An Open Source Power System Analysis Toolbox

Federico Milano, *Member, IEEE*

***Abstract*—This paper describes the Power System Analysis Toolbox (PSAT), an open source Matlab and GNU/Octave-based software package for analysis and design of small to medium size electric power systems. PSAT includes power flow, continuation power flow, optimal power flow, small-signal stability analysis, and time-domain simulation, as well as several static and dynamic models, including nonconventional loads, synchronous and asynchronous machines, regulators, and FACTS. PSAT is also provided with a complete set of user-friendly graphical interfaces and a Simulink-based editor of one-line network diagrams. Basic features, algorithms, and a variety of case studies are presented in this paper to illustrate the capabilities of the presented tool and its suitability for educational and research purposes.**

***Index Terms*—Continuation power flow, GNU/Octave, Matlab, optimal power flow, power flow, smaill-signal stability analysis, time-domain simulation.**

I. Introduction

SOFTWARE packages for power system analysis can be basically divided into two classes of tools: commercial softwares and educational/research-aimed softwares. Commercial software packages available on the market (e.g. PSS/E, EuroStag, Simpow, and CYME) follows an "all-in-one" philosophy and are typically well-tested and computationally efficient. Despite their completeness, these softwares can result cumbersome for educational and research purposes. Even more important, commercial softwares are "closed", i.e., do not allow changing the source code or adding new algorithms. For research purposes, the flexibility and the ability of easy prototyping are often more crucial aspects than computational efficiency. On the other hand, there is a variety of open source research tools, which are typically aimed to a specific aspect of power system analysis. An example is UWPFLOW [1], which provides an extremely robust algorithm for continuation power flow analysis. However, extending and/or modifying this kind of scientific tools also requires keen programming skills, in addition to a good knowledge of a low level language (C in the case of UWPFLOW) and of the structure of the program.

In the last decade, several high-level scientific languages, such as Matlab, Mathematica, and Modelica, have become more and more popular for both research and educational purposes. Any of these languages can lead to good results in the field of power system analysis (see, for example, [2]); however, Matlab proved to be the best user choice. Key features of Matlab are the matrix-oriented programming, excellent plotting capabilities and a graphical environment (Simulink) which highly simplifies control scheme design. For these reasons, several Matlab-based commercial, research and educational power system tools have been proposed, such as Power System Toolbox (PST) [3], MatPower [4], Toolbox (VST) [5], MatEMTP [6], SimPowerSystems (SPS) [7], Power Analysis Toolbox (PAT) [8], and the Educational Simulation Tool (EST) [9]. Among these, only MatPower and VST are open source and freely downloadable.

TABLE I
MATLAB-BASED PACKAGES FOR POWER SYSTEM ANALYSIS

Package	PF	CPF	OPF	SSA	TD	EMT	GUI	GNE
EST	✓			✓	✓			✓
MatEMTP					✓	✓	✓	✓
MatPower	✓		✓					
PAT	✓			✓	✓			✓
PSAT	✓	✓	✓	✓	✓		✓	✓
PST	✓	✓		✓	✓			
SPS	✓			✓	✓	✓	✓	✓
VST	✓	✓		✓	✓		✓	

This paper describes a new Matlab-based power system analysis tool (PSAT) which is freely distributed on line [10]. PSAT includes power flow, continuation power flow, optimal power flow, small-signal stability analysis, and time-domain simulation. The toolbox is also provided with a complete graphical interface and a Simulink-based one-line network editor. Table I depicts a rough comparison of the currently available Matlab-based tools for power system analysis and PSAT. The features illustrated in the table are the power flow (PF), the continuation power flow and/or voltage stability analysis (CPF-VS), the optimal power flow (OPF), the small-signal stability analysis (SSA), and the time-domain simulation (TD), along with "aesthetic" features such as the graphical user interface (GUI) and the graphical network editor (GNE).

An important but often missed issue is that the Matlab environment is a commercial and "closed" product, thus Matlab kernel and libraries cannot be modified nor freely distributed. To allow exchanging ideas and effectively improving scientific research, both the toolbox and the platform on which the toolbox runs should be free [11]. At this aim, PSAT can run on GNU/Octave [12], which is a free Matlab clone.

The paper is organized as follows. Section II illustrates the main PSAT features while Section III describes the models and the algorithms for power system analysis implemented in PSAT. Section IV presents a variety of case studies based on the IEEE 14-bus test system. Finally, Section V presents conclusions and future work directions.

II. PSAT Features

A. Outlines

PSAT has been thought to be portable and open source. At this aim, PSAT has been developed using Matlab, which runs on the commonest operating systems, such as Unix, Linux, Windows,

Manuscript received November 10, 2004; revised March 2, 2005. Paper no. TPWRS-00596-2004.

The author is with the Department of Electrical Engineering, University of Castilla-La Mancha, Ciudad Real 13071, Spain (e-mail: fmilano@ind-cr.uclm.es).

Digital Object Identifier 10.1109/TPWRS.2005.851911

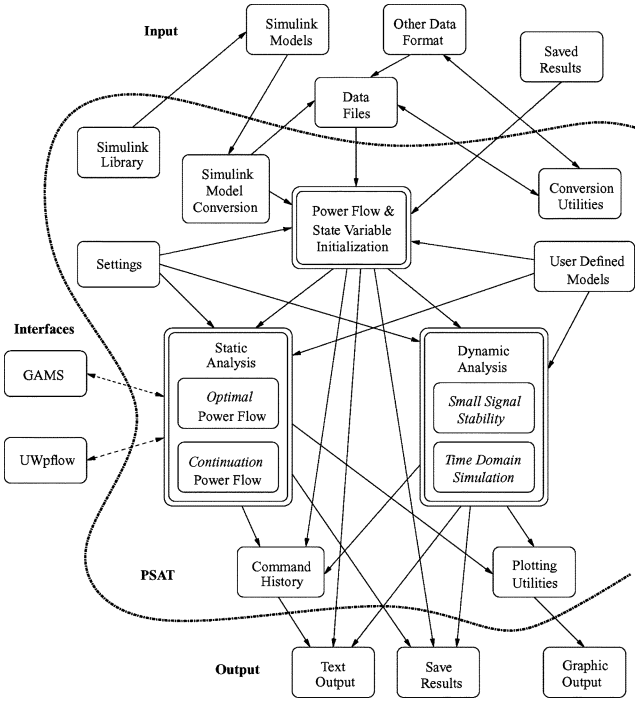

Fig. 1. Synoptic scheme of PSAT.

and Mac OS X. Nevertheless, PSAT would not be completely open source if it run only on Matlab, which is a proprietary software. At this aim PSAT can run also on the latest GNU/Octave releases [12], which is basically a free Matlab clone. In the knowledge of the author, PSAT is actually the first *free software* project in the field of power system analysis. PSAT is also the first power system software which runs on GNU/Octave platforms.

The synoptic scheme of PSAT is depicted in Fig. 1. Observe that PSAT kernel is the power flow algorithm, which also takes care of the state variable initialization. Once the power flow has been solved, the user can perform further static and/or dynamic analyses. These are as follows.

1) Continuation Power Flow (CPF).
2) Optimal Power Flow (OPF).
3) Small-signal stability analysis.
4) Time-domain simulations.

PSAT deeply exploits Matlab vectorized computations and sparse matrix functions in order to optimize performances. Furthermore, PSAT is provided with the most complete set of algorithms for static and dynamic analyses among currently available Matlab-based power system softwares (see Table I). PSAT also contains interfaces to UWPFLOW [1] and GAMS [13], which highly extend PSAT ability to solve CPF and OPF problems, respectively. These interfaces are not discussed here, as they are beyond the main purpose of this paper.

In order to perform accurate and complete power system analyses, PSAT supports a variety of static and dynamic models, as follows.

— *Power Flow Data*: Bus bars, transmission lines and transformers, slack buses, PV generators, constant power loads, and shunt admittances.

TABLE II
FUNCTIONS AVAILABLE ON MATLAB AND GNU/OCTAVE PLATFORMS

Function	Matlab	GNU/Octave
Continuation power flow	yes	yes
Optimal power flow	yes	yes
Small signal stability analysis	yes	yes
Time domain simulation	yes	yes
GUIs and Simulink library	yes	no
Data format conversion	yes	yes
User defined models	yes	no
Command line usage	yes	yes

— *Market Data*: Power supply bids and limits, generator power reserves, and power demand bids and limits.
— *Switches*: Transmission line faults and breakers.
— *Measurements*: Bus frequency measurements.
— *Loads*: Voltage dependent loads, frequency dependent loads, ZIP (polynomial) loads, thermostatically controlled loads, and exponential recovery loads [14].
— *Machines*: Synchronous machines (dynamic order from 2 to 8) and induction motors (dynamic order from 1 to 5).
— *Controls*: Turbine Governors, AVRs, PSSs, Over-excitation limiters, and secondary voltage regulation.
— *Regulating Transformers*: Under load tap changers and phase shifting transformers.
— *FACTS*: SVCs, TCSCs, SSSCs, UPFCs.
— *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, subsynchronous resonance model, solid oxide fuel cell, and subtransmission area equivalents.

Besides mathematical algorithms and models, PSAT includes a variety of additional tools, as follows.

1) User-friendly graphical user interfaces.
2) Simulink library for one-line network diagrams.
3) Data file conversion to and from other formats.
4) User defined model editor and installer.
5) Command line usage.

The following subsections will briefly describe these tools. Observe that, due to GNU/Octave limitations, not all algorithms/tools are available on this platform (see Table II).

B. Getting Started and Main Graphical User Interface

PSAT is launched by typing at the Matlab prompt

```
>> psat
```

which will create all structures required by the toolbox and open the main GUI (see Fig. 2). All procedures implemented in PSAT can be launched from this window by means of menus, buttons, and/or short cuts.

The main settings, such as the system base or the maximum number of iteration of Newton-Raphson (NR) methods, are shown in the main window. Other system parameters and specific algorithm settings have dedicated GUIs (see Figs. 8 and 11). Observe that PSAT does not rely on GUIs and makes use of global variables to store both setting parameters and

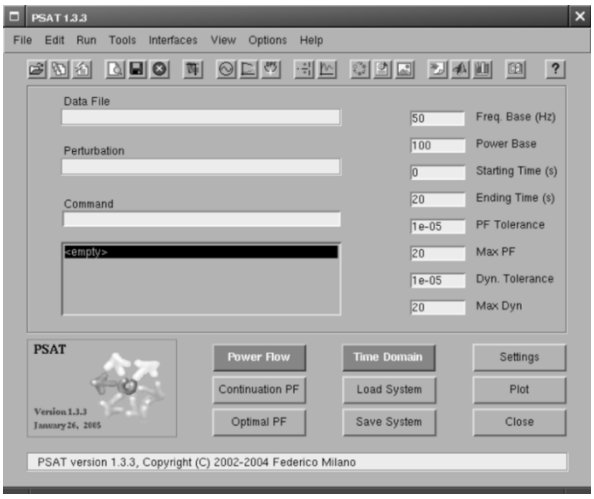

Fig. 2. Main graphical user interface of PSAT.

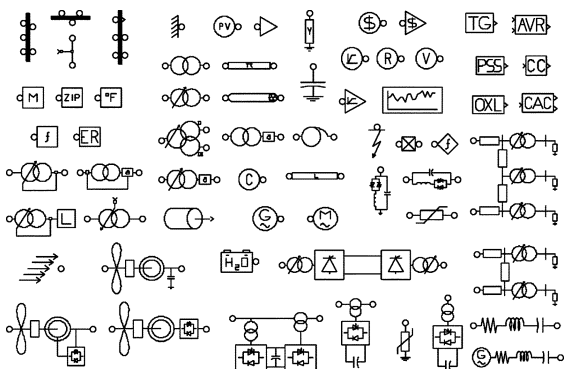

Fig. 3. PSAT simulink library.

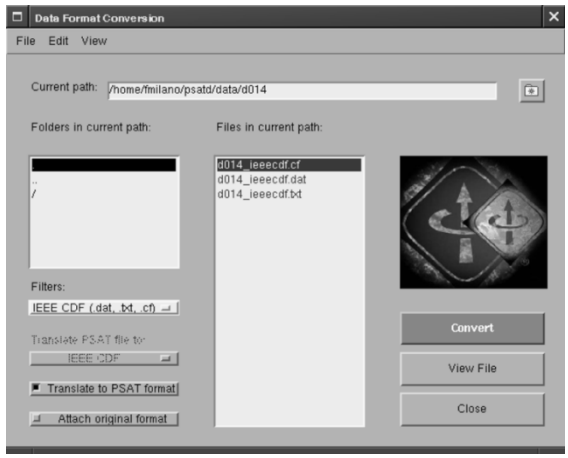

Fig. 4. GUI for data format conversion.

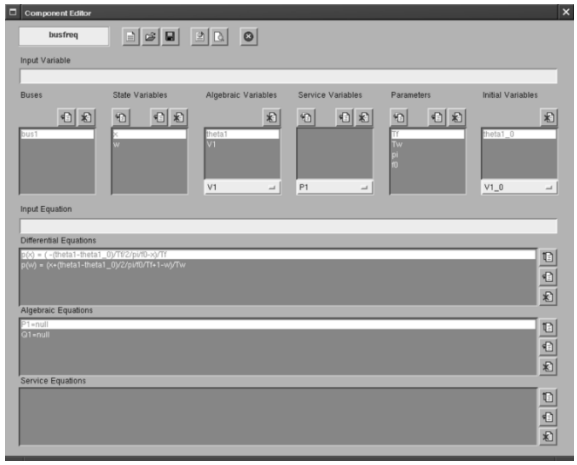

Fig. 5. GUI for user defined models.

data. This approach allows using PSAT from the command line as needed in many applications (see Section II-E).

C. Simulink Library

PSAT allows drawing electrical schemes by means of pictorial blocks. Fig. 3 depicts the complete PSAT-Simulink library (see also Fig. 7, which illustrates the IEEE 14-bus test system).

The PSAT computational engine is purely Matlab-based and the Simulink environment is used only as graphical tool. As a matter of fact, Simulink models are read by PSAT to exploit network topology and extract component data. A byproduct of this approach is that PSAT can run on GNU/Octave, which is currently not providing a Simulink clone.

Observe that some Simulink-based tools, such as PAT [8] and EST [9], use Simulink to simplify the design of new control schemes. This is not possible in PSAT. However, PAT and EST do not allow representing the network topology, thus resulting in a lower readability of the whole system.

D. Data Conversion and User Defined Models

To ensure portability and promote contributions, PSAT is provided with a variety of tools, such as a set of Data Format Conversion (DFC) functions and the capability of defining User Defined Models (UDMs).

The set of DFC functions allows converting data files to and from formats commonly in use in power system analysis. These include: IEEE, EPRI, PTI, PSAP, PSS/E, CYME, MatPower and PST formats. On Matlab platforms, an easy-to-use GUI (see in Fig. 4) handles the DFC.

The UDM tools allow extending the capabilities of PSAT and help end-users to quickly set up their own models. UDMs can be created by means of the GUI depicted in Fig. 5. Once the user has introduced the variables and defined the DAE of the new model in the UDM GUI, PSAT automatically compiles equations, computes symbolic expression of Jacobians matrices (by means of the Symbolic Toolbox) and writes a Matlab function of the new component. Then the user can save the model definition and/or install the model in PSAT. If the component is not needed any longer it can be uninstalled using the UDM installer as well.

E. Command Line Usage

GUIs are useful for education purposes but can in some cases limit the development or the usage of a software. For this reason PSAT is provided with a command line version. This feature allows using PSAT in the following conditions.

1) If it is not possible or very slow to visualize the graphical environment (e.g., Matlab is running on a remote server).
2) If one wants to write scripting of computations or include calls to PSAT functions within user-defined programs.

3) If PSAT runs on the GNU/Octave platform, which currently neither provides GUI tools nor a Simulink-like environment.

III. Models and Algorithms

A. Power System Model

The standard power system model is basically a set of nonlinear differential algebraic equations, as follows:

$$\begin{aligned}\dot{x} &= f(x,y,p)\\ 0 &= g(x,y,p)\end{aligned} \tag{1}$$

where x are the state variables $x \in \mathbb{R}^n$; y are the algebraic variables $y \in \mathbb{R}^m$; p are the independent variables $p \in \mathbb{R}^\ell$; f are the differential equations $f : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^\ell \mapsto \mathbb{R}^n$; and g are the algebraic equations $g : \mathbb{R}^m \times \mathbb{R}^m \times \mathbb{R}^\ell \mapsto \mathbb{R}^m$.

PSAT uses (1) in all algorithms, namely power flow, CPF, OPF, smaill-signal stability analysis, and time-domain simulation, as discussed in the following Sections III-B to III-F. The algebraic equations g are obtained as the sum of all active and reactive power injections at buses

$$g(x,y,p) = \begin{bmatrix} g_p \\ g_p \end{bmatrix} = \begin{bmatrix} g_{pm} \\ g_{qm} \end{bmatrix} - \sum_{c \in \mathcal{C}_m} \begin{bmatrix} g_{pc} \\ g_{qc} \end{bmatrix} \quad \forall m \in \mathcal{M} \tag{2}$$

where g_{pm} and g_{qm} are the power flows in transmission lines as commonly defined in the literature [15], $\mathcal{M}$ is the set of network buses, $\mathcal{C}_m$ and $[g_{pc}^T, g_{qc}^T]^T$ are the set and the power injections of components connected at bus m, respectively.

PSAT is component-oriented, i.e., any component is defined independently of the rest of the program as a set of nonlinear differential-algebraic equations, as follows:

$$\begin{aligned}\dot{x}_c &= f_c(x_c, y_c, p_c)\\ P_c &= g_{pc}(x_c, y_c, p_c)\\ Q_c &= g_{qc}(x_c, y_c, p_c)\end{aligned} \tag{3}$$

where x_c are the component state variables, y_c the algebraic variables (i.e., V and θ at the buses to which the component is connected) and p_c are independent variables. Then differential equations f in (1) are built concatenating f_c of all components.

Equations (3) along with Jacobians matrices are defined in a function which is used for both static and dynamic analyzes. In addition to this function, a component is defined by means of a structure, which contains data, parameters and the interconnection to the grid.

For the sake of clarity, let us consider the following example, namely the exponential recovery load (ERL) [14]. The set of differential-algebraic equations are as follows:

$$\begin{aligned}\dot{x}_{c_1} &= -\frac{x_{c_1}}{T_P} + P_0\left(\frac{V}{V_0}\right)^{\alpha_s} - P_0\left(\frac{V}{V_0}\right)^{\alpha_t}\\ \dot{x}_{c_2} &= -\frac{x_{c_2}}{T_Q} + Q_0\left(\frac{V}{V_0}\right)^{\beta_s} - Q_0\left(\frac{V}{V_0}\right)^{\beta_t}\\ P_c &= \frac{x_{c_1}}{T_P} + P_0\left(\frac{V}{V_0}\right)^{\alpha_t}\\ Q_c &= \frac{x_{c_2}}{T_Q} + Q_0\left(\frac{V}{V_0}\right)^{\beta_t}\end{aligned} \tag{4}$$

TABLE III
Exponential Recovery Load Data Format (`Erload.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	-

where most parameters are defined in Table III and P_0, Q_0 and V_0 are initial powers and voltages, respectively, as given by the power flow solution. Observe that a constant PQ load must be connected at the same bus as the ERL to determine the values of P_0, Q_0, and V_0.

Exponential recovery loads are defined in the structure `Erload`, whose fields are as follows:

1) `con`: exponential recovery load data.
2) `bus`: Indexes of buses to which the ERLs are connected.
3) `dat`: Initial powers and voltages (P_0, Q_0 and V_0).
4) `n`: Total number of ERLs.
5) `xp`: Indexes of the state variable x_{c_1}.
6) `xq`: Indexes of the state variable x_{c_2}.

B. Power Flow

PSAT included the standard NR method [15], the fast decoupled power flow (XB and BX variations [16]), and a power flow with a distributed slack bus model [17]. The latter is a novelty among Matlab-based power system softwares. The power flow problem is formulated as (1) with zero first time derivatives $\dot{x}$

$$\begin{aligned}0 &= f(x,y)\\ 0 &= g(x,y).\end{aligned} \tag{5}$$

Differential equations are included in (5) although some dynamic components are initialized after power flow analysis. This is needed if the known input data of the component are not the input parameters of its dynamic model. For example, the user does not generally know field voltages and mechanical torques of synchronous machines. However the user does know desired voltages and active powers injected into the network by generators. Thus, one can solve the power flow first, using PV buses and then initialize synchronous machine state variables using the power flow solution. Nevertheless, other components can be included in the power flow as one typically knows the input parameters of the dynamic model. For example, in the case of load tap changers, it is likely the user knows the regulator reference voltage rather than the transformer tap ratio.

The distributed slack bus model is based on a generalized power center concept and consists in distributing losses among all generators [17]. This is obtained by rewriting active powers P_G of slack and PV generators as

$$P_G = (1 + k_G\gamma)P_{G_0} \tag{6}$$

where P_{G_0} are the desired generator active powers, k_G is a scalar variable which distributes power losses among all generators and γ are the participation factors of the generators to

the losses. Observe that k_G is an unknown insofar as losses are unknown. Assuming that (6) has been written for all generators, k_G is balanced by the phase reference equation.

C. Continuation Power Flow (CPF)

The CPF function included in PSAT is a novelty among available Matlab-based packages for power system analysis. The CPF algorithm consists in a predictor step which computes a normalized tangent vector and a corrector step that can be obtained either by means of a local parametrization or a perpendicular intersection [18]. The CPF problem is defined based on (1), as follows:

$$\begin{aligned} 0 &= f(x, y, \lambda) \\ 0 &= g(x, y, \lambda) \end{aligned} \tag{7}$$

where $\lambda \in \mathbb{R}$ is the *loading parameter*, which is used to vary base case generator and load powers, P_{G_0}, P_{L_0} and Q_{L_0} respectively, as follows:

$$\begin{aligned} P_G &= (\lambda + \gamma k_G) P_{G_0} \\ [P_L, Q_L] &= \lambda [P_{L_0}, Q_{L_0}]. \end{aligned} \tag{8}$$

D. Optimal Power Flow (OPF)

The OPF is defined as a nonlinear constrained optimization problem. The Interior Point Method (IPM) with a Mehrotra's predictor-corrector method is used to solve the OPF problem [19]. Notice that PSAT is the only Matlab-based software which provides an IPM algorithm to solve the OPF-based market clearing problem. A variety of objective functions are included in PSAT, as follows.

1) Market Clearing Procedure: The "standard" OPF-based market model is represented in PSAT as follows:

$$\begin{aligned} \text{Minimize}_{(y,p)} \quad & F(p) \\ \text{subject to} \quad & g(y, p) = 0 \\ & h_{\min} \le h(y) \le h_{\max} \\ & p_{\min} \le p \le p_{\max} \end{aligned} \tag{9}$$

where g and y are defined as in (1), the control variables p are the power demand and supply bids P_D and P_S, while $F : \mathbb{R}^\ell \mapsto \mathbb{R}$ and $h : \mathbb{R}^m \mapsto \mathbb{R}^q$ are the objective function and the inequality constraints, respectively.

The goal is to maximize the social benefit; thus, the objective function F is defined as

$$F = -\left(\sum_i C_{D_i}(P_{D_i}) - \sum_i C_{S_i}(P_{S_i}) \right) \tag{10}$$

where C_S and C_D are quadratic functions of supply and demand bids in \$/MWh, respectively.

The physical and security limits h included in PSAT are similar to what is used in [20], and take into account transmission line thermal limits, transmission line power flow limits, generator reactive power limits, and voltage "security" limits.

2) VSC-OPF Market Clearing Model: The following optimization problem is used for representing an OPF market clearing model with inclusion of voltage stability constraints, based on what was proposed in [21] and [22], as follows:

$$\begin{aligned} \text{Minimize}_{(y,p,\hat{y},\lambda)} \quad & f(p, \lambda) \\ \text{subject to} \quad & g(y, p) = 0 \\ & \hat{g}(\hat{y}, p, \lambda) = 0 \\ & \lambda \ge \hat{\lambda} \\ & h_{\min} \le h(y) \le h_{\max} \\ & \hat{h}_{\min} \le h(\hat{y}) \le \hat{h}_{\max} \\ & p_{\min} \le p \le p_{\max}. \end{aligned} \tag{11}$$

In (11), a second set of power flow variables $\hat{x} \in \mathbb{R}^m$ and equations $\hat{g} : \mathbb{R}^m \times \mathbb{R}^\ell \times \mathbb{R} \mapsto \mathbb{R}^m$, together with the constraints $h(\hat{x}) : \mathbb{R}^m \mapsto \mathbb{R}^q$, are introduced to represent the solution associated with a loading parameter λ, where λ represents an increase in generator and load powers, as follows:

$$\begin{aligned} \hat{P}_G &= (1 + \lambda + \hat{k}_G) P_G \\ \hat{P}_L &= (1 + \lambda) P_L \end{aligned} \tag{12}$$

where P_G and P_L are total generator and load powers for the current market condition.

Two objective functions are available: the maximization of the distance to the maximum loading condition

$$F = -\lambda \tag{13}$$

and a multi-objective objective function

$$F = -\omega \left(\left(\sum_i C_{D_i}(P_{D_i}) - \sum_i C_{S_i}(P_{S_i}) \right) \right) - (1 - \omega)\lambda \tag{14}$$

where $\omega \in (0, 1)$ is a factor which allows weighting the influence of the system security on the market clearing procedure.

E. Small-Signal Stability Analysis

PSAT allows computing and plotting the eigenvalues and the participation factors of the system, once the power flow has been solved. The eigenvalues can be computed for the state matrix of the dynamic system, and for the power flow Jacobian matrix (QV sensitivity analysis) [23]. Unlike other softwares, such as PST and Simulink-based tools, eigenvalues are computed using analytical Jacobian matrices, thus ensuring high-precision results.

1) Dynamic Analysis: The Jacobian matrix A_C of a dynamic system is defined by linearizing (5), as follows:

$$\begin{bmatrix} \Delta \dot{x} \\ 0 \end{bmatrix} = \begin{bmatrix} F_x & F_y \\ G_x & J_{LFV} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = [A_C] \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \tag{15}$$

where $F_x = \nabla_x f$, $F_y = \nabla_y f$, $G_x = \nabla_x g$, and $J_{LFV} = \nabla_y g$. Then the state matrix A_S is obtained by eliminating Δy, and thus implicitly assuming that J_{LFV} is nonsingular (i.e., no singularity-induced bifurcations)

$$A_S = F_x - F_y J_{LFV}^{-1} G_x. \tag{16}$$

The computation of all eigenvalues can be a lengthy process if the dynamic order of the system is high. At this aim, PSAT allows computing a reduced number of eigenvalues based on

sparse matrix properties and eigenvalue relative values (e.g. largest or smallest magnitude, etc.). PSAT also computes participation factors using right and left eigenvector matrices [15].

2) QV Sensitivity Analysis: The QV sensitivity analysis is computed on a reduced matrix, as it was proposed in [23]. Let us assume that the power flow Jacobian matrix J_{LFV} is divided into four submatrices

$$J_{LFV} = \begin{bmatrix} J_{P\theta} & J_{PV} \\ J_{Q\theta} & J_{QV} \end{bmatrix} \tag{17}$$

Then the reduced matrix used for QV sensitivity analysis is defined as follows:

$$J_{LFVr} = J_{QV} - J_{Q\theta} J_{P\theta}^{-1} J_{PV} \tag{18}$$

where it is assumed that $J_{P\theta}$ is nonsingular [23]. Observe that the power flow Jacobian matrix used in PSAT takes into account all static and dynamic components, e.g. tap changers models, etc.

F. Time-Domain Simulation

1) Integration Methods: Two integration methods are available, i.e., backward Euler and trapezoidal rule, which are implicit A-stable algorithms and solve (1) together [simultaneous-implicit method (SI)]. This method is numerically more stable than the partitioned-explicit method, which solves differential and algebraic equations separately [15]. Observe that PSAT is currently the only Matlab-based tool, which implements a SI method for the numerical integration of (1).

2) Handling Disturbances: The commonest perturbations for transient stability analysis, i.e., faults and breaker operations, are handled by means of embedded functions. Step perturbations can be obtained by changing parameter or variable values after completing the power flow. All other disturbances can be defined through custom "perturbation" functions, which can include and modify any global structure of the system.

IV. Case Studies

This section illustrates some PSAT features for static and dynamic stability analysis by means of the IEEE 14-bus test system (authors interested in reproducing the outputs could retrieve the data from the PSAT web site [10]). All results have been obtained on Matlab 7 running on a Intel Pentium IV 2.66 GHz. Table IV depicts simulation times for the 14-bus test system. Results were double-checked by means of other software packages, namely PST [3], UWPFLOW [1], and GAMS [13].

Fig. 7 depicts the model of the IEEE 14-bus network built using the PSAT Simulink library. Once defined in the Simulink model, one can load the network in PSAT and solve the power flow. Power flow results can be displayed in a GUI (see Fig. 6) and exported to a file in several formats including Excel and LaTeX. PSAT also allows displaying bus voltages and power flows within the Simulink model of the currently loaded system (e.g. see the bus voltage report in Fig. 7). Notice that PSAT uses vectorized computations and sparse matrix functions provided by Matlab, so that computation times increase slowly as the network size increase. Table V illustrates net power flow computation times for a variety of tests network, with different solvers, namely NR method and fast decoupled power flows (both XB and BX variations). Results were obtained using the command

TABLE IV
Performance of PSAT Solvers for the IEEE 14-Bus Test System

Simulation	Elapsed Time [s]
Power flow (Newton-Raphson method)	0.0345
Continuation power flow	2.41
Optimal power flow	0.21
Small signal stability analysis	0.16
Time domain simulation ($\Delta t = 0.1$ s)	22.0

Fig. 6. GUI for power flow reports. The results refer to the IEEE 14-bus test system.

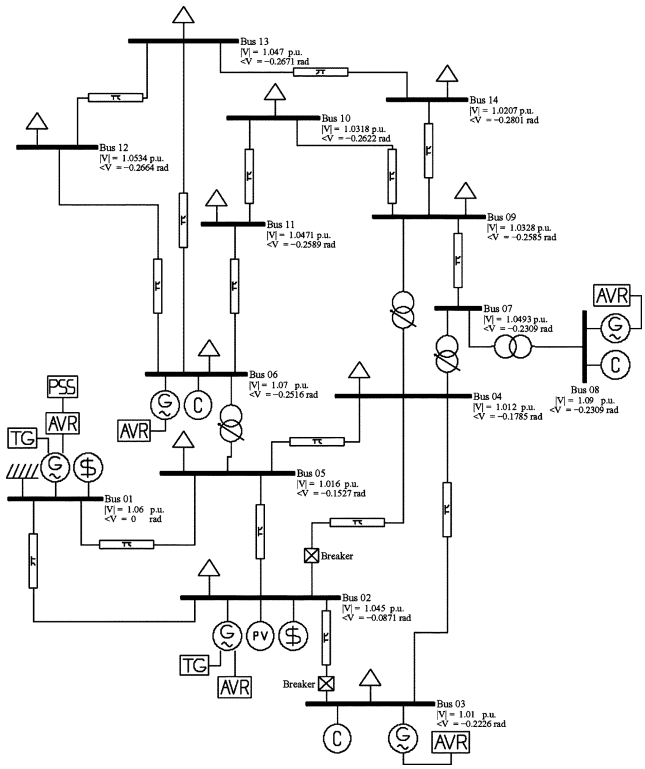

Fig. 7. PSAT-Simulink model of the IEEE 14-bus test system.

TABLE V
PERFORMANCE OF PSAT POWER FLOW SOLVERS

Network	NR [s]	XB [s]	BX [s]
IEEE 14-bus	0.0345	0.0151	0.0166
IEEE 118-bus	0.0586	0.0197	0.0173
IEEE 300-bus	0.1306	0.0447	0.0423
1228-bus (Italian HV grid)	0.6546	0.1413	0.1798

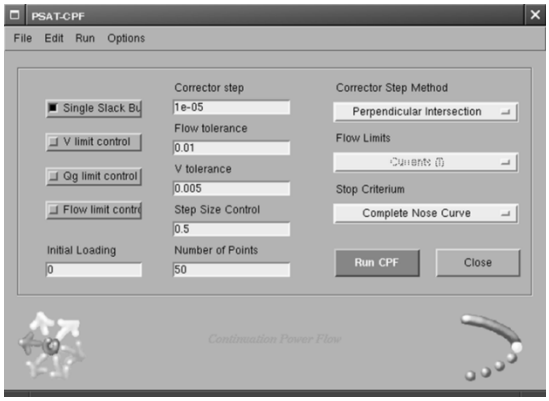

Fig. 8. GUI for continuation power flow settings.

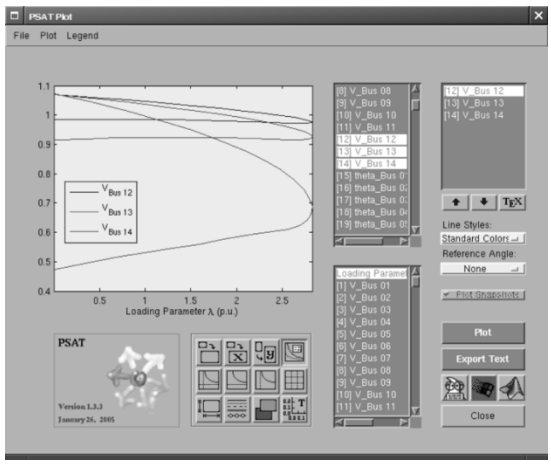

Fig. 9. GUI for plotting CPF results. The plots illustrate voltages at buses 12, 13, and 14 for the IEEE 14-bus test system with no contingency.

line version of PSAT (times are about 0.5 s slower if using GUIs).

CPF analysis is handled by a dedicated GUI, as illustrated in Fig. 8. Nose curves can be plotted using the GUI for plotting simulation results, which is depicted in Fig. 9. Fig. 10 illustrates the nose curves (V, λ_c) obtained using the CPF algorithm implemented in PSAT. The curves refers to mere static equations, i.e., the differential equations of synchronous machines and controls are ignored during the CPF analysis. Fig. 10 depicts three different nose curves considering the base case network and line 2–4 and line 2–3 outages, respectively. Notice that contingencies are simulated by setting the status of breakers as "open" in the Simulink model.

The GUI depicted in Fig. 11 allows adjusting parameters and preferences for OPF analysis. For the sake of comparison with the CPF analysis, Table VI depicts the maximum loading parameter λ^*, the base case power $(BCP = \sum_i P_{L_i})$, the Maximum

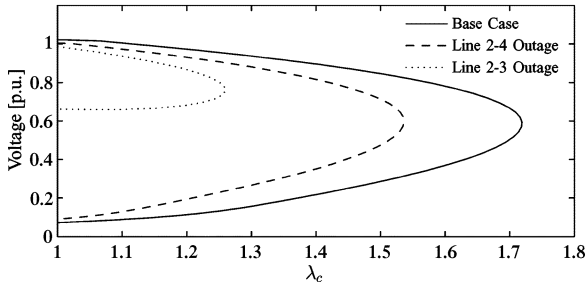

Fig. 10. Nose curves at bus 14 for different contingencies for the IEEE 14-bus test system.

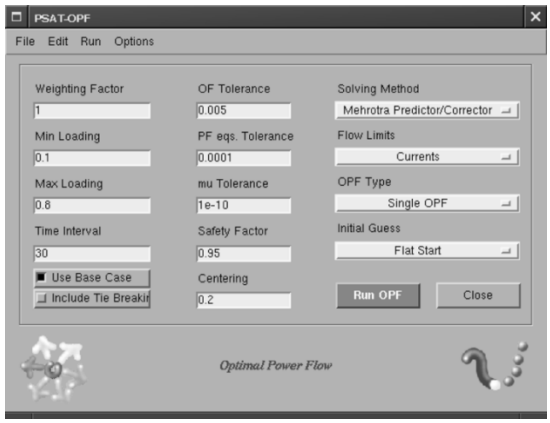

Fig. 11. GUI for OPF settings. Observe that the weighting factor is set to 1 in order to obtain the objective function (13).

TABLE VI
MAXIMUM LOADING CONDITION OPF FOR THE IEEE 14-BUS NETWORK

Contingency	BCP [MW]	λ^* [p.u.]	MLC [MW]	ALC [MW]
None	259	0.7211	445.8	186.8
Line 2-4 Outage	259	0.5427	399.5	148.6
Line 2-3 Outage	259	0.2852	332.8	73.85

Loading Condition $(MLC = (1 + \lambda^*)BCP)$, and the Available Loading Capability $(ALC = \lambda^* BCP)$ for the base case and the lines 2–3 and 2–4 outages. The OPF problem used to compute the MLC is (11) and (13). Notice that, because of the definitions of generator and load powers P_G and P_L given in (8) and (12), one has $\lambda_c = \lambda^* + 1$.

The test case presented in [24] is reproduced here to illustrate smaill-signal stability analysis and time-domain simulation available in PSAT. Firstly it has been used the IEEE 14-bus system with a 40% load increase with respect to the base case loading, and no PSS at bus 1. As illustrated by the time-domain simulation depicted in Fig. 12, a Hopf bifurcation occurs for the line 2–4 outage resulting in undamped oscillations of generator angles. A similar analysis can be carried on the same system with a 40% load increase but considering the PSS of the generator connected at bus 1. Fig. 13 depicts the GUI for eigenvalue analysis and shows that the system is stable.

V. CONCLUSIONS

This paper has presented a new open-source PSAT which runs on Matlab and GNU/Octave. PSAT comes with a variety

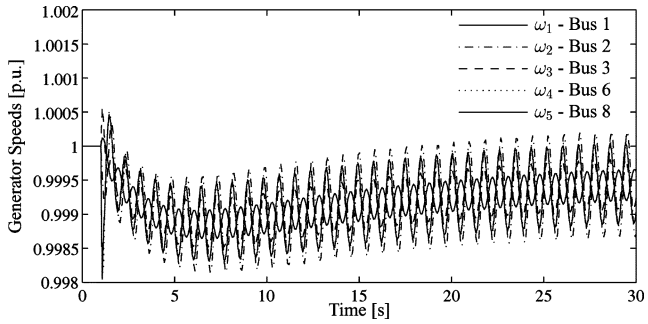

Fig. 12. Generator speed oscillations for the IEEE 14-bus test system due to Hopf bifurcation triggered by line outage at 40% overload.

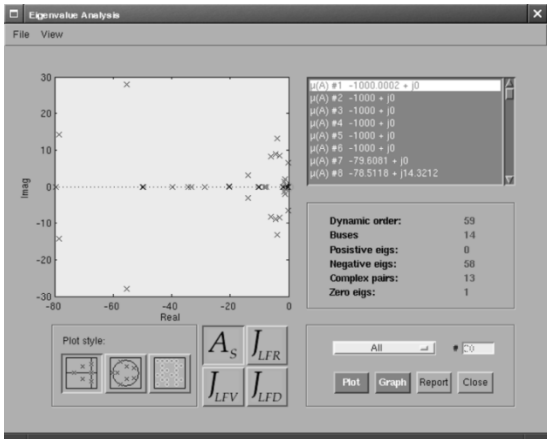

Fig. 13. GUI for eigenvalue analysis. The plot illustrates eigenvalues for the IEEE 14-bus test system with PSS, for a line 2–4 outage at 40% overload.

of procedures for static and dynamic analysis, several models of standard and unconventional devices, a complete GUI, and a Simulink-based network editor. These features make PSAT suited for both educational and research purposes. As a matter of fact, PSAT is currently used by several undergraduates, Ph.D. students, and researchers, and has an active mailing list (http://groups.yahoo.com/groups/psatforum) currently counting over 450 members. Among future projects, there are extending the CPF algorithm to dynamic bifurcation analysis and including new control schemes and renewable energy generator models. Any suggestion and/or bug report are very welcome.

REFERENCES

[1] UWPFLOW, Continuation and Direct Methods to Locate Fold Bifurcations in AC/DC/FACTS Power Systems, C. A. Cañizares and F. L. Alvarado. (1999). *http://www.power.uwaterloo.ca* [Online]

[2] M. Larsson, "ObjectStab—an educational tool for power system stability studies," *IEEE Trans. Power Syst.*, vol. 19, no. 1, pp. 56–63, Feb. 2004.

[3] J. H. Chow and K. W. Cheung, "A toolbox for power system dynamics and control engineering education and research," *IEEE Trans. Power Syst.*, vol. 7, no. 4, pp. 1559–1564, Nov. 1992.

[4] R. D. Zimmerman, C. E. Murrillo-Sánchez, and D. Gan. (2005) Matpower, Version 3.0.0, User's Manual. Power System Engineering Research Center, Cornell Univ., Ithaca, NY. [Online] Available: http://www.pserc.cornell.edu/matpower/matpower.html

[5] A. H. L. Chen, C. O. Nwankpa, H. G. Kwatny, and X. Yu, "Voltage stability toolbox: an introduction and implementation," in *Proc. 28th North American Power Symp.*, Nov. 1996.

[6] J. Mahseredjian and F. Alvarado, "Creating an electromagnetic transient program in MATLAB: MatEMTP," *IEEE Trans. Power Delivery*, vol. 12, no. 1, pp. 380–388, Jan. 1997.

[7] G. Sybille. (2004, Oct.) SimPowerSystems User's Guide, Version 4. published under sublicense from Hydro-Québec, and The MathWorks, Inc.. [Online] Available at: http://www.mathworks.com.

[8] K. Schoder, A. Hasanović, A. Feliachi, and A. Hasanović, "PAT: a power analysis toolbox for MATLAB/Simulink," *IEEE Trans. Power Syst.*, vol. 18, no. 1, pp. 42–47, Feb. 2003.

[9] C. D. Vournas, E. G. Potamianakis, C. Moors, and T. Van Cutsem, "An educational simulation tool for power system control and stability," *IEEE Trans. Power Syst.*, vol. 19, no. 1, pp. 48–55, Feb. 2004.

[10] F. Milano. (2002) PSAT, Matlab-Based Power System Analysis Toolbox. [Online] Available at: http://thunderbox.uwaterloo.ca/~fmilano

[11] R. M. Stallman, *Free Software, Free Society: Selected Essays of Richard M. Stallman*. Boston, MA: Free Software Foundation, 2002.

[12] J. W. Eaton, *GNU Octave Manual*. Bristol, U.K.: Network Theory Ltd., 1997.

[13] A. Brooke, D. Kendrick, A. Meeraus, R. Raman, and R. E. Rosenthal. (1998, Dec.) GAMS, a User's Guide. GAMS Development Corporation. [Online] Available at: http://www.gams.com/

[14] D. Karlsson and D. J. Hill, "Modeling and identification of nonlinear dynamic loads in power systems," *IEEE Trans. Power Syst.*, vol. 9, no. 1, pp. 157–166, Feb. 1994.

[15] P. W. Sauer and M. A. Pai, *Power System Dynamics and Stability*. Upper Saddle River, NJ: Prentice-Hall, 1998.

[16] R. A. M. van Amerongen, "A general-purpose version of the fast decoupled loadflow," *IEEE Trans. Power Syst.*, vol. 4, no. 2, pp. 760–770, May 1989.

[17] W. R. Barcelo and W. W. Lemmon, "Standardized sensitivity coefficients for power system networks," *IEEE Trans. Power Syst.*, vol. 3, no. 4, pp. 1591–1599, Nov. 1988.

[18] C. A. Cañizares, Ed., "Voltage Stability Assessment: Concepts, Practices and Tools," IEEE/FES Power System Stability Subcommittee, Final Document, Tech. Rep., Aug. 2002.

[19] G. L. Torres and V. H. Quintana, "Introduction to interior-point methods," in *Proc. IEEE PICA*, Santa Clara, CA, May 1999.

[20] K. Xie, Y.-H. Song, J. Stonham, E. Yu, and G. Liu, "Decomposition model and interior point methods for optimal spot pricing of electricity in deregulation environments," *IEEE Trans. Power Syst.*, vol. 15, no. 1, pp. 39–50, Feb. 2000.

[21] W. D. Rosehart, C. A. Cañizares, and V. H. Quintana, "Multi-objective optimal power flows to evaluate voltage security costs in power networks," *IEEE Trans. Power Syst.*, vol. 18, no. 2, pp. 578–587, May 2003.

[22] F. Milano, C. A. Cañizares, and M. Invernizzi, "Multi-objective optimization for pricing system security in electricity markets," *IEEE Trans. Power Syst.*, vol. 18, no. 2, May 2003.

[23] G. K. Morison, B. Gao, and P. Kundur, "Voltage stability analysis using static and dynamic approaches," *IEEE Trans. Power Syst.*, vol. 8, no. 3, pp. 1159–1171, Aug. 1993.

[24] S. K. Mena Kodsi and C. A. Cañizares. (2003) Modeling and Simulation of IEEE 14 Bus System With FACTS Controllers. Elect. Comput. Eng. Dept., Univ. Waterloo, Waterloo, ON, Canada. [Online] Available: http://www.power.uwaterloo.ca

Federico Milano (M'03) received the Elect. Eng. degree and the Ph.D. degree in electrical engineering in 1999 and 2003, respectively, both from the University of Genoa, Genoa, Italy.

From September 2001 to December 2002, he was with the Electrical and Computer Engineering Department, University of Waterloo, Waterloo, ON, Canada, as a Visiting Scholar. He is currently an Assistant Professor of Electrical Engineering at the University of Castilla-La Mancha, Ciudad Real, Spain. His research interests are voltage stability, electricity markets, and computer-based power system analysis and control.