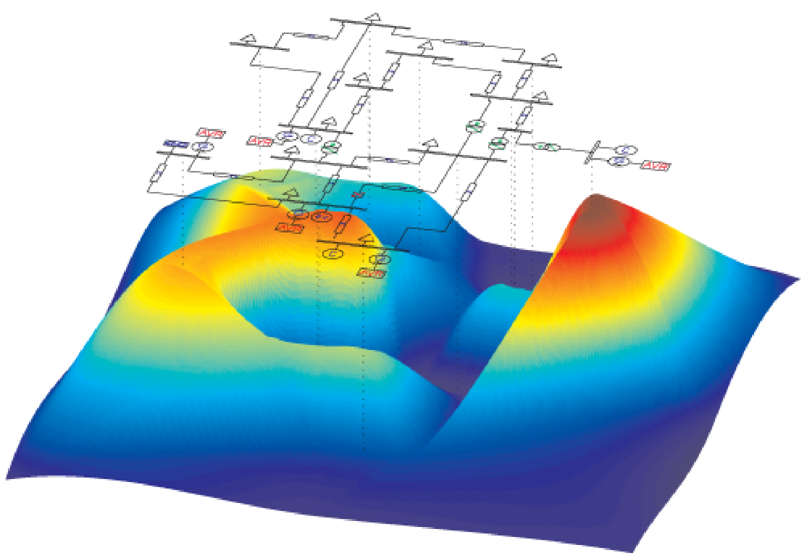
PSAT

Power System Analysis Toolbox

Quick Reference Manual for PSAT version 2.1.2, June 26, 2008



Federico Milano

Note

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

PSAT is a work in progress. Features, structures and data formats can 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`

Important Note. Although the PSAT code and a reduced manual that briefly describes the PSAT format are distributed for free, the full documentation is no longer provided for free. Also technical assistance on the program is no longer provided for free. If you are interested in such service, please contact the author to get an agreement.

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 Deparment, 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

Chapter 1

Introduction

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

1.1 Overview

PSAT is a MATLAB toolbox for electric power system analysis and control. The command line version of PSAT is also 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 [8,11], thermostatically controlled loads [9], Jimma's loads [10], 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.

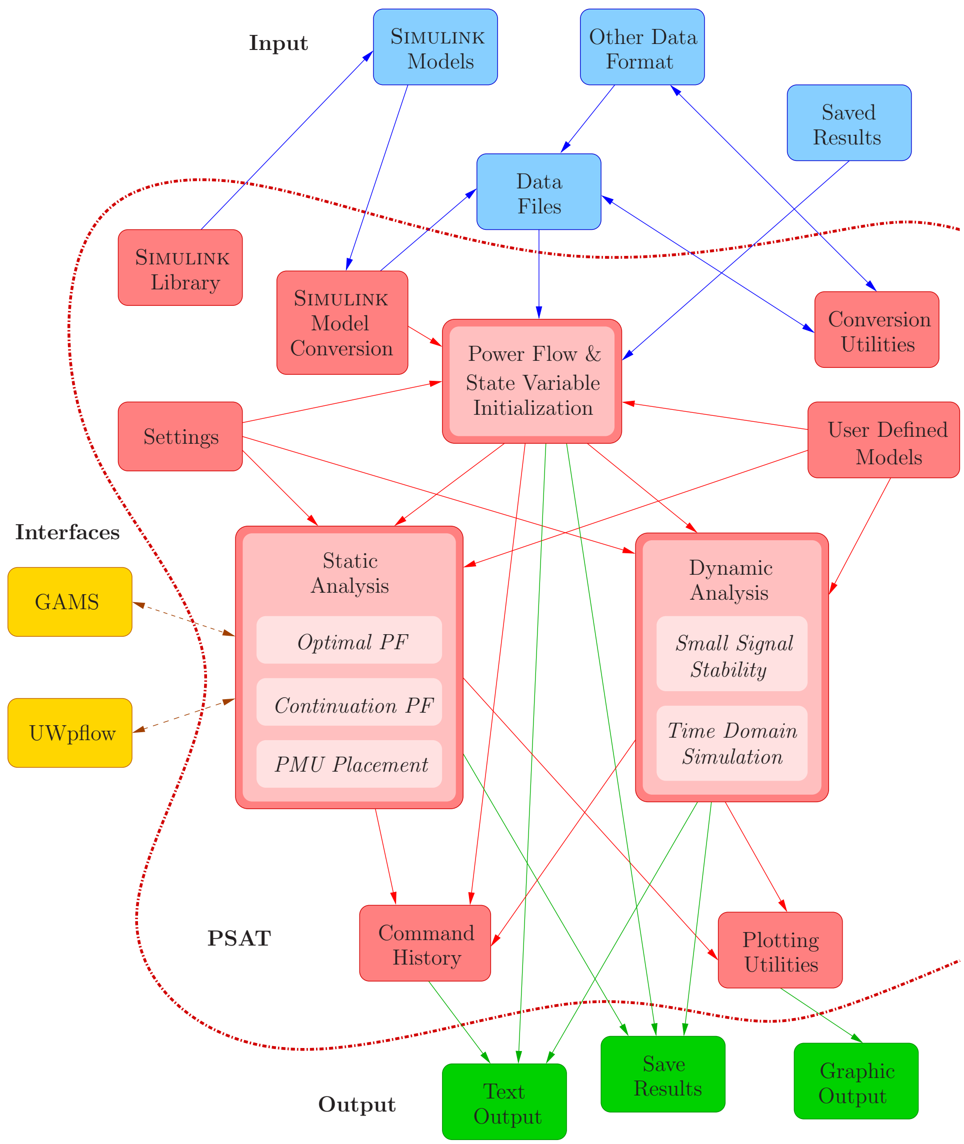


Figure 1.1: PSAT at a glance.

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) [16];
2. MatEMTP [12];
3. MATPOWER [18];
4. Power System Toolbox (PST) [7, 6, 5]
5. Power Analysis Toolbox (PAT) [14];
6. SimPowerSystems (SPS) [15];[1]
7. Voltage Stability Toolbox (VST) [4, 13].

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

1.3 Outlines of the Full PSAT Documentation

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

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

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

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

[1]Since 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 Outlines of the Quick Reference Manual

The quick reference manual describes the installation; the complete PSAT format; the PSAT-Simulink Library; the command line usage on Matlab and GNU Octave; and the complete list of stuctures, classes and functions.

1.5 Users

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

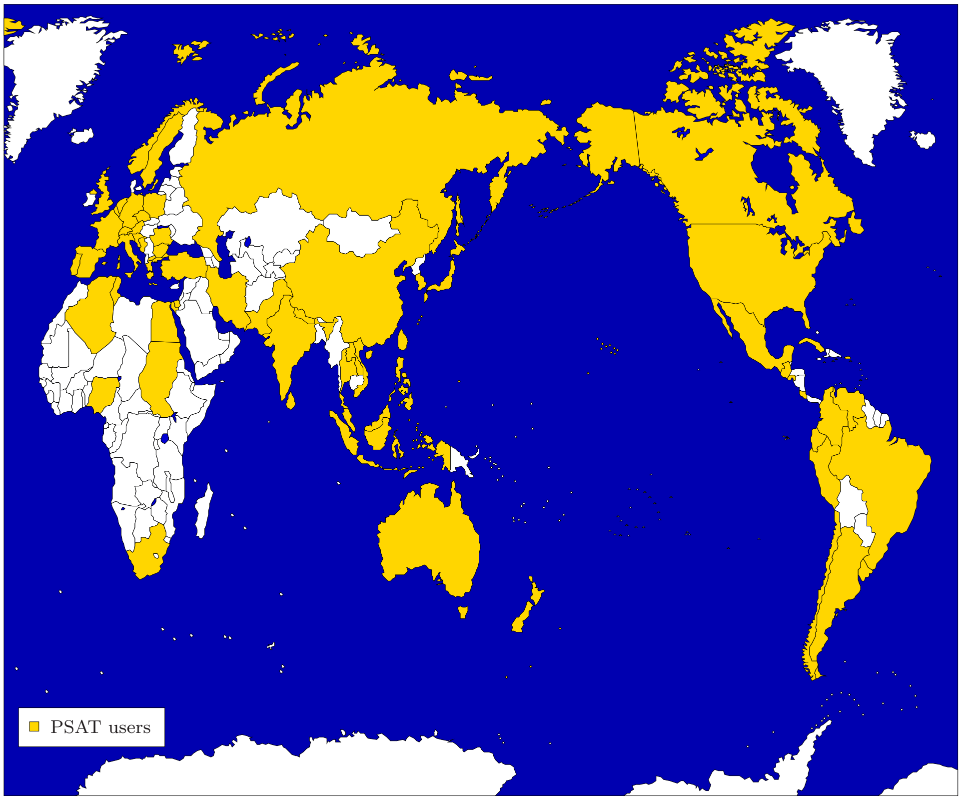


Figure 1.2: PSAT around the world.

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 2.1.2 can run on Linux, Unix, Mac OS X, and Windows operating systems and on Matlab versions from 5.3 to 7.6 (R2008a) and Octave version 3.0.0.[1] The Simulink library and the GUIs can be used on Matlab 7.0 (R14) or higher. On older versions of Matlab and on GNU Octave, only the command line mode of PSAT is available. Chapters 5 and 6 provide further details on the command line usage on Matlab and on GNU Octave.

The requirements of PSAT for running on Matlab are minimal: only the basic Matlab and Simulink packages are needed, except for compiling user defined models, which requires the Symbolic Toolbox. If using Octave 3.0.0, the extra packages Java and JHandles,[2] even though not necessary right now, will likely be required in future releases.

[1] Available at `www.gnu.org/software/octave`

[2] Available at `octave.sourgeforge.net`

2.3 Installation

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

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

or:

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

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

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

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

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

1. Open the GUI available at the menu *File/Set Path* of the main MATLAB window. Then type or browse the PSAT folder and save the session. Note that on some Unix platforms, it is not allowed to overwrite the `pathdef.m` file and you will be requested to write a new `pathdef.m` in a writable location. If this is the case, save it in a convenient folder but remember to start future MATLAB session from that folder in order to make MATLAB to use your custom path list.

2. If you started MATLAB with the `-nojvm` option, you cannot launch the GUI from the main window menu. In this case, use the `addpath` function, which will do the same job as the GUI but at the MATLAB prompt. For example:

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

   or:

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

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

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

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 highly recommended not to change the position and the names of these folders. PSAT can work properly only if the current Matlab folder and the data file folders are writable. Furthermore, if you want to build and install user defined components, the PSAT folder should also be writable.

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

2.4 Launching PSAT

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

```
>> psat
```

This will create all the classes and the structures required by the toolbox, as follows:[3]

```
>> 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
```

[3]By 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`.

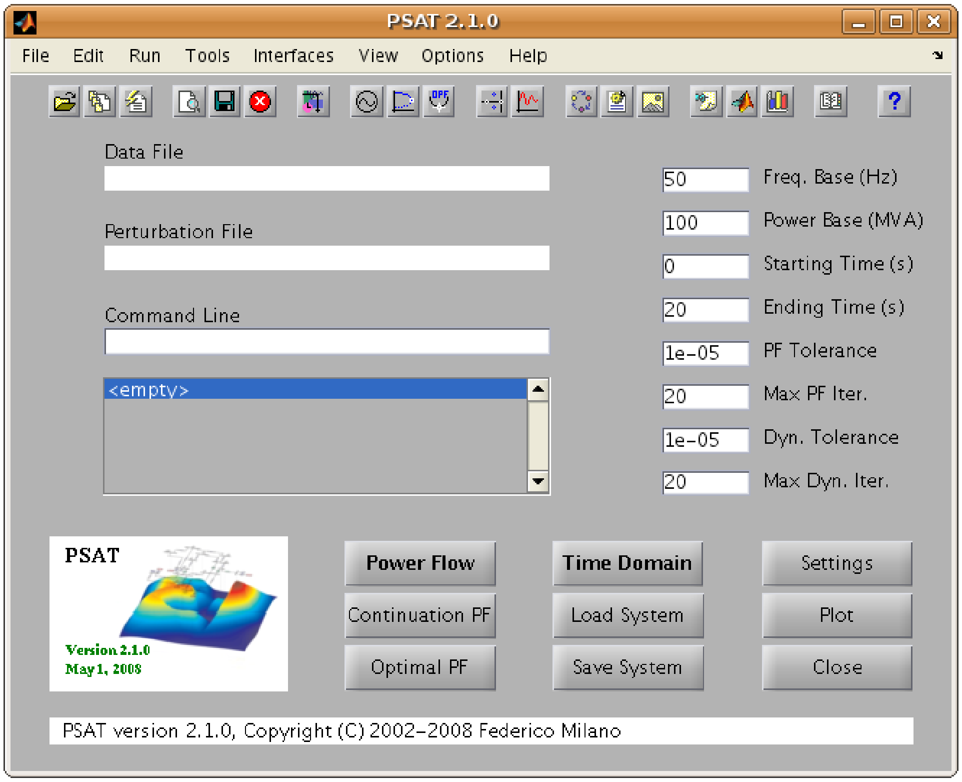


Figure 2.1: Main graphical user interface of PSAT.

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 window[4] 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.

[4]This 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.

2.5 Loading Data

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

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

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

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

2.6 Running the Program

Setting a data file does not actually load or update the component structures. To do this, one has to run the power flow routine, which can be launched in several ways from the main window (e.g. by the shortcut <Ctr-p>). 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, Optimal Power Flow, Small Signal Stability Analysis, Time Domain Simulation, PMU placement, etc. Each of these procedures can be launched from the tool-bar or the menu-bar of the main window.

2.7 Displaying Results

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

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]

2.8 Saving Results

At any time the menu *File/Save/Current System* or the shortcut <Ctr-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 LaTeX.

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 an they can be modified at any time. These settings are the frequency and power bases,

[5] All 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.

[6] Well, after writing the 99^{th} file, the file with the number *01* is actually overwritten without asking for any confirmation.

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 <Ctr-k>). All these settings and data are stored in the `Settings` structure which is fully described in Appendix A. The default values for some fields of the `Settings` structure can be restored by means of the menu *Edit/Set Default*. Customized settings can be saved and used as default values for the next sessions by means of the menu *File/Save/Settings*.

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

A different class of settings is related to the PSAT graphical interface appearance, the preferred text viewer for the text outputs and the settings for the command history interface.

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>.

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 4) 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.

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 [1].

UWPFLOW is an open source program for sophisticated continuation power flow analysis [2]. 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.

Chapter 3

PSAT Data Fomat

This chapter describes the complete data format of all components and devices implementes in PSAT. The mathematical models are not included in the quick reference manual. Refer to the full PSAT documentation for the description of the models.

Table 3.1: 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 *(not used yet...)*	int
† 6	R_i	Region number *(not used yet...)*	int

Table 3.2: 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	-	*not used*	-
8	r	Resistance	p.u. (Ω/km)
9	x	Reactance	p.u. (H/km)
10	b	Susceptance	p.u. (F/km)
† 11	-	*not used*	-
† 12	-	*not used*	-
† 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 3.3: 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	-	*not used*	-
7	k_T	Primary and secondary voltage ratio	kV/kV
8	r	Resistance	p.u.
9	x	Reactance	p.u.
10	-	*not used*	-
† 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	$\{0, 1\}$

Table 3.4: 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 3.5: 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 3.6: 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	$\{0,1\}$

Table 3.7: 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	$\{0,1\}$

Table 3.8: 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	$\{0,1\}$

Table 3.9: 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 3.10: Shunt Admittance Data Format (`Shunt.con`)

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

Table 3.11: 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	$P_{\rm ex}$	Interchange export ($> 0 =$ out)	p.u.
† 5	$P_{\rm tol}$	Interchange tolerance	p.u.
† 6	$\Delta P_{\%}$	Annual growth rate	%
† 7	$P_{\rm net}$	Actual real power net interchange	p.u.
† 8	$Q_{\rm net}$	Actual reactive power net interchange	p.u.

Table 3.12: 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	$P_S^{\rm max}$	Maximum power bid	p.u.
5	$P_S^{\rm min}$	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	$Q_g^{\rm max}$	Maximum reactive power $Q_g^{\rm max}$	p.u.
17	$Q_g^{\rm min}$	Minimum reactive power $Q_g^{\rm min}$	p.u.
18	$C^{{\rm up}_S}$	Congestion up cost	\$/h
19	$C^{{\rm dw}_S}$	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.

Table 3.13: 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	$\{0, 1\}$

Table 3.14: 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	$\{0, 1\}$

Table 3.15: Load Ramping Data Format (`Rmpl.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	R_{up}	Ramp rate up	p.u./min
4	R_{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	$\{0, 1\}$

Table 3.16: 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.

Table 3.17: 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	$\{1, \ldots, 6\}$
205	β	Day of the week	$\{1, \ldots, 7\}$
206	γ	Week of the year	$\{1, \ldots, 52\}$

Table 3.18: 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.

Table 3.19: Breaker Data Format (`Breaker.con`)

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

Table 3.20: 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	$\{0,1\}$

Table 3.21: 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	$\{0,1\}$

Table 3.22: 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	$\{1, 0\}$

Table 3.23: ZIP Load Data Format (`Pl.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	$\{1, 0\}$

Table 3.24: Frequency Dependent Load Data Format (`Fl.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 3.25: 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	$\{1, 0\}$

Table 3.26: 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	$\{0, 1\}$

Table 3.27: 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_{L_Z}	Percentage of active power $\propto V^2$	%
7	P_{L_I}	Percentage of active power $\propto V$	%
8	P_{L_P}	Percentage of constant active power	%
9	Q_{L_Z}	Percentage of reactive power $\propto V^2$	%
10	Q_{L_I}	Percentage of reactive power $\propto V$	%
11	Q_{L_P}	Percentage of constant reactive power	%
12	K_V	Coefficient of the voltage time derivative	1/s
13	u	Connection status	$\{0, 1\}$

Table 3.28: 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	$\{0, 1\}$

Table 3.29: 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

Table 3.30: 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

Table 3.31: Turbine Governor Type I Data Format (`Tg.con`)

Column	Variable	Description	Unit
1	-	Generator number	int
2	1	Turbine governor type	int
3	$\omega_{\mathrm{ref}0}$	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_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	$\{0,1\}$

Table 3.32: Turbine Governor Type II Data Format (`Tg.con`)

Column	Variable	Description	Unit
1	-	Generator number	int
2	2	Turbine governor type	int
3	$\omega_{\mathrm{ref}0}$	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	-	*Not used*	-
10	-	*Not used*	-
11	-	*Not used*	-
12	u	Connection status	$\{0,1\}$

Table 3.33: 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	$\{0, 1\}$

Table 3.34: 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	$\{0, 1\}$

Table 3.35: 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_{f_0}	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	-	*Not used*	-
13	-	*Not used*	-
14	u	Connection status	$\{0, 1\}$

Table 3.36: 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	$I_{f\lim}$	Maximum field current	p.u.
7	$v_{\max}$	Maximum output signal	p.u.
8	u	Connection status	$\{0, 1\}$

Table 3.37: 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 $\Rightarrow$ ω, 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

Table 3.38: 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	$V_{P_{\rm ref}}$	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 3.39: Cluster Controller Data Format (`Cluster.con`)

Column	Variable	Description	Unit
1	-	Central Area Controller number	int
2	-	AVR or SVC number	int
3	-	Control type (1) AVR; (2) SVC	int
4	T_g (T_{svc})	Integral time constant	s
5	x_{t_g}	Generator transformer reactance	p.u.
6	x_{eq_g} ($x_{eq_{svc}}$)	Equivalent reactance	p.u.
7	Q_{g_r} (Q_{svc_r})	Reactive power ratio	p.u.
8	$V_{s_{\max}}$	Maximum output signal	p.u.
9	$V_{s_{\min}}$	Minimum output signal	p.u.
10	u	Connection status	$\{0,1\}$

Table 3.40: Power Oscillation Damper Data Format (`Pod.con`)

Column	Variable	Description	Unit
1	-	Bus or line number	int
2	-	FACTS number	int
3	-	Input signal: 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}	int
4	-	FACTS type: 1 SVC; 2 TCSC; 3 STATCOM; 4 SSSC; 5 UPFC	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	$\{0, 1\}$

Table 3.41: 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	$V_{\rm ref}$ ($Q_{\rm ref}$)	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	-	Control: 1 Secondary voltage V_m; 2 Reactive power Q_m; 3 Remote voltage V_r	int
17	u	Connection status	$\{0, 1\}$

Table 3.42: 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	$\{0,1\}$

Table 3.43: 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	$\alpha_{\max}$	Maximum phase angle	rad
14	$\alpha_{\min}$	Minimum phase angle	rad
15	m	Transformer fixed tap ratio	p.u./p.u.
16	u	Connection status	$\{0,1\}$

Table 3.44: SVC Type 1 Data Format (`Svc.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	S_n	Power rating	MVA
3	V_n	Voltage rating	kV
4	f_n	Frequency rating	Hz
5	1	Model type	int
6	T_r	Regulator time constant	s
7	K_r	Regulator gain	p.u./p.u.
8	V_{ref}	Reference Voltage	p.u.
9	$b_{\max}$	Maximum susceptance	p.u.
10	$b_{\min}$	Minimum susceptance	p.u.
17	u	Connection status	$\{0,1\}$

Table 3.45: 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	$\{0,1\}$

Table 3.46: TCSC Data Format (`Tcsc.con`)

Column	Variable	Description	Unit
1	i	Line number	int
2	-	Model type: 1 Reactance x_{TCSC}; 2 Firing angle α	int
3	-	Operation mode: 1 Constant x_{TCSC}; 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	$x_{\mathrm{TCSC}}^{\max}$ ($\alpha^{\max}$)	Maximum reactance (firing angle)	rad
11	$x_{\mathrm{TCSC}}^{\min}$ ($\alpha^{\min}$)	Minimum reactance (firing angle)	rad
12	K_P	Proportional gain of PI controller	p.u./p.u.
13	K_I	Integral gain of PI controller	p.u./p.u.
14	x_L	Reactance (inductive)	p.u.
15	x_C	Reactance (capacitive)	p.u.
16	K_r	Gain of the stabilizing signal	p.u./p.u.
17	u	Connection status	$\{0, 1\}$

Table 3.47: STATCOM Data Format (`Statcom.con`)

Column	Variable	Description	Unit
1	k	Bus number	int
2	S_n	Power rating	MVA
3	V_n	Voltage rating	kV
4	f_n	Frequency rating	Hz
5	K_r	Regulator gain	p.u./p.u.
6	T_r	Regulator time constant	s
7	$I_{\max}$	Maximum current	p.u.
8	$I_{\min}$	Minimum current	p.u.
9	u	Connection status	$\{0, 1\}$

Table 3.48: 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
8	$v_s^{\max}$	Maximum series voltage v_s	p.u.
9	$v_s^{\min}$	Minimum series voltage v_s	p.u.
10	-	Scheduling type: 1 Constant P_{km}; 2 Constant θ_{km}	int
11	K_P	Proportional gain of PI controller	p.u./p.u.
12	K_I	Integral gain of PI controller	p.u./p.u.
13	u	Connection status	$\{0, 1\}$

Table 3.49: 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	$\{0, 1\}$

Table 3.50: 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_{t_R}	Transformer reactance (rectifier)	p.u.
10	X_{t_I}	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, 3: voltage)	int.
26	I_{ord}	dc current order	p.u.
27	P_{ord}	dc active power order	p.u.
28	V_{ord}	dc voltage order	p.u.
29	u	Connection status	$\{0, 1\}$

Table 3.51: Wind Speed Data Format (`Wind.con`)

Column	Variable	Description	Unit
1	-	Wind model: 1 Measurement data; 2 Weibull distribution; 3 Composite model	int
2	v_{w_N}	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

Table 3.52: Constant Speed Wind Turbine Data Format (`Cswt.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	-	Wind speed number	int
3	S_n	Power rating	MVA
4	V_n	Voltage rating	kV
5	f_n	Frequency rating	Hz
6	r_s	Stator resistance	p.u.
7	x_s	Stator reactance	p.u.
8	r_r	Rotor resistance	p.u.
9	x_r	Rotor reactance	p.u.
10	x_μ	Magnetizing reactance	p.u.
11	H_t	Wind turbine inertia	kWs/kVA
12	H_m	Rotor inertia	kWs/kVA
13	K_s	Shaft stiffness	p.u.
14	R	Rotor radius	m
15	p	Number of poles	int
16	n_b	Number of blades	int
17	η_{GB}	Gear box ratio	-
18	u	Connection status	$\{0, 1\}$

Table 3.53: Doubly Fed Induction Generator Data Format (`Dfig.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	-	Wind speed number	int
3	S_n	Power rating	MVA
4	V_n	Voltage rating	kV
5	f_n	Frequency rating	Hz
6	r_s	Stator resistance	p.u.
7	x_s	Stator reactance	p.u.
8	r_r	Rotor resistance	p.u.
9	x_r	Rotor reactance	p.u.
10	x_μ	Magnetizing reactance	p.u.
11	H_m	Rotor inertia	kWs/kVA
12	K_p	Pitch control gain	-
13	T_p	Pitch control time constant	s
14	K_V	Voltage control gain	-
15	T_ϵ	Power control time constant	s
16	R	Rotor radius	m
17	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	$\{0,1\}$

Table 3.54: Direct Drive Synchronous Generator Data Format (`Ddsg.con`)

Column	Variable	Description	Unit
1	-	Bus number	int
2	-	Wind speed number	int
3	S_n	Power rating	MVA
4	V_n	Voltage rating	kV
5	f_n	Frequency rating	Hz
6	r_s	Stator resistance	p.u.
7	x_d	d-axis reactance	p.u.
8	x_q	q-axis reactance	p.u.
9	ψ_p	Permanent field flux	p.u.
10	H_m	Rotor inertia	kWs/kVA
11	K_p	Pitch control gain	-
12	T_p	Pitch control time constant	s
13	K_V	Voltage control gain	-
14	T_V	Voltage control time constant	s
15	$T_{\epsilon p}$	Active power control time constant	s
16	$T_{\epsilon q}$	Reactive power control time constant	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	$\{0, 1\}$

Table 3.55: 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	$\{0, 1\}$

Table 3.56: 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	$\{0, 1\}$

Table 3.57: 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.

Chapter 4

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] PSAT data can be also converted into the ODM format.

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 4.1 depicts the this GUI.

The following filters have been implemented so far:[3]

`cepel2psat`: conversion from CEPEL data format;

`chapman2psat`: conversion from Chapman's data format [3];

`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;

[1] Most of these filters have been kindly contributed by Juan Carlos Morataya R., Planificación y Control, EEGSA, Iberdrola, Guatemala. E-mail: `JMorataya@eegsa.net`.

[2] Details on the IEEE Common Data Format can be find in [17]. Furthermore, a description of the IEEE CDF and on the EPRI ETMSP formats can be found at `www.power.uwaterloo.ca/`

[3] All filters can be found in the folder `psat/filters`.

`ge2psat`: conversion from General Electric data format;

`ieee2psat`: conversion from IEEE common data format;

`inptc12psat`: conversion from CESI INPTC1 data format;

`ipss2psat`: conversion from InterPSS XML data format;

`ipssdat2psat`: conversion from InterPSS plain data format;

`matpower2psat.m`: conversion from Matpower data format;

`neplan2psat`: conversion from NEPLAN data format;[4]

`odm2psat`: conversion from ODM data format;

`pcflo2psat`: conversion from PCFLO data format;

`psap2psat`: conversion from PSAP data format;[5]

`psat2ieee.m`: conversion to IEEE common data format;

`psat2epri.m`: conversion to EPRI/WSCC data format;

`psat2odm.m`: conversion to ODM format;

`psse2psat`: conversion from PSS/E data format (up to version 29);[6]

`pst2psat.m`: conversion from PST data format;

`pwrworld2psat`: conversion from PowerWorld data format;

`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]
```

[4] This filter supports both comma and tab separated data formats.

[5] A description of the PSAP data format can be found at `www.ee.washington.edu/research/pstca/`

[6] The 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/`

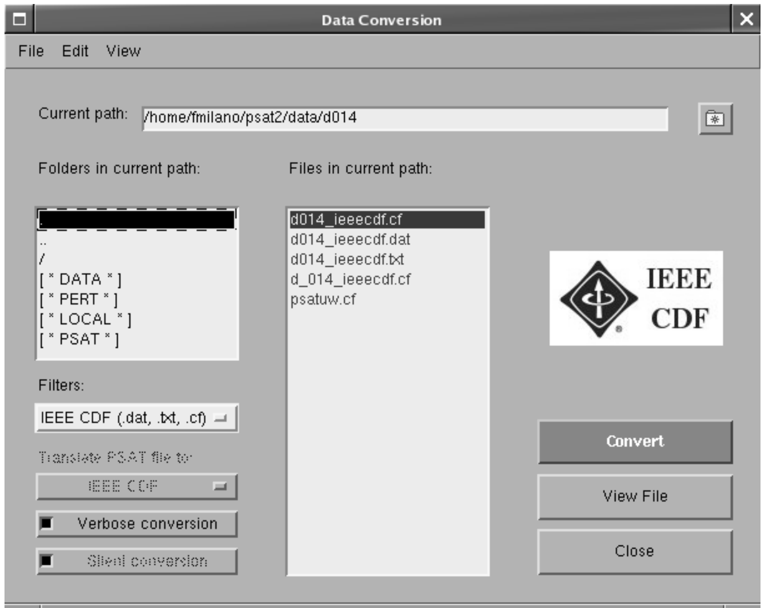


Figure 4.1: GUI for data format conversion.

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.

`-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 5

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.

5.1 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 form the MATLAB command line of 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)
```

[1] The latter should not have been launched from within a function.

Table 5.1: 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 5.1. 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 5.2. 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 5.2: General Options for Command Line Usage.

String	Associated routine
`data`	set data file
`pert`	set perturbation file
`opensys`	open solved case
`savesys`	save current system
`log`	write log file of the current session
`pfrep`	write current power flow solution
`eigrep`	write eigenvalue report file
`pmurep`	write PMU placement report file

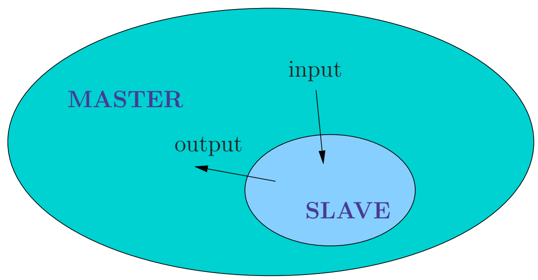


Figure 5.1: Master-slave architecture.

5.2 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 5.1). 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 5.3: 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 5.3 depicts PSAT routines and the associated global structures which define routine options. A full description of these structures is presented in Appendix A.

5.3 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.

[2]In 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.

5.4 *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 6

Running PSAT on GNU Octave

GNU Octave[1] 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 . 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.

Since version 2.1, PSAT can be adapted to run on GNU Octave. PSAT has been tested on Octave version 3.0.0 and 3.0.1 [2] for Linux and Mac OS X. The extra packages provided by the Octave-forge community are not currently needed for running PSAT.[3] The following restrictions apply:

1. Only the command line usage of PSAT is allowed.

2. There is no support for Simulink models.

3. Only a rudimental plotting utility is available.

6.1 Setting up PSAT for Running on GNU Octave

PSAT functions have to be adapted before being able to use PSAT on GNU Octave. The Perl filter `psat2octave` does the job automatically. The steps are as follows:

1. Decompress the PSAT tarball as described in Section 2.3.

[1] Octave is available at `www.octave.org`.
[2] Ealier versions of Octave have not been tested.
[3] Octave-forge is available at `http://octave.sourgeforge.net/`.

2. Be sure that the folder `filters` within the main SPAT folder is in the search path or copy the file `psat2octave` to a folder that is in your search path.

3. Make sure that the file `psat2octave` is executable.

4. Open a terminal and move to the PST folder.

5. Launch `psat2octave`. To launch the file, keep in mind that `psat2octave` is a Perl script. Thus, on Unix-like systems is sufficient to call the script right away, while on Windows, it may be necessary to use the command `perl psat2octave`.

The script `psat2octave` takes a while for completing all necessary changes. It is possible to display some messages during the process by using the *verbose* options:

```
>> psat2octave -v
```

or the *really verbose* option, as follows:

```
>> psat2octave -w
```

If everything goes well and there are no error messages, PSAT is ready to run on GNU Octave. One can revert the conversion and come back to the original PSAT distribution using the command:

```
>> psat2octave -r
```

Other options of the `psat2octave` can be found printing out the help, as follows:

```
>> psat2octave -h
```

6.1.1 How does the conversion works?

PSAT deeply exploits Matlab classes. Unfortunately, Octave does not currently support the definition of custom data types such as classes. The basic idea that is behind the script `psat2octave` is to downgrade all PSAT classes to a structure and a group of functions. The major problem is to solve the issue of function "overloading" that is a basic property of classes but cannot work on GNU Octave. Thus, the script `psat2octave` basically modifies PSAT functions and class methods so that all methods become function with a unique name.

A minor issue is the not full compatibility of Octave function with the correspondent Matlab functions. The small differences between the two environments are taken into account through embedded code in the PSAT functions.

6.2 Basic Commands

All commands provided by the command line usage (see Chapter 5) work well on GNU Octave. However observe that, on GNU Octave, the syntax

```
>> runpsat command
```

is not allowed and one of the following functional forms

```
>> runpsat('command')
>> runpsat("command")
```

must be used. Furthermore, on GNU Octave, both `initpsat` and `psat` launch the command line version of PSAT, which will result in the following message:

```
                           < P S A T >
             Copyright (C) 2002-2008 Federico Milano
                          Version 2.1.0
                           May 1, 2008

   PSAT comes with ABSOLUTELY NO WARRANTY; type 'gnuwarranty'
   for details. This is free software, and you are welcome to
   redistribute it under certain conditions; type 'gnulicense'
   for details.

   Host:          Octave 3.0.0
   Session:       04-May-2008 12:19:59
   Usage:         Command Line
   Path:          /home/fmilano/temp/psat2
```

6.3 Plot Variables

The `runpsat` function admits the additional option `plot` on GNU Octave. The routine will print a menu and wait for the user answer, as follows:

```
octave:100> runpsat('plot')
Plot variables:

  [ 1] States
  [ 2] Voltage Magnitudes
  [ 3] Voltage Angles
  [ 4] Active Powers
  [ 5] Reactive Powers
  [ 6] Generator speeds
  [ 7] Generator angles

pick a number, any number:
```

Figure 6.1 depicts an example of plot obtained using Octave and gplot. The graphs refers to the generator speeds of the WSCC 9-bus example.

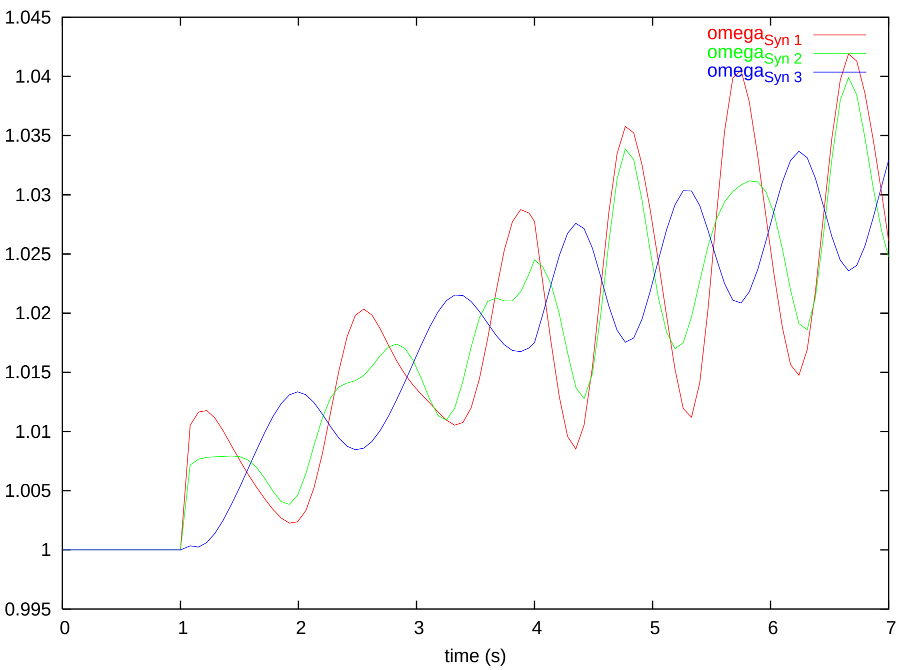


Figure 6.1: Example of graph obtained using OCTAVE and gplot.

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 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

`matlab` true if the current PSAT session is running on MATLAB

0	PSAT is not running on MATLAB
1	PSAT is running on MATLAB

`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 fo lines, load tap changers, phase shifters and HVDC lines.

`octave` treu if the current PSAT session is running on OCTAVE

0	PSAT is not running on OCTAVE
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

`platform` computer architecture or platform: `UNIX`, `MAC`, or `PC`.

`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

`report` style of the power flow report

0 bus report and line report are separated (default)
1 the line flows are embedded in the bus report

`resetangles` after clearing a fault, reset bus angles to the pre-fault values. This can in some cases improve the convergence of the post-fault point during time domain simulations.

0 disabled (default)
1 enabled

`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

`switch2nr` in power flow analysis, switch a robust power flow method to the standard NR method if the tolerance error is smaller than 10^{-2}.

0 disabled (default)
1 enabled

`t0` initial simulation time [s]

`tf` final simulation time [s]

`tstep` fixed time step value [s]

`tviewer` current text viewer

`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:

`about`	PSAT information GUI
`author`	author's pic
`clock`	analogical watch window
`comp`	user defined component browser
`cpf`	continuation power flow GUI
`cset`	mask for user defined component properties
`dir`	file browser and data format conversion GUI
`eigen`	small signal stability analysis GUI
`gams`	GUI for the PSAT-GAMS interface
`hist`	command history GUI
`laprint`	GUI for the LaTeXsettings
`lib`	GUI for limit-induced bifurcations
`license`	GUI that displays the program licence
`line`	GUI for editing the plotted line properties
`main`	PSAT main window
`make`	GUI for building user defined components
`matrx`	GUI for Jacobian matrix visualization
`opf`	optimal power flow GUI
`plot`	GUI for plotting variables
`plotsel`	GUI for selecting output variables for TDs
`pmu`	PMU placement GUI
`pset`	mask for parameter properties
`simset`	GUI for setting SIMULINK model properties
`setting`	general setting GUI
`snap`	GUI for setting snapshots
`snb`	direct method for SNB GUI
`sset`	mask for auxiliary variable properties (*not used*)
`stat`	power flow report GUI
`theme`	theme browser
`threed`	3D system visualization
`tviewer`	GUI for selecting the text viewer
`update`	GUI for installing and uninstalling user defined components
`uwpflow`	GUI for the PSAT-UWPFLOW interface
`xset`	mask for state variable properties
`warranty`	GUI that displays the warranty conditions

`File`: data and disturbance file names, as follows:

`data`	current data file name
`pert`	current disturbance file name

`Path`: path strings of the most commonly used folders, as follows:

`local`	current workspace path
`data`	current data file path
`pert`	current disturbance file path
`psat`	PSAT path
`images`	absolute path of the secondary folder `images`
`build`	absolute path of the secondary folder `build`
`themes`	absolute path of the secondary folder `themes`
`filters`	absolute path of the secondary folder `filters`

`Hdl`: handles of the most used graphic objects.

`hist`	command history listbox in the command history GUI
`text`	message static text in the main window
`status`	axis for convergence status in the main window
`frame`	frame of message text in the main window
`bar`	axis for the progress bar in the main window
`axes`	PSAT logo axis in the main window

`Snapshot`: snapshot data.

`name`	cell array of snapshot names
`time`	array of times associated to the defined snapshots
`y`	vector of algebraic variables
`x`	vector of state variables
`Ybus`	network admittance matrix
`Pg`	vector of generator real powers injected at buses
`Qg`	vector of generator reactive powers injected at buses
`Pl`	vector of load real powers absorbed from buses
`Ql`	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 $n =$ `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 (*not used*)

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 $G_y = \nabla_y g$
`Gl`	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_{LFV_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

`dpdl` 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

`dpdl` 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 λ

`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` equalitiy 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` cureent 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:

`t`	time vector
`vars`	output variables
`idx`	indexes of currently stored output variables
`surf`	handle of the surface plot object
`hdl`	handles of the network scheme in the 3D plots
`zlevel`	high of network scheme in 3D visualization
`movie`	3D movie of the simulation
`alpha`	transparency level for 3D visualization
`caxis`	set voltage limits for 3D visualization
`xb`	x-axis grid data for 3D visualization
`yb`	y-axis grid data for 3D visualization

`Varname`: system variable TEX and plain names. Formatted TEX names are used for creating legends in the plotting variable GUI. Fields are as follows:

`compx`	names of components with state variables
`fnamex`, `unamex`	names of all state variables
`compy`	names of components with algebraic variables
`fnamey`, `unamey`	names of all algebraic variables

`fvars`	formatted names of output variables
`uvars`	unformatted names of output variables
`nvars`	total number of output variables
`idx`	indexes of selected plot variables
`custom`	1 if custom selection of plot variables
`fixed`	1 if fixed selection of plot variables
`x`	1 if selecting all state variables
`y`	1 if selecting all algebraic voltages
`P`	1 if selecting all active power bus injections
`Q`	1 if selecting all reactive power bus injections
`Pij`	1 if selecting all active power flows
`Qij`	1 if selecting all reactive power flows
`Iij`	1 if selecting all current power flows
`Sij`	1 if selecting all apparent power flows
`pos`	vector of positions of plotted variables
`areas`	indexes of the selected areas
`regions`	indexes of the selected regions

A.5 User Defined Models

`Comp`: component general settings

`funct`	cell array of all component functions
`number`	cell array of all component `.n` fields
`prop`	component properties
`n`	total number of installed components
`init`	enable initialization $\{0, 1\}$
`descr`	current component description
`name`	current component name
`shunt`	shunt component $\{0, 1\}$

`Buses` Bus connection variables

`name`	cell array of bus names
`n`	number of buses

`Algeb` Algebraic equations and variables

`name`	cell array of algebraic variables
`n`	number of algebraic variables
`idx`	indexes of algebraic variables
`eq`	cell array of algebraic equations
`eqidx`	indexes of algebraic equations
`neq`	number of algebraic equations

`State` Differential equations and state variables

`name`	cell array of state variables
`n`	number of state variables
`eq`	cell array of differential equations
`eqidx`	indexes of differential equations
`neq`	number of differential equations
`init`	state variable initialization
`limit`	enable anti-windup limiters
`fn`	TeX name of the state variable
`un`	Matlab name of the state variable
`time`	time constant name
`offset`	offset value
`nodyn`	allow time constant being $T = 0$

`Servc` Service equations and variables (*not used...*)

`name`	cell array of service variables
`n`	number of service variables
`eq`	cell array of service equations
`eqidx`	indexes of service equations
`neq`	number of service equations
`init`	service variable initialization
`limit`	enable anti-windup limiters
`fn`	TeX name of the service variable
`un`	Matlab name of the service variable
`type`	service variable type
`offset`	offset value
`oldidx`	cell array of current "external" service variable

`Param` Parameter variables

`name`	cell array of parameter names
`n`	number of parameters
`descr`	parameter description
`type`	parameter type
`unit`	parameter unit

`Initl` 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	Table 3.1
`Line`	Transmission line and transformer	Tables 3.2-3.3
`Lines`	Alternative transmission line	Table 3.4
`Twt`	Three-winding transformer	Table 3.5
`SW`	Slack bus	Table 3.6
`PV`	PV generator	Table 3.7
`PQ`	Constant power load	Table 3.8
`PQgen`	Constant power generator	Table 3.9
`Shunt`	Shunt admittance	Table 3.10
`Areas`	Interchange area	Table 3.11

CPF and OPF Data

`Supply`	Power supply	Table 3.12
`Rsrv`	Generator power reserve	Table 3.13
`Rmpg`	Generator ramping	Table 3.14
`Demand`	Power demand	Table 3.16
`Ypdp`	Demand profile	Table 3.17
`Rmpl`	Power demand ramping	Table 3.15
`Vltn`	Violation parameters	*not used...*

Faults & Breakers

`Fault`	Transmission line fault	Table 3.18
`Breaker`	Transmission line breaker	Table 3.19

Measurements

`Busfreq`	Bus frequency measurement	Table 3.20
`Pmu`	Phasor measurement units	Table 3.21

Loads

`Mn`	Voltage dependent load	Table 3.22
`Fl`	Frequency dependent load	Table 3.24
`Pl`	ZIP (polynomial) load	Table 3.23
`Exload`	Exponential recovery load	Table 3.25
`Thload`	Thermostatically controlled load	Table 3.26
`Jimma`	Jimma's load	Table 3.27
`Mixload`	Mixed load	Table 3.28

Machines

`Syn`	Synchronous machine	Table 3.29
`COI`	Center of inertia	Table 3.29
`Mot`	Induction motor	Table 3.30

Controls

`Tg`	Turbine Governor	Table 3.31-3.32
`Exc`	Automatic Voltage Regulator	Tables 3.33-3.35
`Pss`	Power System Stabilizer	Table 3.37
`Oxl`	Overexcitation Limiter	Table 3.36
`CAC`	Central Area Controller	Table 3.38
`Cluster`	Cluster Controller	Table 3.39
`Pod`	Power Oscillation Damper	Table 3.40

Regulating Transformers

`Ltc`	Load tap changer	Table 3.41
`Tap`	Tap changer with embedded load	Table 3.42
`Phs`	Phase shifting transformer	Table 3.43

FACTS

`Svc`	Static Var Compensator	Tables 3.44-3.45
`Tcsc`	Thyristor Controlled Series Capacitor	Table 3.46
`Statcom`	Static Var Compensator	Table 3.47
`Sssc`	Static Synchronous Source Series Compensator	Table 3.48
`Upfc`	Unified Power Flow Controller	Table 3.49
`Hvdc`	High Voltage DC transmission system	Table 3.50

Wind Turbines

`Wind`	Wind models	Table 3.51
`Cswt`	Constant speed wind turbine	Table 3.52
`Dfig`	Doubly fed induction generator	Table 3.53
`Ddsg`	Direct drive synchronous generator	Table 3.54

Other Models

`Mass`	Synchronous machine dynamic shaft	Table 3.55
`SSR`	Subsynchronous resonance model	Table 3.56
`Sofc`	Solid Oxyde Fuel Cell	Table 3.57

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 λ (float)

`lmin_s` minimum value of λ (string)

`omega` weighting factor ω values (float)

`omega_s` weighting factor ω values (string)

`lmax` maximum value of λ (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 [2].

`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 `Fl` 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 Vltn 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

`psat`	start the program
`fm_set`	general settings and utilities
`fm_var`	definition of global variables
`fm_main`	main GUI

Power Flow

`fm_spf`	standard power flow routine
`fm_nrlf`	power flow with fixed state variables
`fm_flows`	network current/power flows and connectivity
`fm_dynlf`	indexes of state variables (before power flow)
`fm_dynidx`	indexes of state variables (after power flow)
`fm_xfirst`	initial guess of state variables
`fm_ncomp`	indexes of components
`fm_inilf`	reset variables for power flow computations
`fm_stat`	GUI for displaying power flow results
`fm_base`	report of component quantities on system bases
`fm_report`	writes power flow report files
`fm_restore`	reset all components data using the `store` matrices

Direct Methods

`fm_snb`	Saddle-node bifurcation routine
`fm_snbfig`	GUI for saddle-node bifurcations
`fm_limit`	Limit-induced bifurcation routine
`fm_snbfig`	GUI for limit-indeuced 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

autorun	secure routine launch
fm_idx	definition of variable names
fm_iidx	find bus interconnetcions
fm_errv	check component voltage rating
fm_filenum	enumeration of output files
fm_getxy	get x and y indexes within a network zone
fm_laprint	export graphics to *eps* and LaTeX files
fm_qlim	get static generator reactive power limits
fm_rmgen	find and remove static generators connected to a bus
fm_setgy	delete row and columns in power flow Jacobian G_y
fm_status	display convergence error status on main GUI
fm_strjoin	platform independent clone of the `strcat` function
fm_vlim	get bus voltage limits
fm_windup	set windup hard limits
fvar	convert variables in strings
pgrep	search *.m* files for string
psatdomain	dummy function for the PMC Simulink library
psed	substitute string in *.m* files
settings	define customized settings (optional)
sizefig	determine figure size

Simulink Library and Functions

fm_lib	Simulink library
fm_simrep	power flow report for Simulink models
fm_simset	GUI for Simulink model settings
fm_simsave	save a Simulink 5, 4.1 or 4 model as a Simulink 3 model
fm_block	set Simulink block parameters
fm_inout	create and delete Simulink block input/output ports
fm_draw	draw Simulink block icons

Data File Conversion

fm_dir	browser for data conversion
fm_dirset	utilities for data conversion
filters/cepel2psat	CEPEL to PSAT filter (perl file)
filters/chapman2psat	Chapman to PSAT filter (perl file)
filters/cloneblock	Update obsolete PSAT-Simulink blocks
filters/cyme2psat	CYMFLOW to PSAT filter (perl file)
filters/digsilent2psat	DIgSILENT to PSAT filter (perl file)
filters/epri2psat	EPRI to PSAT filter (perl file)
filters/eurostag2psat	Eurostag to PSAT filter (perl file)
filters/flowdemo2psat	FlowDemo.net to PSAT filter (perl file)
filters/ge2psat	GE to PSAT filter (perl file)
filters/ieee2psat	IEEE CDF to PSAT filter (perl file)

`filters/inptc12psat`	CESI INPTC1 to PSAT filter (perl file)
`filters/ipss2psat`	InterPSS to PSAT filter (perl file)
`filters/ipssdat2psat`	InterPSS plain text to PSAT filter (perl file)
`filters/matpower2psat`	Matpower to PSAT filter (m-file)
`filters/neplan2psat`	NEPLAN to PSAT filter (perl file)
`filters/odm2psat`	ODM to PSAT filter (perl file)
`filters/pcflo2psat`	PCFLO to PSAT filter (perl file)
`filters/psap2psat`	PSAP to PSAT filter (perl file)
`filters/psat2epri`	PSAT to EPRI filter (m-file)
`filters/psat2ieee`	PSAT to IEEE filter (m-file)
`filters/psat2octave`	adapt PSAT for Octave (perl file)
`filters/psat2odm`	PSAT to ODM filter (m-file)
`filters/psse2psat`	PSS/E to PSAT filter (perl file)
`filters/pst2psat`	PST to PSAT filter (m-file)
`filters/pwrworld2psat`	PowerWorld to PSAT filter (perl file)
`filters/sim2psat`	Simulink to PSAT filter (m-file)
`filters/simpow2psat`	SIMPOW to PSAT filter (perl file)
`filters/th2psat`	Tsinghua Univ. to PSAT filter (perl file)
`filters/ucte2psat`	UCTE to PSAT filter (perl file)
`filters/vst2psat`	VST to PSAT filter (perl file)
`filters/webflow2psat`	WebFlow to PSAT filter (perl file)

Plotting Utilities

`fm_plot`	general function for plotting results
`fm_plotfig`	GUI for plotting results
`fm_axesdlg`	GUI for axes properties settings
`fm_linedlg`	GUI for line properties settings
`fm_linelist`	GUI for line list browser
`fm_view`	general function for sparse matrix visualization
`fm_matrx`	GUI for sparse matrix visualization
`fm_bar`	plots status bar on main window

Command History

`fm_text`	command history general functions and utilities
`fm_hist`	GUI for command history visualization
`fm_disp`	command, message and error display
`fval`	message line for variable manipulation

Output

`fm_write`	call function for writing output results
`fm_writexls`	write output results in HTML format
`fm_writetex`	write output results in LaTeX format
`fm_writetxt`	write output results in plain text
`fm_writexls`	write output results in Excel format

Themes

fm_theme	theme manager
fm_themefig	GUI of theme manager
fm_mat	background for GUI images

Other GUI Utilities

fm_setting	GUI for general settings
fm_enter	welcome GUI
fm_tviewer	GUI for text viewer selection
fm_about	about PSAT
fm_iview	image viewer
fm_author	author's pic
fm_clock	analogic watch
fm_choice	dialog box

GNU License Functions

gnulicense	type the GNU-GPL
fm_license	GUI for the GNU-GPL
gnuwarranty	type the "no warranty" conditions
fm_warranty	GUI for the "no warranty" conditions

PMU Placement Functions

fm_pmuloc	PMU placement manager
fm_pmun1	PMU placement for device outages
fm_pmurec	recursive method for PMU placement
fm_pmurep	write PMU placement report
fm_pmutry	filter for zero-injection buses
fm_lssest	linear static state estimation
fm_spantree	spanning tree of existing PMUs
fm_mintree	minimum tree search
fm_annealing	annealing method for PMU placement
fm_pmufig	GUI for PMU placement

Command Line Usage

initpsat	initialize PSAT global variables
closepsat	clear all PSAT global variables from workspace
runpsat	launch PSAT routine

Interface Functions

fm_gams	GAMS interface for single-period OPF
fm_gamsfig	GUI of the GAMS interface
fm_uwpflow	UWPFLOW interface

`fm_uwfig`	GUI of the UWPFLOW interface

Linear Analysis Functions

`fm_abcd`	compute input/output matrices A, B, C, D

Numerical Differentiation Functions

`checkjac`	compare numeric and analytic Jacobian matrices
`numjacs`	evaluate numeric Jacobian matrices F_x, F_y, G_x and G_x

Network Equivalents

`fm_busfig`	GUI for selecting bus zones
`fm_equiv`	compute simple static and dynamic network equivalents
`fm_equivfig`	GUI of the network equivalent procedure

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 synthax 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/odm2psat` filter for the ODM data format.

`filters/pcflo2psat` filter for the PCFLOH 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

PSAT Forum

A PSAT Forum (see Fig. D.1) is currently available at:

`tech.groups.yahoo.com/groups/psatforum`

Main functions are as follows:

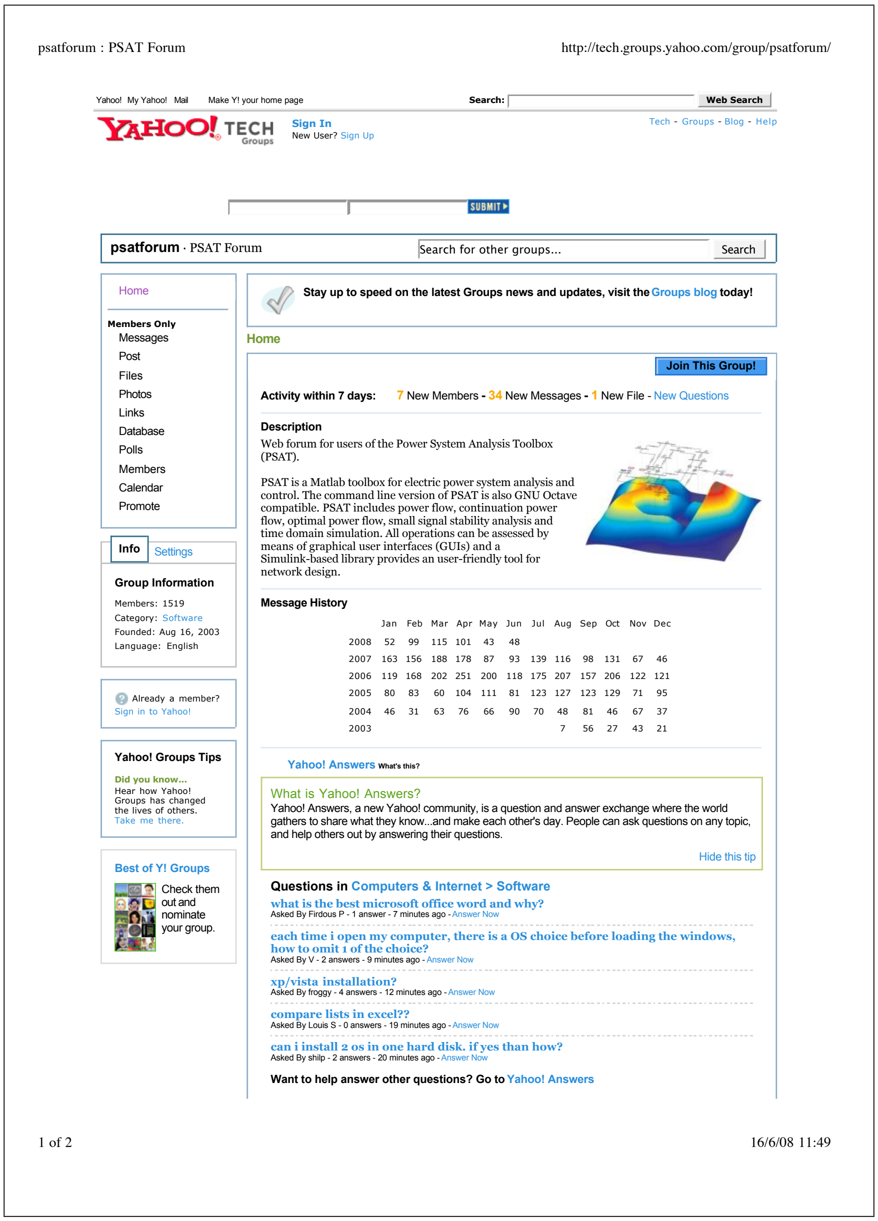
Function	e-mail
Subscribe	psatforum-subscribe@yahoogroups.com
Post message	psatforum@yahoogroups.com
Unsubscribe	psatforum-unsubscribe@yahoogroups.com
List owner	psatforum-owner@yahoogroups.com

To post a message directly to me, 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. D.2.



psatforum : PSAT Forum http://tech.groups.yahoo.com/group/psatforum/

Yahoo! My Yahoo! Mail Make Y! your home page

Search: Web Search

Sign In
New User? Sign Up

Tech - Groups - Blog - Help

SUBMIT

psatforum · PSAT Forum

Search for other groups... Search

Home

Members Only
- Messages
- Post
- Files
- Photos
- Links
- Database
- Polls
- Members
- Calendar
- Promote

Info Settings

Group Information

Members: 1519
Category: Software
Founded: Aug 16, 2003
Language: English

Already a member?
Sign in to Yahoo!

Yahoo! Groups Tips

Did you know...
Hear how Yahoo! Groups has changed the lives of others.
Take me there.

Best of Y! Groups

Check them out and nominate your group.

Stay up to speed on the latest Groups news and updates, visit the Groups blog today!

Home

Join This Group!

Activity within 7 days: 7 New Members - 34 New Messages - 1 New File - New Questions

Description

Web forum for users of the Power System Analysis Toolbox (PSAT).

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

Message History

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2008	52	99	115	101	43	48						
2007	163	156	188	178	87	93	139	116	98	131	67	46
2006	119	168	202	251	200	118	175	207	157	206	122	121
2005	80	83	60	104	111	81	123	127	123	129	71	95
2004	46	31	63	76	66	90	70	48	81	46	67	37
2003								7	56	27	43	21

Yahoo! Answers What's this?

What is Yahoo! Answers?
Yahoo! Answers, a new Yahoo! community, is a question and answer exchange where the world gathers to share what they know...and make each other's day. People can ask questions on any topic, and help others out by answering their questions.

Hide this tip

Questions in Computers & Internet > Software

what is the best microsoft office word and why?
Asked By Firdous P - 1 answer - 7 minutes ago - Answer Now

each time i open my computer, there is a OS choice before loading the windows, how to omit 1 of the choice?
Asked By V - 2 answers - 9 minutes ago - Answer Now

xp/vista installation?
Asked By froggy - 4 answers - 12 minutes ago - Answer Now

compare lists in excel??
Asked By Louis S - 0 answers - 19 minutes ago - Answer Now

can i install 2 os in one hard disk. if yes than how?
Asked By ship - 2 answers - 20 minutes ago - Answer Now

Want to help answer other questions? Go to Yahoo! Answers

1 of 2 16/6/08 11:49

Figure D.1: PSAT Forum main page. Data refer to June 16, 2008.

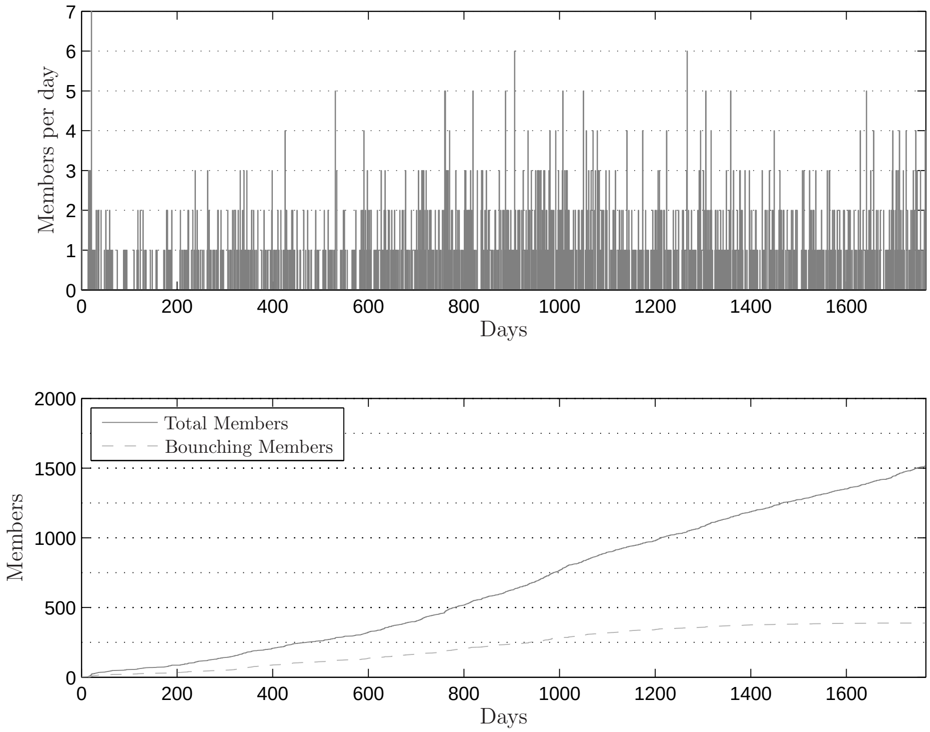


Figure D.2: PSAT Forum statistics. Data refer to June 16, 2008.

Bibliography

[1] A. Brooke, D. Kendrick, A. Meeraus, R. Raman, and R. E. Rosenthal, *GAMS, a User's Guide*, GAMS Development Corporation, 1217 Potomac Street, NW, Washington, DC 20007, USA, Dec. 1998, available at www.gams.com.

[2] C. A. Cañizares and F. L. Alvarado, "UWPFLOW Program," 2000, university of Waterloo, available at http://www.power.uwaterloo.ca.

[3] S. J. Chapman, *Electric Machinery and Power System Fundamentals.* New York: McGraw Hill, 2002.

[4] 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.

[5] J. Chow, *Power System Toolbox Version 2.0: Dynamic Tutorial and Functions*, Cherry Tree Scientific Software, RR-5 Colborne, Ontario K0K 1S0, 1991-1997.

[6] ——, *Power System Toolbox Version 2.0: Load Flow Tutorial and Functions*, Cherry Tree Scientific Software, RR-5 Colborne, Ontario K0K 1S0, 1991-1999.

[7] 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.

[8] 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.

[9] P. Hirsch, *Extended Transient-Midterm Stability Program (ETMSP) Ver. 3.1: User's Manual*, EPRI, TR-102004-V2R1, May 1994.

[10] 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.

[11] 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.

[12] 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.

[13] C. Nwankpa, *Voltage Stability Toolbox, version 2*, Center for Electric Power Engineering, Drexel University, 2002, available at power.ece.drexel.edu/-research/VST/vst.htm.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.