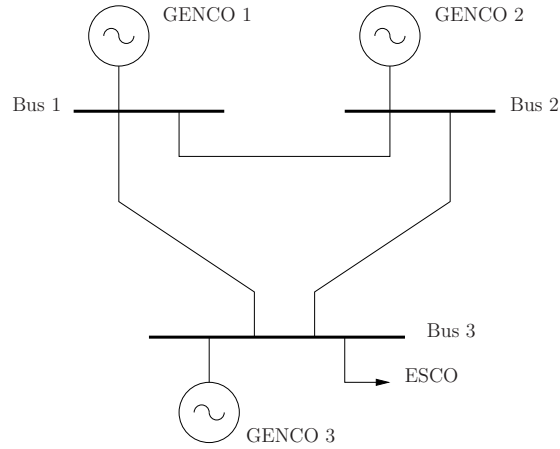


Figure F.1: 3-bus test system.

```
Supply.con = [ ...
1 100 0 0.6 0.1 0 6 9.8 0.1 0 0 0 0 0;
2 100 0 0.6 0.1 0 4 10.7 0.2 0 0 0 0 0;
3 100 0 0.6 0.1 0 8 12.6 0.25 0 0 0 0 0];
```

```
Rmpg.con = [ ...
2 100 0.1 0.1 2 2 5 0 1;
1 100 0.05 0.05 2 2 5 0 1;
3 100 0.15 0.15 2 2 0 5 1];
```

```
Ypdp.con = [ ...
55 75 100 120 100];
```

```
Varname.bus = {...
'Bus1'; 'Bus2'; 'Bus3'};
```

F.2 6-bus Test System

Figure F.2 depicts the 6-bus test case, which is extracted from [105], 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;
2 400 1 0;
3 400 1 0;
4 400 1 0;
5 400 1 0;
6 400 1 0];
```

```

Line.con = [ ...
2   3   100   400   60 0 0 0.05 0.25 0.06 0 0 0.3082;
3   6   100   400   60 0 0 0.02 0.1 0.02 0 0 1.3973;
4   5   100   400   60 0 0 0.2 0.4 0.08 0 0 0.1796;
3   5   100   400   60 0 0 0.12 0.26 0.05 0 0 0.6585;
5   6   100   400   60 0 0 0.1 0.3 0.06 0 0 0.2000;
2   4   100   400   60 0 0 0.05 0.1 0.02 0 0 1.3740;
1   2   100   400   60 0 0 0.1 0.2 0.04 0 0 0.2591;
1   4   100   400   60 0 0 0.05 0.2 0.04 0 0 0.9193;
1   5   100   400   60 0 0 0.08 0.3 0.06 0 0 0.8478;
2   6   100   400   60 0 0 0.07 0.2 0.05 0 0 0.9147;
2   5   100   400   60 0 0 0.1 0.3 0.04 0 0 0.7114];

SW.con = [ ...
2      100      400      1.05  0  1.5  -1.5  1.1  0.9  1.4  1];

PV.con = [ ...
1      100      400      0.9  1.05  1.5  -1.5  1.1  0.9  1;
3      100      400      0.6  1.05  1.5  -1.5  1.1  0.9  1];

PQ.con = [ ...
4      100      400      0.9  0.6  1.1  0.9  0;
5      100      400      1  0.7  1.1  0.9  0;
6      100      400      0.9  0.6  1.1  0.9  0 ];

Demand.con = [ ...
4      100      0.25 0.16665 0.25 1e-05 0 0 12 0 0 0 0 1;
5      100      0.1 0.07 0.1 1e-05 0 0 10.5 0 0 0 0 1;
6      100      0.2 0.13333 0.2 1e-05 0 0 9.5 0 0 0 0 1];

Supply.con = [ ...
1      100      0.2 0.2 1e-05 0 0 9.7 0 0 0 0 1;
2      100      0.25 0.25 1e-05 0 0 8.8 0 0 0 0 1;
3      100      0.2 0.2 1e-05 0 0 7 0 0 0 0 1];

Varname.bus = {...
'Bus 1'; 'Bus 2'; 'Bus 3'; 'Bus 4'; 'Bus 5'; 'Bus 6'};

```

F.3 9-bus Test System

Figure F.3 depicts the 9-bus test system, which is extracted from [101] 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;
2  18    1  0;
3  13.8  1  0;
4  230   1  0;
5  230   1  0;
6  230   1  0;
7  230   1  0;
8  230   1  0;
9  230   1  0 ];

```

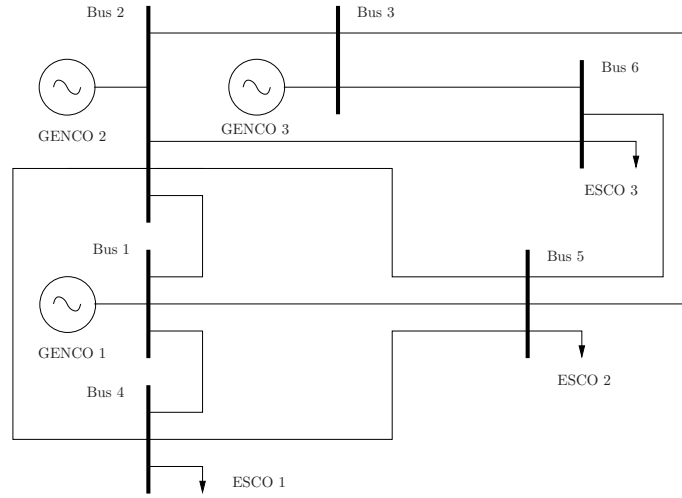


Figure F.2: 6-bus test system.

```

Line.con = [ ...
9  8  100  230  60  0  0      0.0119  0.1008  0.209  0  0  0;
7  8  100  230  60  0  0      0.0085  0.072  0.149  0  0  0;
9  6  100  230  60  0  0      0.039  0.17  0.358  0  0  0;
7  5  100  230  60  0  0      0.032  0.161  0.306  0  0  0;
5  4  100  230  60  0  0      0.01  0.085  0.176  0  0  0;
6  4  100  230  60  0  0      0.017  0.092  0.158  0  0  0;
2  7  100  18  60  0  0.0782609  0  0.0625  0  0  0  0;
3  9  100  13.8  60  0  0.06  0  0.0586  0  0  0  0;
1  4  100  16.5  60  0  0.0717391  0  0.0576  0  0  0  0 ];

```

```

SW.con = [ ...
1  100  16.5  1.04  0  99 -99  1.1  0.9  0.8  1 ];

```

```

PV.con = [ ...
2  100  18  1.63  1.025  99 -99  1.1  0.9  1;
3  100  13.8  0.85  1.025  99 -99  1.1  0.9  1 ];

```

```

PQ.con = [ ...
8  100  230  1  0.35  1.2  0.8  0;
5  100  230  1.25  0.5  1.2  0.8  0;
6  100  230  0.9  0.3  1.2  0.8  0 ];

```

```

Syn.con = [ ...
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;
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;
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 ];

```

```

Exc.con = [ ...
3 2 5 -5 20 0.2 0.063 0.35 0.01 0.314 0.001 0.0039 1.555;
1 2 5 -5 20 0.2 0.063 0.35 0.01 0.314 0.001 0.0039 1.555;
2 2 5 -5 20 0.2 0.063 0.35 0.01 0.314 0.001 0.0039 1.555 ];

```

```

Varname.bus = {...
'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 AVRs. 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;
7 8 100 230 60 0 0 0.0085 0.072 0.149 0 0 0 0 0;
9 6 100 230 60 0 0 0.039 0.17 0.358 0 0 0 0 0;
7 5 100 230 60 0 0 0.032 0.161 0.306 0 0 0 0 0;
5 4 100 230 60 0 0 0.01 0.085 0.176 0 0 0 0 0;
6 4 100 230 60 0 0 0.017 0.092 0.158 0 0 0 0 0;
2 7 100 18 60 0 0.078 0 0.0625 0 0 0 0 0;
3 9 100 13.8 60 0 0.06 0 0.0586 0 0 0 0 0;
1 4 100 16.5 60 0 0.072 0 0.0576 0 0 0 0 0 ];

```

```

Breaker.con = [ ...
4 7 100 230 60 1 1.083 4 ];

```

```

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 ];

```

```

PV.con = [ ...
2 100 18 1.63 1.025 99 -99 1.1 0.9 1;
3 100 13.8 0.85 1.025 99 -99 1.1 0.9 1 ];

```

```

PQ.con = [ ...
6 100 230 0.9 0.3 1.2 0.8 0;
8 100 230 1 0.35 1.2 0.8 0;
5 100 230 1.25 0.5 1.2 0.8 0 ];

```

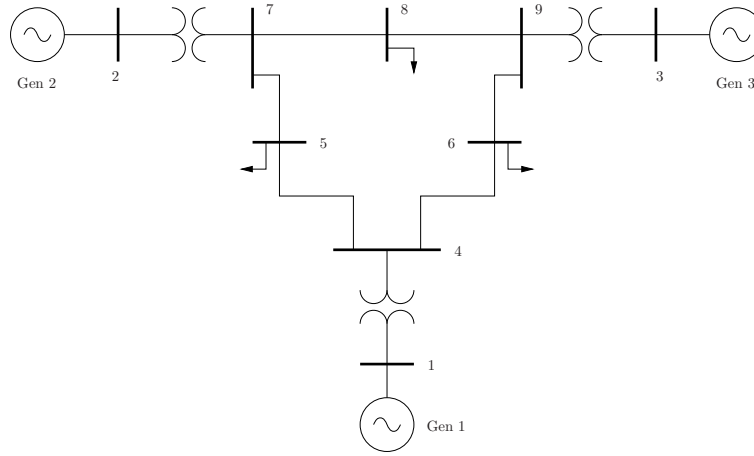


Figure F.3: WSCC 3-generator 9-bus test system.

```
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;
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;
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 ];

Varname.bus = {...
'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;
2 69 1 0;
3 69 1 0;
4 69 1 0;
5 69 1 0;
6 13.8 1 0;
7 13.8 1 0;
8 18 1 0;
9 13.8 1 0;
10 13.8 1 0;
```

²Available at <http://www.ee.washington.edu/research/pstca/>.


```

11 13.8 1 0;
12 13.8 1 0;
13 13.8 1 0;
14 13.8 1 0 ];

Line.con = [ ...
2 5 100 69 60 0 0 0.05695 0.17388 0.034 0 0 0;
6 12 100 13.8 60 0 0 0.12291 0.25581 0 0 0 0;
12 13 100 13.8 60 0 0 0.22092 0.19988 0 0 0 0;
6 13 100 13.8 60 0 0 0.06615 0.13027 0 0 0 0;
6 11 100 13.8 60 0 0 0.09498 0.1989 0 0 0 0;
11 10 100 13.8 60 0 0 0.08205 0.19207 0 0 0 0;
9 10 100 13.8 60 0 0 0.03181 0.0845 0 0 0 0;
9 14 100 13.8 60 0 0 0.12711 0.27038 0 0 0 0;
14 13 100 13.8 60 0 0 0.17093 0.34802 0 0 0 0;
7 9 100 13.8 60 0 0 0 0.11001 0 0 0 0;
1 2 100 69 60 0 0 0.01938 0.05917 0.0528 0 0 0;
3 2 100 69 60 0 0 0.04699 0.19797 0.0438 0 0 0;
3 4 100 69 60 0 0 0.06701 0.17103 0.0346 0 0 0;
1 5 100 69 60 0 0 0.05403 0.22304 0.0492 0 0 0;
5 4 100 69 60 0 0 0.01335 0.04211 0.0128 0 0 0;
2 4 100 69 60 0 0 0.05811 0.17632 0.0374 0 0 0;
5 6 100 69 60 0 5 0 0.25202 0 0.932 0 0;
4 9 100 69 60 0 5 0 0.55618 0 0.969 0 0;
4 7 100 69 60 0 5 0 0.20912 0 0.978 0 0;
8 7 100 18 60 0 1.3043 0 0.17615 0 0 0 0 ];

SW.con = [ ...
1 100 69 1.06 0 9.9 -9.9 1.2 0.8 2.324 1 ];

PV.con = [ ...
2 100 69 0.4 1.045 0.5 -0.4 1.2 0.8 1;
6 100 13.8 0 1.07 0.24 -0.06 1.2 0.8 1;
3 100 69 0 1.01 0.4 0 1.2 0.8 1;
8 100 18 0 1.09 0.24 -0.06 1.2 0.8 1 ];

PQ.con = [ ...
2 100 69 0.217 0.127 1.2 0.8 0;
3 100 69 0.942 0.19 1.2 0.8 0;
14 100 13.8 0.149 0.05 1.2 0.8 0;
4 100 69 0.478 0.04 1.2 0.8 0;
5 100 69 0.076 0.016 1.2 0.8 0;
9 100 13.8 0.295 0.166 1.2 0.8 0;
6 100 13.8 0.112 0.075 1.2 0.8 0;
10 100 13.8 0.09 0.058 1.2 0.8 0;
13 100 13.8 0.135 0.058 1.2 0.8 0;
12 100 13.8 0.061 0.016 1.2 0.8 0;
11 100 13.8 0.035 0.018 1.2 0.8 0 ];

Varname.bus = {...
'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'};

```

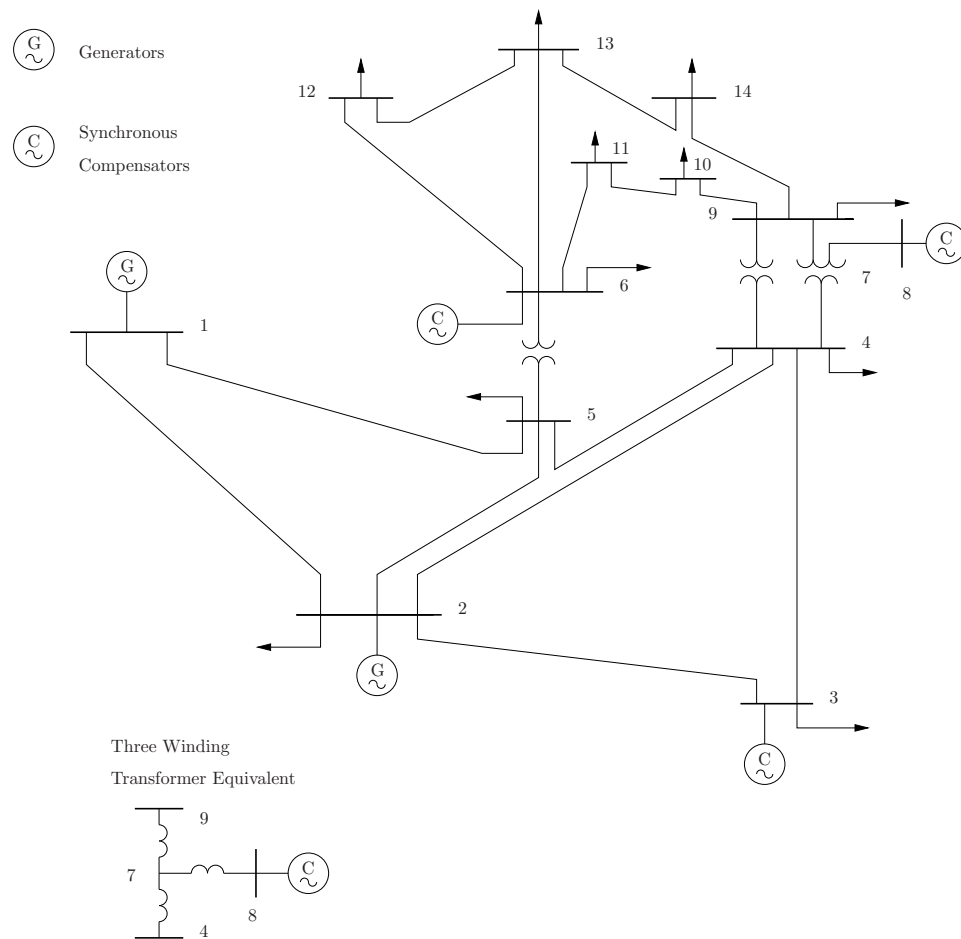


Figure F.4: IEEE 14-bus test system.

Appendix G

FAQs

This appendix presents the most frequent asked questions related to PSAT installation and usage. Following FAQs were mostly arisen 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.

I have Matlab version older than 6.1 and when I try to run the program the following error shows up:

```
>> psat
C:\MATLABR11
??? C:\MATLABR11\toolbox\psat\fm_enter.p is a P-file written by a newer
version MATLAB and cannot be read.

Error in ==> C:\MATLABR11\toolbox\psat\psat.m
On line 24 ==> hdl = fm_enter(a(1).path);
```

PSAT version 1.0.1 files were written and pre-compiled using MATLAB 6.1 R12 and will not run under MATLAB 6.0 or older. PSAT version 2.0.0 was written using mainly MATLAB 6.5 R13 and afterwards tested on MATLAB 5.3, 6.0 and 6.1. Pre-compiled files (p-code) were built using MATLAB 5.3, which ensures the compatibility with newer MATLAB versions.

However, less commonly used PSAT functions can still contain calls to built-in functions which were not present in MATLAB versions older than 6.5. Please report all inconsistencies in order to fix these bugs.

Because of the compatibility issue, some of the latest features of the current MATLAB release 6.5 are not used or, when used, are disabled when PSAT runs under older MATLAB versions. This is the case of some built-in functions (e.g. `uigetdir`) and Perl modules.³ Furthermore, the interfaces with GAMS and UWPFLOW programs can be used only with MATLAB 6.5.

PSAT distribution comes in p-code files, how can I get source m-files?

Since version 1.3.0, PSAT comes in open source files. PSAT is also free software (see the GNU General Public License which is reported in Appendix K).

However, p-code (pre-compiled) files run faster on some platforms and the most of the users do not need to change the code on a daily basis. Thus, PSAT includes a small GUI to create one's own PSAT distribution as p-code files. Refer to Section 26.5 for details.

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 electromagnetical transient studies (including detailed models of power electronic

¹The MATLAB current path is returned by the `pwd` function.

²The MATLAB search path is returned by the `path` function.

³Perl filters for data file conversion can be used only with MATLAB 6.5. Older MATLAB files such as `fm_cdf.m` are still included in the PSAT distribution but will be no longer maintained.

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, SimPowerSystems has a longer history and has been written by a team of people. Thus, SimPowerSystems should be generally more reliable than PSAT.

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 27 for a detailed documentation about the command line usage of PSAT.

Can I run PSAT on Octave?

As for version 1.3.0, PSAT can run on GNU Octave. Restrictions and limitations apply; see Chapter 28.

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. If you want to add a new component or a new control scheme refer to Chapter 25 which describes how to build user defined models under PSAT.

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 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 analyses 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(:, [3 4]) = 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(:, [3 4]) = [ ...
```

```

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.

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. Thus, 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. However, please report all inconsistencies to me in order to improve the filters.

G.7 Interfaces

I have installed the demo version of GAMS 21.1 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 succesfully and output files (expected by `fm_gams.m`) were not created.

I have done all steps indicated in Chapter 29, but the PSAT-GAMS interface is still not working.

First, please make sure that you have done **all** the appropriate steps indicated in Chapter 29 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 startup 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	<code>psatforum-subscribe@yahoogroups.com</code>
Post message	<code>psatforum@yahoogroups.com</code>
Unsubscribe	<code>psatforum-unsubscribe@yahoogroups.com</code>
List owner	<code>psatforum-owner@yahoogroups.com</code>

To post a message directly to me, use one of the following e-mails:

1. `Federico.Milano@uclm.es`
2. `fmilano@thunderbox.uwaterloo.ca`
3. `psatforum@yahoo.com`

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](#)

Search:

Web Search

[Sign In](#)
[New User?](#)
[Sign Up](#)

[Tech](#)
[Groups](#)
[Help](#)

[psatforum](#) · PSAT Forum

[Home](#)

[Members Only](#)
[Messages](#)
[Post](#)
[Files](#)
[Photos](#)
[Links](#)
[Database](#)
[Polls](#)
[Members](#)
[Calendar](#)
[Promote](#)

[Info](#)
[Settings](#)

Group Information

Members: 1138

Category: [Software](#)

Founded: Aug 16, 2003

Language: English

Already a member?
[Sign in to Yahoo!](#)

Yahoo! Groups Tips

Did you know...

Message search is now enhanced, find messages faster. Take it for a spin.

Yahoo! 360°

Share your life through photos, blogs, more.

Home

Join This Group!

Activity within 7 days:
4 New Members - 62 New Messages - 1 New File

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
2007	163	146										
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

Group Email Addresses

Related Link: <http://www.power.uwaterloo.ca/~fmlano/>

Post message: psatforum@yahoogroups.com

Subscribe: psatforum-subscribe@yahoogroups.com

Unsubscribe: psatforum-unsubscribe@yahoogroups.com

List owner: psatforum-owner@yahoogroups.com

Join This Group!

YAHOO! SPONSOR RESULTS

Computer Software - Web enabled ERP software for manufacturing, retail, construction, food, mining, FMCG, inventory, engineering and others. We change the way you do business - forever.
www.ascorp.com

Computer Monitoring Spy Software - Visual TimeAnalyzer automatically tracks all computer usage and presents detailed, richly illustrated reports. Track work time, pauses, projects, costs, software and Internet use.
www.neuber.com

Figure H.1: PSAT Forum main page. Data refer to February 23, 2007.

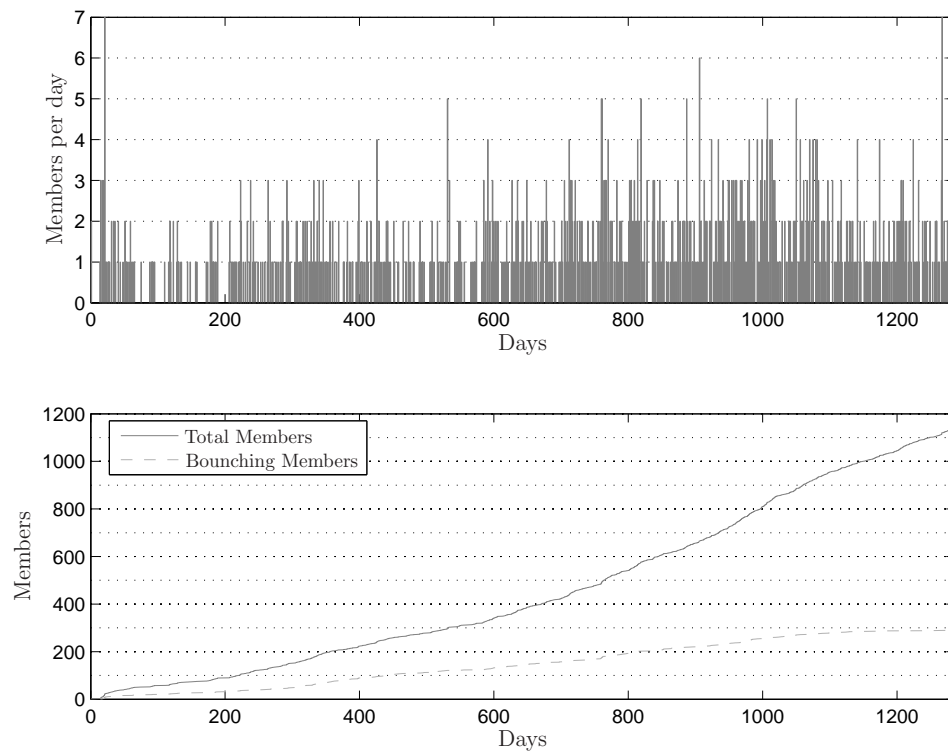


Figure H.2: PSAT Forum statistics. Data refer to February 23, 2007.

Appendix I

Citations & Links

A list of 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 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] 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.
- [3] S. V. N. Jithin-Sundar and M. Reshmi, “Utilization of Controlled Shunt Reactor in a 400 kV Interconnected Network,” *International Journal of Emerging Electric Power Systems*, vol. 2, no. 1, 2005.
- [4] 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.
- [5] F. Milano, “An Open Source Power System Analysis Toolbox,” *IEEE Transactions on Power Systems*, vol. 20, no. 3, pp. 1199–1206, Aug. 2005.

- [6] 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.

I.3 Conference Proceedings

- [1] H. B. Çetinkaya, S. Öztürk, and B. Alboyacı, “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] 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.
- [6] 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.
- [7] 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.
- [8] 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.

I.4 Webpages

☞ 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/UsefullLinks.htm`

- ☞ Webpage of Warren King, University of Waterloo, Canada, available at:

`www.power.uwaterloo.ca/~ewking/`

- ☞ Webpage on Open-Source Software for Electricity Market Research, Teaching, and Training, by Leigh Tesfatsion, 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/~vanfrrl/psat.html/`

- ☞ Webpage of Electrical and Computer Engineering, University of Alberta, Canada:

`www.ece.ualberta.ca/~ee433/`

- ☞ Webpage of Sheng How Goh, University of Queensland, Australia, available at:

`www.itee.uq.edu.au/~shgoh/`

- ☞ Webpage of *Moisés* Roberto Lanner Carvalho, Instituto Militar de Engenharia, Brasil, available at:

`aquarius.ime.eb.br/~mrlc/`

- ☞ Webpage 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.

Appendix K

The GNU General Public License

Version 2, June 1991

Copyright © 1989, 1991 Free Software Foundation, Inc.

59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) 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 this service 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 make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. 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.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute 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 and disclaimer of warranty; keep intact all the notices that refer to this License and to the

absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:
 - (a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
 - (b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
 - (c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- (a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- (b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- (c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

- 4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.
- 5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.
- 6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy,

distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), 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 distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
9. The Free Software Foundation may publish revised and/or new versions of the 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 a version number of this License which applies to it and “any later version”, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the

Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, 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.
12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE 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.

END OF TERMS AND CONDITIONS

Appendix: 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 convey 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) <year> <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 2 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, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) <year> <name of author>
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details
type 'show w'.
This is free software, and you are welcome to redistribute it under cer-
tain 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, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items—whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a “copyright disclaimer” for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James
Hacker.
```

```
<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice
```

This 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 Library General Public License instead of this License.

Appendix L

GNU Free Documentation License

Version 1.1, March 2000

Copyright (C) 2000 Free Software Foundation, Inc.

59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document “free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License.

The “Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “you.”

A “Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A “Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not “Transparent” is called “Opaque.”

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license

notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section entitled “Endorsements.” Such a section may not be included in the Modified Version.

N. Do not retitle any existing section as “Endorsements” or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice. These titles must be distinct from any other section titles.

You may add a section entitled “Endorsements”, provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled “History” in the various original documents, forming one section entitled “History”; likewise combine any sections entitled “Acknowledgements”, and any sections entitled “Dedications.” You must delete all sections entitled “Endorsements.”

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an “aggregate”, and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document’s Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation 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. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

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, the documentation is available at <http://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] H. B. Çetinkaya, S. Öztürk, and B. Alboyacı, "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.
- [29] —, "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.
- [30] S. J. Chapman, *Electric Machinery and Power System Fundamentals*. New York: McGraw Hill, 2002.
- [31] 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.
- [32] J. Chow, *Power System Toolbox Version 2.0: Dynamic Tutorial and Functions*, Cherry Tree Scientific Software, RR-5 Colborne, Ontario K0K 1S0, 1991-1997.
- [33] —, *Power System Toolbox Version 2.0: Load Flow Tutorial and Functions*, Cherry Tree Scientific Software, RR-5 Colborne, Ontario K0K 1S0, 1991-1999.
- [34] 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.
- [35] 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.

- [36] 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.
- [37] 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.
- [38] —, "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.
- [39] 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.
- [40] —, "Multiperiod Auction for a Pool-based electricity market," *IEEE Transactions on Power Systems*, vol. 17, no. 4, pp. 1225–1231, Nov. 2002.
- [41] 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.
- [42] 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.
- [43] 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/>.
- [44] J. W. Eaton, *GNU Octave Manual*. Bristol, UK: Network Theory Ltd., 1997.
- [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] M. Ilić and J. Zaborszky, *Dynamic and Control of Large Electric Power Systems*. New York: Wiley-Interscience Publication, 2000.
- [60] 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.
- [61] 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.

- [62] S. V. N. Jithin-Sundar and M. Reshmi, "Utilization of Controlled Shunt Reactor in a 400 kV Interconnected Network," *International Journal of Emerging Electric Power Systems*, vol. 2, no. 1, 2005.
- [63] 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.
- [64] 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.
- [65] 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.
- [66] 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.
- [67] 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.
- [68] P. Kundur, *Power System Stability and Control*. New York: McGraw Hill, 1994.
- [69] 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.
- [70] M. Madrigal, "Optimization Model and Techniques for Implementation and Pricing of Electricity Markets," Ph.D. dissertation, University of Waterloo, Waterloo, ON, Canada, 2000.
- [71] 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.
- [72] 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.
- [73] 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.

- [74] 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.
- [75] F. Milano, "Pricing System Security in Electricity Market Models with Inclusion of Voltage Stability Constraints," Ph.D. dissertation, University of Genova, Genova, Italy, 2003, available at <http://thundebox.uwaterloo.ca/~fmilano>.
- [76] —, "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.
- [77] —, "An Open Source Power System Analysis Toolbox," *IEEE Transactions on Power Systems*, vol. 20, no. 3, pp. 1199–1206, Aug. 2005.
- [78] 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.
- [79] 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.
- [80] J. Miliadis-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.
- [81] —, "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.
- [82] 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.
- [83] 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.
- [84] 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.
- [85] A. Monticelli and A. Garcia, "Fast Decoupled State Estimators," *IEEE Transactions on Power Systems*, vol. 5, no. 2, pp. 556–564, May 1990.
- [86] 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.

- [87] 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, the documentation is available at <http://www.gams.com/>.
- [88] 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.
- [89] 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.
- [90] C. Nwankpa, *Voltage Stability Toolbox, version 2*, Center for Electric Power Engineering, Drexel University, 2002, available at <http://power.ece.drexel.edu/research/VST/vst.htm>.
- [91] 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.
- [92] H. A. Panofsky and J. A. Dutton, *Atmospheric Turbulence; Models and Methods for Engineering Applications*. New York: John Wiley & Sons, 1984.
- [93] 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.
- [94] 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.
- [95] A. H. M. A. Rahim, S. A. Al-Baiyat, and H. M. Al-Maghrabi, "Robust damping controller design for a static compensator," *IEE Proceedings on Generation, Transmission and Distribution*, vol. 149, pp. 491–496, July 2002.
- [96] I. C. Report, "Reader's Guide to SSR," *IEEE Transactions on Power Apparatus and Systems*, vol. 7, no. 2, pp. 150–157, Feb. 1992.
- [97] 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.
- [98] 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.
- [99] N. Rostamkolai, C. A. Wegner, R. J. Piwko, H. Elahi, M. A. Eitzmann, G. Garzi, and P. Tietz, "Control Design of Santo Tomé Back-to-Back HVDC Link," *IEEE Transactions on Power Systems*, vol. 8, no. 3, pp. 1250–1256, Aug. 1993.

- [100] 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.
- [101] P. W. Sauer and M. A. Pai, *Power System Dynamics and Stability*. Upper Saddle River, New Jersey: Prentice Hall, 1998.
- [102] F. Schweppe, 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.
- [103] 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.
- [104] R. Seydel, *Practical Bifurcation and Stability Analysis: From Equilibrium to Chaos*. New York: Second Edition, Springer-Verlag, 1994.
- [105] G. B. Sheblé, *Computational Auction Mechanism for Restructured Power Industry Operation*. Boston: Kluwer Academic Publishers, 1998.
- [106] T. Shome, A. M. Gole, D. P. Brandt, and R. J. Hamlin, "Adjusting Converter Control for Paralled DC Converters Using a Digital Transient Simulation Program," *IEEE Transactions on Power Systems*, vol. 5, no. 1, pp. 12–19, Feb. 1990.
- [107] E. Simiu and R. H. Scanlan, *Wind Effects on Structures; an Introduction to Wind Engineering*. New York: John Wiley & Sons, 1986.
- [108] J. G. Slootweg, "Wind Power: Modelling and Impact on Power System Dynamics," Ph.D. dissertation, Delft University of Technology, Delft, Netherlands, 2003.
- [109] Y. H. Song and A. T. Johns, *Flexible AC Transmission System (FACTS)*. London: The Institute of Electrical Engineers, 1999.
- [110] B. Stott, "Review of load-Flow Calculation Methods," *Proceedings of the IEEE*, vol. 62, no. 7, pp. 916–929, July 1974.
- [111] 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.
- [112] 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.
- [113] G. Sybille, *SimPowerSystems User's Guide, Version 4*, published under sublicense from Hydro-Québec, and The MathWorks, Inc., Oct. 2004, available at <http://www.mathworks.com>.

- [114] 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.
- [115] 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.
- [116] 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.
- [117] G. L. Torres and V. H. Quintana, "Introduction to Interior-Point Methods," in *IEEE PICA*, Santa Clara, CA, May 1999.
- [118] 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.
- [119] 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.
- [120] 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.
- [121] 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.
- [122] 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.
- [123] 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.
- [124] 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.
- [125] Working Group H-7 of the Relaying Channels Subcommittee of the IEEE Power System Relaying Committee, "Synchronoized Sampling and Phasor Measurements for Relaying and Contorl," *IEEE Transactions on Power Delivery*, vol. 9, no. 1, pp. 442–452, Jan. 1994.

- [126] 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.
- [127] 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.
- [128] W. Xu and Y. Mansour, "Voltage Stability Analysis Using Generic Dynamic Load Models," *IEEE Transactions on Power Systems*, vol. 9, no. 1, Feb. 1994.
- [129] Yao-Nan-Yu, *Electric Power System Dynamic*. New York: Academic Press, 1983.
- [130] 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.
- [131] 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.
- [132] 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

A

Area, 19, 103, **114–115**, 368
Asynchronous machine, *see* Induction motor
Automatic voltage regulator, *see* AVR
AVR, 4, 85, 147, 154, 165, **169–171**, 265, 268, 391
 type I, 169
 type II, 170
 type III, 171

B

Bus, **103–104**, 257, 258, 260

C

CAC, 4, 165, 180, **181**, 265
CC, 4, 165, 180, **181**, 265
Central Area Controller, *see* CAC
Cluster Controller, *see* CC
Constant power generator, *see* PQ generator
Constant power load, *see* PQ load
Constant Speed Wind Turbines, *see* CSWT
Continuation Power Flow, *see* CPF
CPF, 3, 6, 15, 17, 20, 22, 23, 39, 40, **41–48**, 105, 117, 245, 248, 303–305, 331, 332, 334, 337, 338, 361, 368, 376
CSWT, 4, **219–221**
CygWin, 309, 331, 332

D

DDSG, 4, **227–229**
Demand, 39, 46, 118, 121–122, 262, 334, 368
Demand profile, 122–124, 368

DFIG, 4, **221–226**

Direct Drive Synchronous Generator, *see* DDSG

Doubly Fed Induction Generator, *see* DFIG

Dynamic shaft, **231–232**, 268

E

Excitation, *see* AVR
Exponential recovery load, 4
Exponential recovery load, 135, **138–140**, 262

F

FACTS, 4, 17, 193, 245, 331, 332, 369
Fast Decoupled Power Flow, *see* FDPF
FDPF, 27, **28–29**, 355
Flexible ac transmission system, *see* FACTS
Frequency dependent load, 4
Frequency dependent load, 135, **137–138**, 262
Frequency regulation, *see* TG
Fuel cell, *see* Solid oxide fuel cell

G

GAMS, 4, 7, 17, 262, 303, 305, **315–324**, 371, 379, 382, 383, 398, 403
GNU Linux, *see* Linux
GNU Octave, *see* Octave

H

High voltage dc transmission system, *see* HVDC
HVDC, 4, 17, 193, **209–211**, 331, 332, 369

I

IEEE, 16, 46, 48, 70, 84, 169, 255, 283, 334, 382, 394, 396, 400

Induction motor, 147, **158–163**

double cage, 162

mechanical model, 159

order I, 159

order III, 161

order V, 162

single cage, 161

Infinite bus, *see* Slack generator

Interior Point Method, *see* IPM

IPM, 53

J

Jimma's load, 4, 135, **142–143**, 262

L

Line, *see* Transmission line

Linux, 9, 10, 296, 309, 316, 331, 403

Load tap changer, *see* Tap changer

LTC, *see* Tap Changer

M

Matlab, v, 3, 6, 9–12, 14, 16, 19, 21, 23, 28, 72, 83, 85, 213, 271, 276, 277, 283, 287, 289, 296, 301, 302, 304, 305, 316, 332, 334, 341, 344, 354, 355, 359, 367, 375, 383, 385, 397–400, 403

Merhotra's predictor-corrector, 53, 363

Mixed load, 4

Mixed load, 135, **143–144**, 262

N

Newton direction, 53, 363

Newton-Raphson algorithm, 27, **27–28**, 29, 30, 40, 45, 75, 76, 78, 89, 341, 355

O

Octave, 3, 10, **309–312**, 355, 399

OLTC, *see* Tap Changer

OPF, 3, 6, 15, 17, 22, 23, 39, **53–62**, 105, 117–119, 122, 126, 245, 248, 262, 303–306, 316, 317, 324, 362–364, 368, 370–372, 376, 379, 382, 401

Optimal Power Flow, *see* OPF

Overexcitation limiter, *see* OXL

OXL, 4, 165, **177–180**, 265

P

Phase shifter, *see* Phase shifting transformer

Phase shifting transformer, 4, 185, **189–190**

Phasor Measurement Unit, *see* PMU

PMU, 3, 4, 14, **89–100**, 132–134, 303–305, 357, 365, 368, 379

POD, 165, **183–184**, 193, 369

Power system stabilizer, *see* PSS

PQ generator, **113**

PQ load, 40, 41, 46, **111–112**, 117, 122, 135, 137, 138, 186, 258, 262, 264, 306, 402

Primary freq. regulation, *see* TG

Primary voltage regulation, *see* AVR

PSS, 4, 165, **174–177**, 265

type I, 176

type II, 176

type III, 177

type IV, 177

type V, 177

PST, *see* Phase shifting transformer

PV bus, *see* PV generator

PV generator, 40, 41, **110–111**, 117, 147, 186, 194, 213, 231, 236, 262, 264, 266, 306

R

Region, *see* Area

S

Secondary voltage control, 4, 165, **180–181**, 265, 331, 332

Shaft, *see* Subsynchronous resonance, Dynamic shaft

Shunt, 104, **113–114**

Simulink, 3, 4, 7, 10, 12, 14, 16, 19, 21, 145, 217, 245, 253, 257, 258, 260, 264, 271–279, 302, 306, 309, 355, 357, 370, 377, 389, 399

Slack bus, *see* Slack generator

Slack generator, 104, **108–110**, 156, 158,
231, 262, 264, 306
Solid oxide fuel cell, **236–241**, 268
SSCL, 4
SSSC, 4, 193, **201–202**, 369
STATCOM, 266
Statcom, 4, 193, **198–200**, 369
Static Compensator, *see* Statcom
Static Synchronous Series Compensator,
see SSSC
Static VAr compensator, *see* SVC
Subsynchronous resonance, **233–236**
Supplementary Stabilizing Control Loop,
see POD
Supply, 39, 46, 117–119, 262, 334, 368
SVC, 4, 193, **194–196**, 265, 266, 369
Swing bus, *see* Slack generator
Synchronous machine, 4, 20, 21, 27, 127,
147–158, 165, 169, 174, 179,
231, 264, 343, 344
electromechanical model, 154
order II, 154
order III, 154
order IV, 154
order V, type 1, 155
order V, type 2, 155
order V, type 3, 156
order VI, 157
order VIII, 157

T

Tap changer, 4, 266
dynamic model, **185–186**
with embedded load, **186–189**
TCSC, 4, 193, **196–198**, 369
TCUL, *see* Tap Changer
TG, 4, 147, 165, **165–168**, 265
type I, 166
type II, 168
Thermostatically controlled load, 4, 135,
140–141, 262
Thyristor Controlled Series Compensa-
tor, *see* TCSC
Transformer, **105–108**
Transmission line, **104–105**
Turbine governor, *see* TG

U

ULTC, *see* Tap Changer
Unified PF controller, *see* UPFC
Unix, 10, 296, 331
UPFC, 4, 193, **204–205**, 369
UWPFLOW, 4, 7, 17, 303, 305, **331–**
334, 358, 372, 379, 398

V

Voltage dependent load, 4
Voltage dependent load, 135, **135–136**,
262
Voltage regulation, *see* AVR
Voltage sourced inverter, *see* VSI
VSI, 193

W

Wind, 4, **213–229**, 245, 369
Wind model, **213–217**
Windows, 9, 10, 296, 309, 315, 316, 331,
332, 403

Z

ZIP load, 4, 135, **136–137**, 262, 264