Semi-implicit Formulation of Differential-Algebraic Equations for Transient Stability Analysis

F. Milano, *Fellow, IEEE*

Abstract— **The paper proposes a semi-implicit formulation of the differential-algebraic equations (DAEs) describing power system models for transient stability analysis. This formulation, if coupled to an implicit integration scheme, shows two relevant advantages with respect to the conventional explicit formulation: (i) reduce the computational burden; and (ii) increase the sparsity of the Jacobian matrix of the system. The proposed model also allows using null time constants and thus simplifies the computer implementation of the DAEs. The properties and the performance of the proposed semi-implicit formulation and the conventional explicit one are compared through a dynamic 21,177-bus model of the European transmission system.**

Index Terms— **Transient stability analysis, differential-algebraic equations (DAEs), implicit time integration scheme.**

I. Introduction

A. Motivation

POWER systems models for transient stability analysis are traditionally defined as a set of explicit differential-algebraic equations (DAEs) [1]–[4]:

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

where $\boldsymbol{f}$ are the differential equations, $\boldsymbol{g}$ are the algebraic equations, $\boldsymbol{x}$ are the state variables, and $\boldsymbol{y}$ are the algebraic variables.

Equations (1) are ubiquitous in the literature on the dynamic analysis of power systems. The explicit formulation, however, is not the only possible formulation of DAEs. This paper shows that (1) are not the most efficient for the numerical solution of time domain integration and proposes the following semi-implicit form:

$$\begin{aligned} \boldsymbol{T}(\boldsymbol{x}, \boldsymbol{y})\dot{\boldsymbol{x}} &= \tilde{\boldsymbol{f}}(\boldsymbol{x}, \boldsymbol{y}) \\ \boldsymbol{R}(\boldsymbol{x}, \boldsymbol{y})\dot{\boldsymbol{x}} &= \tilde{\boldsymbol{g}}(\boldsymbol{x}, \boldsymbol{y}) \end{aligned} \tag{2}$$

where $\boldsymbol{T}(\boldsymbol{x}, \boldsymbol{y})$ and $\boldsymbol{R}(\boldsymbol{x}, \boldsymbol{y})$ are time-variant, not necessarily diagonal nor full-rank matrices. Equations (2) provide several advantages with respect to (1), namely, reduce the number of operations to compute equations and the elements of the Jacobian matrix of the DAE; increase the sparsity of the Jacobian matrix of the DAE; and allow effortlessly switching state variables to algebraic ones.

Federico Milano is with the School of Electrical and Electronic Engineering, University College Dublin, Ireland (e-mail: federico.milano@ucd.ie).

This paper has emanated with the financial support of EC Marie Skłodowska-Curie Career Integration Grant No. PCIG14-GA-2013-630811.

B. Literature Review

As said above, (1) is the most common formulation and is used in all textbooks and seminal works on transient stability analysis. The literature based on (1) spreads in several directions. Relevant topics are, to cite some, stability analysis [5], [6]; inverse problems [7]; methods to improve computational efficiency [8], [9]; parallelization techniques [10]–[13]; variable time step methods [14], [15]; implicit integration schemes [16], [17]; multi-time scales and integration of power systems and electronic circuits [18], [19]; methods for long term dynamic models [20], [21]; modified augmented nodal analysis (MANA) [22]; and quasi-static simulation [23], [24]. Moreover, all most important proprietary software tools for power system analysis are based on the explicit formulation (1). Thanks to its generality, the proposed semi-implicit formulation (2) can be directly applied to all numerical techniques considered above.

To the best of the knowledge of the author, there are no previous examples in the literature of implicit or semi-implicit formulations of power systems similar to (2). The closest form is that proposed in [12], where state variable dynamics can be switched off by means of a diagonal matrix that pre-multiplies the $\dot{\boldsymbol{x}}$ vector. However, in [12], the authors use an explicit formulation. Section II shows that the formulation given in [12] is a special case of the one proposed in this paper.

C. Contributions

The novel contributions of the paper are twofold.

- A semi-implicit formulation of DAEs for power system time domain analysis. The formulation is general and can be applied to any dynamic continuous system. However, it appears particularly suited for the DAE describing HV transmission grids as the differential equations of common devices and controllers used in power systems result simplified if written according to the proposed semi-implicit formulation.
- A detailed description on how to numerically exploit the features of the proposed semi-implicit formulation through implicit integration schemes.

D. Paper Organization

The remainder of the paper is organized as follows. Section II provides the definitions of explicit and semi-implicit formulations and describes how the former is a special case of the latter. Implicit time integration schemes are also described in this section. Section III provides a variety of examples to illustrate the properties of the proposed semi-implicit DAE model and compares it with the conventional explicit model. Section IV presents simulation results based on a dynamic 21,177-bus model of the European ENTSO-E transmission system. Conclusions are drawn in Section V.

II. Differential-Algebraic Equations for Power System Modelling

The conventional explicit DAE model for transient stability analysis is recalled in Subsection II-A. The proposed semi-implicit formulation is presented and duly discussed in Subsection II-B. Subsections II-C and II-D discuss the numerical advantages of the proposed technique when it is coupled to an implicit integration scheme. Finally, Subsections II-E and II-F provide a quick overview on the quasi-steady-state analysis and on how the proposed semi-implicit formulation affects the computation of the state matrix of the system, respectively.

A. Conventional Explicit DAE Formulation

The set of explicit DAEs (1) can be written in a slightly more general form considering discrete events. The following model is considered in the remainder of this paper [7]:

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}) \\ \boldsymbol{0} &= \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}) \end{aligned} \tag{3}$$

where $\boldsymbol{f}$ ($\boldsymbol{f} : \mathbb{R}^{n+m+p} \mapsto \mathbb{R}^n$) are the differential equations, $\boldsymbol{g}$ ($\boldsymbol{g} : \mathbb{R}^{n+m+p} \mapsto \mathbb{R}^m$) are the algebraic equations, $\boldsymbol{x}$ ($\boldsymbol{x} \in \mathbb{R}^n$) are the state variables, $\boldsymbol{y}$ ($\boldsymbol{y} \in \mathbb{R}^m$) are the algebraic variables, and $\boldsymbol{u}$ ($\boldsymbol{u} \in \mathbb{R}^p$) are discrete variables modeling events, e.g., line outages and faults. In common practice, equations (3) are split into a collection of subsystems where discrete variables $\boldsymbol{u}$ are substituted for *if-then* rules. Thus, (3) can be conveniently rewritten as a finite collection of continuous DAEs, one per each discrete variable change (see the definition of the *hybrid automaton* given in [7]).

B. Proposed Semi-Implicit DAE Formulation

The proposed semi-implicit DAE model with inclusion of discrete events is as follows:

$$\begin{aligned} \boldsymbol{T}(\boldsymbol{x}, \boldsymbol{y})\dot{\boldsymbol{x}} &= \tilde{\boldsymbol{f}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}) \\ \boldsymbol{R}(\boldsymbol{x}, \boldsymbol{y})\dot{\boldsymbol{x}} &= \tilde{\boldsymbol{g}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}) \end{aligned} \tag{4}$$

where $\boldsymbol{T}(\boldsymbol{x}, \boldsymbol{y})$ and $\boldsymbol{R}(\boldsymbol{x}, \boldsymbol{y})$ are $n \times n$ and $m \times n$ matrices. In general, as stated in the introduction, $\boldsymbol{T}$ and $\boldsymbol{R}$ are time-variant, non-diagonal and non-full rank. Equations $\tilde{\boldsymbol{f}}$ ($\tilde{\boldsymbol{f}} : \mathbb{R}^{n+m+p} \mapsto \mathbb{R}^n$) are the explicit part of differential equations, while $\tilde{\boldsymbol{g}}$ ($\tilde{\boldsymbol{g}} : \mathbb{R}^{n+m+p} \mapsto \mathbb{R}^m$) are the explicit part of algebraic equations. Clearly, $\tilde{\boldsymbol{f}} \neq \boldsymbol{f}$ and $\tilde{\boldsymbol{g}} \neq \boldsymbol{g}$ unless $\boldsymbol{T} = \boldsymbol{I}_n$, where $\boldsymbol{I}_n$ is the identity matrix of order n, and $\boldsymbol{R} = \boldsymbol{0}$.

The explicit model (3) can be obtained from (4) if and only if $\boldsymbol{T}$ is full rank. Then, one has:

$$\begin{aligned} \boldsymbol{f} &= \boldsymbol{T}^{-1}\tilde{\boldsymbol{f}} \\ \boldsymbol{g} &= \tilde{\boldsymbol{g}} - \boldsymbol{R} \cdot \boldsymbol{T}^{-1}\tilde{\boldsymbol{f}} \end{aligned} \tag{5}$$

where the dependence on $\boldsymbol{x}$, $\boldsymbol{y}$ and $\boldsymbol{u}$ has been omitted for simplicity.

In some recent papers, e.g., [12], the following formulation is proposed:

$$\begin{aligned} \boldsymbol{\Gamma}\dot{\boldsymbol{x}} &= \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}) \\ \boldsymbol{0} &= \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}) \end{aligned} \tag{6}$$

where $\boldsymbol{\Gamma}$ is a $n \times n$ time-invariant diagonal matrix. The elements of $\boldsymbol{\Gamma}$, say $\gamma_{i,i}$, are as follows:

$$\begin{cases} \gamma_{i,i} = 1, & \text{if the dynamic of } x_i \text{ is retained,} \\ \gamma_{i,i} = 0, & \text{otherwise.} \end{cases}$$

Equations (6), however, are explicit and their only added value with respect to (3) is the ability to transform state variables into algebraic ones. This feature is a byproduct of (4); (6), in fact, is a particular case of (4).

Equation (4) could be rewritten in a more general way by defining a total vector of variable $\boldsymbol{z} = [\boldsymbol{x}^T, \boldsymbol{y}^T]^T$:

$$\boldsymbol{\Xi}(\boldsymbol{z})\dot{\boldsymbol{z}} = \boldsymbol{F}(\boldsymbol{z}, \boldsymbol{u}) \tag{7}$$

where $\boldsymbol{F} = [\tilde{\boldsymbol{f}}^T, \tilde{\boldsymbol{g}}^T]^T$ and

$$\boldsymbol{\Xi}(\boldsymbol{z}) = \begin{bmatrix} \boldsymbol{T}(\boldsymbol{z}) & \boldsymbol{0} \\ \boldsymbol{R}(\boldsymbol{z}) & \boldsymbol{0} \end{bmatrix} \tag{8}$$

Equation (7) is a compact notation and is the preferred one in mathematical monographs (see, for example, [25]). Based on (7), it is straightforward to define the fully implicit form of differential equation, as follows:

$$\boldsymbol{0} = \boldsymbol{\Phi}(\dot{\boldsymbol{z}}, \boldsymbol{z}, \boldsymbol{u}) \tag{9}$$

The implicit forms (7) and (9) indicate that the distinction between state and algebraic variables, namely $\boldsymbol{x}$ and $\boldsymbol{y}$, is actually unnecessary from the mathematical viewpoint. In the remainder of this paper, however, the notation of (4) is preferred because of the following considerations:

- Equations (4) – similarly to (6) – allow distinguishing between variables that always have infinitely fast dynamics, i.e., algebraic variables, and state variables, whose time constants can be null in occasions. From an engineering point of view this distinction can help understand the physical meaning and behaviour of the variables. For example, in the standard transient stability analysis, bus voltage phasors are algebraic variables and are conceptually different from, say, synchronous machine rotor fluxes. The latter, in fact, are associated with differential equations whose dynamics can be enabled or not. On the other hand, the dynamics of bus voltage phasors, in particular if using polar coordinates, cannot be easily recovered – unless the admittance matrix model is dropped and the transmission system is modelled using dynamic electromagnetic equations. The difference between algebraic and state variables is also evident if one considers extra algebraic constraints as in the MANA approach [22] (see also Subsection II-D).
- The formulation as in (4) allows for a simpler comparison with the standard explicit model (3). For the same reason, the authors of [12] proposed (6).

- Equations (4) suggest how to move from the explicit formulation (3) to the semi-explicit one. This can be useful in case an interested reader decides to modify a software tool based on (3) and implement the semi-explicit formulation. Equations (4) also indicate that (3) is a special case of (4). The explicit formulation can, in fact, coexist with the semi-implicit one (see also Subsection II-D).
- Depending on the integration scheme, (4) can lead to a slightly lighter computational burden than (7), as discussed in the following subsection.

C. Implicit Integration Schemes for DAEs

Power system dynamic models are known to be *stiff* [25], i.e., time constants can span several orders of magnitude. DAEs are actually an extreme case of stiff problems, as the time constants associated with algebraic variables are null. Explicit methods such as Runge-Kutta schemes of linear multi-step methods cannot deal properly with stiffness and, for this reason, implicit schemes are the most commonly used methods for the dynamic analysis of power systems [15]–[17].

When using implicit methods, each step of the numerical integration is obtained as the solution of a set of nonlinear equations. At a generic time t, and assuming a step length h, one has to solve:

$$\begin{aligned} \mathbf{0} &= \boldsymbol{p}(\boldsymbol{x}(t+h), \boldsymbol{y}(t+h), \boldsymbol{u}(t+h), h) \\ \mathbf{0} &= \boldsymbol{q}(\boldsymbol{x}(t+h), \boldsymbol{y}(t+h), \boldsymbol{u}(t+h), h) \end{aligned} \tag{10}$$

where $\boldsymbol{p}$, ($\boldsymbol{p} : \mathbb{R}^{n+m+p} \mapsto \mathbb{R}^n$) and $\boldsymbol{q}$, ($\boldsymbol{q} : \mathbb{R}^{n+m+p} \mapsto \mathbb{R}^m$) are nonlinear functions that depend on the DAE and on the implicit numerical method. In particular, $\boldsymbol{p}$ accounts for differential equations, while $\boldsymbol{q}$ for algebraic ones.

Since (10) are nonlinear, their solution is generally obtained by means of a direct solver, e.g., Newton's method, which, in turn, consists in computing iteratively the increments $\Delta \boldsymbol{x}^{(i)}$ and $\Delta \boldsymbol{y}^{(i)}$ and updating state and algebraic variables:

$$\begin{aligned} \begin{bmatrix} \Delta \boldsymbol{x}^{(i)} \\ \Delta \boldsymbol{y}^{(i)} \end{bmatrix} &= -[\boldsymbol{A}^{(i)}]^{-1} \begin{bmatrix} \boldsymbol{p}^{(i)} \\ \boldsymbol{q}^{(i)} \end{bmatrix} \\ \begin{bmatrix} \boldsymbol{x}^{(i+1)}(t+h) \\ \boldsymbol{y}^{(i+1)}(t+h) \end{bmatrix} &= \begin{bmatrix} \boldsymbol{x}^{(i)}(t+h) \\ \boldsymbol{y}^{(i)}(t+h) \end{bmatrix} + \begin{bmatrix} \Delta \boldsymbol{x}^{(i)} \\ \Delta \boldsymbol{y}^{(i)} \end{bmatrix} \end{aligned} \tag{11}$$

where $\boldsymbol{A}^{(i)}$ is the Jacobian matrix of (10), as follows:

$$\boldsymbol{A}^{(i)} = \begin{bmatrix} \boldsymbol{p}_{\boldsymbol{x}}^{(i)} & \boldsymbol{p}_{\boldsymbol{y}}^{(i)} \\ \boldsymbol{q}_{\boldsymbol{x}}^{(i)} & \boldsymbol{q}_{\boldsymbol{y}}^{(i)} \end{bmatrix} \tag{12}$$

To understand the discussion below, it is important to note that time derivatives $\dot{x}$ **do not appear explicitly in (10) and (11)**. The proposed semi-implicit formulation is based on this observation, which has a relevant impact on the computational burden of (10), on the sparsity pattern of the Jacobian matrix $\boldsymbol{A}^{(i)}$ in (12) and, in turn, on the performance of the solution of the numerical integration. This is illustrated through some examples of implicit method schemes. To simplify the notation of the following subsections, the dependence on $t+h$ and the iteration index i are omitted. So, for example, $\boldsymbol{x} \equiv \boldsymbol{x}^{(i)}(t+h)$.

1) Backward Euler Method (BEM):

The BEM is a first order implicit method. It is often used to restart multi-step methods such as those based on backward differentiation formulæ. Applying the BEM to the conventional DAE formulation (3) leads to:

$$\begin{aligned} \boldsymbol{p} &= \boldsymbol{\xi} - h\boldsymbol{f} \\ \boldsymbol{q} &= -h\boldsymbol{g} \end{aligned} \tag{13}$$

where $\boldsymbol{\xi} = \boldsymbol{x} - \boldsymbol{x}(t)$, and $\boldsymbol{x}(t)$ is the known vector of state variables computed at the previous step. The expression of $\boldsymbol{q}$ is obtained by scaling the algebraic equations through the step length h. While this is not strictly necessary when using the explicit form (3), scaling the algebraic equations $\boldsymbol{g}$ can reduce the number of iterations required to solve (11). From (13), one can readily obtain:

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{I}_n - h\boldsymbol{f}_{\boldsymbol{x}} & -h\boldsymbol{f}_{\boldsymbol{y}} \\ -h\boldsymbol{g}_{\boldsymbol{x}} & -h\boldsymbol{g}_{\boldsymbol{y}} \end{bmatrix} \tag{14}$$

As observed above, $\dot{x}$ does not appear explicitly in (13). The time derivatives, in fact, are approximated with a finite difference, namely $\dot{x} \approx \boldsymbol{\xi}/h$.

The BEM applied to the proposed semi-implicit form (4) leads to the following expression:

$$\begin{aligned} \tilde{\boldsymbol{p}} &= \boldsymbol{T} \cdot \boldsymbol{\xi} - h\tilde{\boldsymbol{f}} \\ \tilde{\boldsymbol{q}} &= \boldsymbol{R} \cdot \boldsymbol{\xi} - h\tilde{\boldsymbol{g}} \end{aligned} \tag{15}$$

and

$$\tilde{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{T} + \boldsymbol{T}_{\boldsymbol{x}}\boldsymbol{\xi} - h\tilde{\boldsymbol{f}}_{\boldsymbol{x}} & \boldsymbol{T}_{\boldsymbol{y}}\boldsymbol{\xi} - h\tilde{\boldsymbol{f}}_{\boldsymbol{y}} \\ \boldsymbol{R} + \boldsymbol{R}_{\boldsymbol{x}}\boldsymbol{\xi} - h\tilde{\boldsymbol{g}}_{\boldsymbol{x}} & \boldsymbol{R}_{\boldsymbol{y}}\boldsymbol{\xi} - h\tilde{\boldsymbol{g}}_{\boldsymbol{y}} \end{bmatrix} \tag{16}$$

where $\boldsymbol{T}_{\boldsymbol{x}}$ and $\boldsymbol{T}_{\boldsymbol{y}}$ are the gradients of $\boldsymbol{T}$ with respect to $\boldsymbol{x}$ and $\boldsymbol{y}$, respectively, and $\boldsymbol{R}_x$ and $\boldsymbol{R}_y$ are defined similarly. Note that, in this case, scaling the expression of $\boldsymbol{q}$ with the step length h is mandatory, otherwise the term $\boldsymbol{R}\boldsymbol{\xi}$ does not approximate $\boldsymbol{R}\dot{\boldsymbol{x}}/h$.

2) Implicit Trapezoidal Method (ITM):

The Crank-Nicolson's or ITM is the workhorse solver for electro-mechanical DAE, and is widely used, in a variety of flavors, in most commercial and non-commercial power system software packages. The ITM has proved to be very robust and reliable for a variety of stiff ODE and DAE systems. Using the same notation as for the BEM, $\boldsymbol{p}$ are $\boldsymbol{q}$ for the ITM are:

$$\begin{aligned} \boldsymbol{p} &= \boldsymbol{\xi} - 0.5h(\boldsymbol{f} + \boldsymbol{f}(t)) \\ \boldsymbol{q} &= -h\boldsymbol{g} \end{aligned} \tag{17}$$

$\boldsymbol{f}(t)$ is the known vector of state variable derivatives computed at the previous step. The Jacobian matrix of (17) is:

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{I}_n - 0.5h\boldsymbol{f}_{\boldsymbol{x}} & -0.5h\boldsymbol{f}_{\boldsymbol{y}} \\ -h\boldsymbol{g}_{\boldsymbol{x}} & -h\boldsymbol{g}_{\boldsymbol{y}} \end{bmatrix} \tag{18}$$

The expressions of $\tilde{\boldsymbol{p}}$ and $\tilde{\boldsymbol{q}}$ for the semi-explicit formulation are straightforwardly obtained:

$$\begin{aligned} \tilde{\boldsymbol{p}} &= \boldsymbol{T} \cdot \boldsymbol{\xi} - 0.5h(\tilde{\boldsymbol{f}} + \tilde{\boldsymbol{f}}(t)) \\ \tilde{\boldsymbol{q}} &= \boldsymbol{R} \cdot \boldsymbol{\xi} - h\tilde{\boldsymbol{g}} \end{aligned} \tag{19}$$

and

$$\tilde{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{T} + \boldsymbol{T}_{\boldsymbol{x}}\boldsymbol{\xi} - 0.5h\tilde{\boldsymbol{f}}_{\boldsymbol{x}} & \boldsymbol{T}_{\boldsymbol{y}}\boldsymbol{\xi} - 0.5h\tilde{\boldsymbol{f}}_{\boldsymbol{y}} \\ \boldsymbol{R} + \boldsymbol{R}_{\boldsymbol{x}}\boldsymbol{\xi} - h\tilde{\boldsymbol{g}}_{\boldsymbol{x}} & \boldsymbol{R}_{\boldsymbol{y}}\boldsymbol{\xi} - h\tilde{\boldsymbol{g}}_{\boldsymbol{y}} \end{bmatrix} \tag{20}$$

3) Backward Differentiation Formulæ (BDF): These are a family of implicit methods and are largely used for circuit and power system analysis due to their L-stability properties. The BDF formula of order ν for the explicit form (3) is given by [26]:

$$\begin{aligned} \boldsymbol{p} &= \tilde{\boldsymbol{\xi}} - \beta h \boldsymbol{f} \\ \boldsymbol{q} &= -h\boldsymbol{g} \end{aligned} \tag{21}$$

where

$$\tilde{\boldsymbol{\xi}} = \boldsymbol{x} - \sum_{\ell=1}^{\nu} \gamma_\ell \boldsymbol{x}(t - (\ell - 1)h) \tag{22}$$

and, for simplicity but without a substantial loss of generality, a constant step length h is assumed for $\boldsymbol{x}(t-(\ell-1)h)$ values. The Jacobian matrix of (21), is as follows:

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{I}_n - \beta h \boldsymbol{f}_{\boldsymbol{x}} & -\beta h \boldsymbol{f}_{\boldsymbol{y}} \\ -h\boldsymbol{g}_{\boldsymbol{x}} & -h\boldsymbol{g}_{\boldsymbol{y}} \end{bmatrix} \tag{23}$$

The coefficients γ_ℓ and β are computed according to a straightforward procedure given in [16].

The semi-implicit counterparts of (21) and (23) are as follows:

$$\begin{aligned} \tilde{\boldsymbol{p}} &= \boldsymbol{T} \cdot \tilde{\boldsymbol{\xi}} - \beta h (\tilde{\boldsymbol{f}} + \kappa \tilde{\boldsymbol{f}}(t)) \\ \tilde{\boldsymbol{q}} &= \boldsymbol{R} \cdot \tilde{\boldsymbol{\xi}} - h\tilde{\boldsymbol{g}} \end{aligned} \tag{24}$$

and

$$\tilde{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{T} + \boldsymbol{T}_{\boldsymbol{x}}\tilde{\boldsymbol{\xi}} - \beta h\tilde{\boldsymbol{f}}_{\boldsymbol{x}} & \boldsymbol{T}_{\boldsymbol{y}}\tilde{\boldsymbol{\xi}} - \beta h\tilde{\boldsymbol{f}}_{\boldsymbol{y}} \\ \boldsymbol{R} + \boldsymbol{R}_{\boldsymbol{x}}\tilde{\boldsymbol{\xi}} - h\tilde{\boldsymbol{g}}_{\boldsymbol{x}} & \boldsymbol{R}_{\boldsymbol{y}}\tilde{\boldsymbol{\xi}} - h\tilde{\boldsymbol{g}}_{\boldsymbol{y}} \end{bmatrix} \tag{25}$$

where the coefficient κ is included to generalize (24) and allow using it for both BDF and ITM. Table I summarizes the coefficients for the BEM, ITM and order-2 BDF. Note that, from the computer implementation point of view, expressions (24) and (25) can be used for both the implicit and semi-implicit formulations of the DAE. In fact, (21) is a particular case of (24) where $\boldsymbol{T} = \boldsymbol{I}_n$ and $\boldsymbol{R} = \boldsymbol{0}$ (see also the discussion above on (5)). The software tool Dome [27] used for obtaining the results that are discussed in Section IV is based on (24) and (25). Note also that the BEM is the order-1 BDF, as well-known, as the first order approximation is unique [28].

Implicit methods of order higher than 2 are not considered in this paper. The proposed semi-implicit formulation, however, is not limited to second-order methods. For example, the Hammer-Hollingsworth 4th order method discussed in [17] can be straightforwardly formulated using (4).

D. Remarks on Implicit Integration Methods

Semi-implicit expressions (24) and (25) may look more involved and computationally demanding than the expressions obtained using the explicit formulations (21) and (23). Actually, this is not the case in practice. The following remarks are relevant:

TABLE I
COEFFICIENTS OF THE ORDER 1 AND ORDER 2 BDF AND ITM

Scheme	Order	γ_1	γ_2	β	κ
BEM	1	1	-	1	0
BDF	2	4/3	$-1/3$	2/3	0
ITM	2	1	-	0.5	1

- Matrices $\boldsymbol{T}$ and $\boldsymbol{R}$ are very sparse and typically sparser than $\boldsymbol{f}_{\boldsymbol{x}}$ and $\boldsymbol{g}_{\boldsymbol{x}}$. $\boldsymbol{T}$ is almost diagonal while $\boldsymbol{R}$ contains only very few off-diagonal elements. Off-diagonal elements of $\boldsymbol{T}$ and $\boldsymbol{R}$, however, allow increasing the sparsity of the Jacobian matrices of $\tilde{\boldsymbol{f}}$ and $\tilde{\boldsymbol{g}}$. As a result, $\tilde{\boldsymbol{A}}$ is consistently sparser than $\boldsymbol{A}$.[1]
- In practical power system models almost all elements of $\boldsymbol{T}$ and $\boldsymbol{R}$ are constant. In (25), the terms that multiply $\tilde{\boldsymbol{\xi}}$ are null or almost all null. Non-null terms, however, if any, improve the sparsity of the full Jacobian matrix $\tilde{\boldsymbol{A}}$.
- The additional operations required to compute (24) and (25) are largely compensated by the fact that the computational burden of $\tilde{\boldsymbol{f}}$ and $\tilde{\boldsymbol{g}}$ and their Jacobians is lower than that of $\boldsymbol{f}$ and $\boldsymbol{g}$ and their Jacobians. The latter statement is true only if sparsity is exploited but this is a given of any efficient power system software tool.
- From observing (15) and (19) – or the general expression (24) – one may conclude that the integration scheme of algebraic constraints is different than that of differential equations. In particular, it may seem that algebraic constraints are always integrated using the BEM. This is actually not the case. For algebraic constraints, in fact, $\dot{\boldsymbol{y}} \equiv \boldsymbol{0}$, $\forall t$. Hence, the set of algebraic constraints does not need to be "integrated" as it happens to differential equations, but just "solved" at each step of the time domain simulation. Moreover, note that, in (19) – and in (24) – the step length h multiplies the algebraic constraints for consistency with the term $\boldsymbol{R} \cdot \boldsymbol{\xi}$. The ratio $\boldsymbol{\xi}/h$ is, in fact, the numerical approximation of $\dot{x}$. The accuracy of the term $\boldsymbol{\xi}/h$ does depend on the integration scheme (hence ITM and BDF will provide better estimations than BEM) but, since $\dot{\boldsymbol{y}} \equiv \boldsymbol{0}$, no estimation is needed for algebraic variables and their accuracy is the same regardless the integration scheme.

The latter point also indicates that exploiting the knowledge that $\dot{\boldsymbol{y}} \equiv \boldsymbol{0}$ and, hence, using (4) instead of (7), allows slightly reducing the number of operations if implicit integration schemes of order higher than 1 are used. For example, considering (24), it appears that one does not have to compute the algebraic counterpart of $\tilde{\boldsymbol{\xi}}$ and the term $\kappa\tilde{\boldsymbol{g}}(t)$ is not required to compute $\tilde{\boldsymbol{q}}$.

Sections III and IV support the statements above by means of several analytically examples and a large real-world power

[1]It may appear confusing or, at least, counter-intuitive that the existence of off-diagonal elements of matrices $\boldsymbol{T}$ and $\boldsymbol{R}$ lead $\tilde{\boldsymbol{A}}$ to be sparser than $\boldsymbol{A}$. This is so because $\tilde{\boldsymbol{A}}$ is obtained as the sum of $\boldsymbol{T}$ and $\boldsymbol{R}$ with $\boldsymbol{f}_{\boldsymbol{x}}$ and $\boldsymbol{g}_{\boldsymbol{x}}$, respectively. Each off-diagonal element of $\boldsymbol{T}$ and $\boldsymbol{R}$ often allows removing two or more off-diagonal elements of $\tilde{\boldsymbol{f}}_{\boldsymbol{x}}$ and $\tilde{\boldsymbol{g}}_{\boldsymbol{x}}$. Moreover, the positions occupied by off-diagonal elements of $\boldsymbol{T}$ and $\boldsymbol{R}$ are generally also occupied by some non-zero elements of $\tilde{\boldsymbol{f}}_{\boldsymbol{x}}$ and $\tilde{\boldsymbol{g}}_{\boldsymbol{x}}$. These considerations justify the statements above.

system model.

E. Steady-state Analysis and Quasi-Static Simulation

In steady-state, $\dot{\boldsymbol{x}} = \boldsymbol{0}$ and (4) reduces to:

$$\begin{aligned} \boldsymbol{0} &= \tilde{\boldsymbol{f}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}) \\ \boldsymbol{0} &= \tilde{\boldsymbol{g}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}) \end{aligned} \tag{26}$$

This model is also adequate for quasi-static simulations (QSS). Since $\tilde{\boldsymbol{f}}$ and $\tilde{\boldsymbol{g}}$ require less operations than explicit expressions $\boldsymbol{f}$ and $\boldsymbol{g}$, the QSS analysis based on the semi-implicit formulation results computationally more efficient. The Jacobian matrix for QSS is simplified with respect to (25), as follows:

$$\tilde{\boldsymbol{A}}_{\rm QSS} = \begin{bmatrix} -\beta h \tilde{\boldsymbol{f}}_{\boldsymbol{x}} & -\beta h \tilde{\boldsymbol{f}}_{\boldsymbol{y}} \\ -h \tilde{\boldsymbol{g}}_{\boldsymbol{x}} & -h \tilde{\boldsymbol{g}}_{\boldsymbol{y}} \end{bmatrix} \tag{27}$$

Coefficients β and h can be dropped in this case as $\tilde{\boldsymbol{\xi}} = \boldsymbol{0}, \forall t$.

Empirical observations by the author on several test cases have shown that solving (26) tends to be numerically more stable and/or require less iterations than solving (3) for $\dot{\boldsymbol{x}} = \boldsymbol{0}$.

F. Small-signal Stability Analysis

As it is well-known, the state matrix of the explicit form (3) is:

$$\boldsymbol{A}_s = \boldsymbol{f}_{\boldsymbol{x}} - \boldsymbol{f}_{\boldsymbol{y}} \boldsymbol{g}_{\boldsymbol{y}}^{-1} \boldsymbol{g}_{\boldsymbol{x}} \tag{28}$$

and the n roots of the characteristic equation:

$$\det(\boldsymbol{A}_s - \lambda \boldsymbol{I}_n) = 0 \tag{29}$$

are the n eigenvalues of (28).

To define the state matrix $\boldsymbol{A}_s$ is more involved for the semi-implicit form (4) as more matrix factorizations are required, as follows. Differentiating (4) at an equilibrium point leads to:

$$\begin{aligned} \boldsymbol{T} \Delta \dot{\boldsymbol{x}} &= \tilde{\boldsymbol{f}}_{\boldsymbol{x}} \Delta \boldsymbol{x} + \tilde{\boldsymbol{f}}_{\boldsymbol{y}} \Delta \boldsymbol{y} \\ \boldsymbol{R} \Delta \dot{\boldsymbol{x}} &= \tilde{\boldsymbol{g}}_{\boldsymbol{x}} \Delta \boldsymbol{x} + \tilde{\boldsymbol{g}}_{\boldsymbol{y}} \Delta \boldsymbol{y} \end{aligned} \tag{30}$$

Note that the terms $\boldsymbol{T}_{\boldsymbol{x}} \dot{\boldsymbol{x}}$, $\boldsymbol{T}_{\boldsymbol{y}} \dot{\boldsymbol{x}}$, etc., are null. In fact, at the equilibrium point $\dot{\boldsymbol{x}} = \boldsymbol{0}$. Then one has:

$$\tilde{\boldsymbol{B}}_s \Delta \dot{\boldsymbol{x}} = \tilde{\boldsymbol{A}}_s \Delta \boldsymbol{x} \tag{31}$$

where

$$\begin{aligned} \tilde{\boldsymbol{A}}_s &= \tilde{\boldsymbol{f}}_{\boldsymbol{x}} - \tilde{\boldsymbol{f}}_{\boldsymbol{y}} \tilde{\boldsymbol{g}}_{\boldsymbol{y}}^{-1} \tilde{\boldsymbol{g}}_{\boldsymbol{x}} \\ \tilde{\boldsymbol{B}}_s &= \boldsymbol{T} - \tilde{\boldsymbol{f}}_{\boldsymbol{y}} \tilde{\boldsymbol{g}}_{\boldsymbol{y}}^{-1} \boldsymbol{R} \end{aligned} \tag{32}$$

Let define the pencil $\tilde{\boldsymbol{A}}_s - \lambda \tilde{\boldsymbol{B}}_s$, then the generalized eigenvalue problem of (31) is as follows [29]:

$$\det(\tilde{\boldsymbol{A}}_s - \lambda \tilde{\boldsymbol{B}}_s) = 0 \tag{33}$$

Hence, two factorizations of the Jacobian matrix $\tilde{\boldsymbol{g}}_{\boldsymbol{y}}$ are required for the semi-implicit form, as opposed to one factorization of $\boldsymbol{g}_{\boldsymbol{y}}$ required to compute (28).

If the rank matrix $\boldsymbol{T}$ is $n-k$, i.e., $\boldsymbol{T}$ is not full-rank, k eigenvalues of (33) are infinite. Robust algorithms that solve (33) should return only the $n-k$ finite eigenvalues [30], [31]. Note that, if $\boldsymbol{T}$ is not full-rank, one cannot compute $\boldsymbol{A}_s = \tilde{\boldsymbol{B}}_s^{-1} \tilde{\boldsymbol{A}}_s$ and solve the conventional eigenvalue problem (29) as $\tilde{\boldsymbol{B}}_s$ is not invertible. In fact, if $\boldsymbol{T}$ has a null diagonal element, say $T_{i,i} = 0$, then also the whole i-th columns of $\boldsymbol{T}$ and $\boldsymbol{R}$ are null as $\dot{x}_i = 0$, $\forall t$ must apply. Hence, $\tilde{\boldsymbol{B}}_s$ is singular as its i-th column is null.

The small-signal stability analysis of the semi-implicit form is thus more computationally demanding than that of the explicit one. However, the computation of (33) is only required once per equilibrium point of (4).

A further generalization of the matrix pencil makes unnecessary to factorize $\tilde{\boldsymbol{g}}_{\boldsymbol{y}}$. In fact, one can solve the following augmented general eigenvalue problem:

$$\det(\tilde{\boldsymbol{A}}_c - \lambda \tilde{\boldsymbol{B}}_c) = 0 \tag{34}$$

where:

$$\tilde{\boldsymbol{A}}_c = \begin{bmatrix} \tilde{\boldsymbol{f}}_x & \tilde{\boldsymbol{f}}_y \\ \tilde{\boldsymbol{g}}_x & \tilde{\boldsymbol{g}}_y \end{bmatrix}, \quad \text{and} \quad \tilde{\boldsymbol{B}}_c = \begin{bmatrix} \boldsymbol{T} & \boldsymbol{0} \\ \boldsymbol{R} & \boldsymbol{0} \end{bmatrix}.$$

In (34), the number of infinite eigenvalues is $k+m$ and clearly, the size of the problem is $n \cdot m \times n \cdot m$. The size increase is compensated by the fact that $\tilde{\boldsymbol{A}}_c$ is typically much sparser than $\tilde{\boldsymbol{A}}_s$. Note that $\tilde{\boldsymbol{B}}_c$ is structurally identical to $\boldsymbol{\Xi}(\boldsymbol{z})$ in (8), but is composed of constant elements computed at the equilibrium point.

The generalized eigenvalue problem can be clearly defined also for the explicit formulation (3) and its variant (6). For example, in the case of (6), one has:

$$\det(\boldsymbol{A}_c - \lambda \boldsymbol{B}_c) = 0 \tag{35}$$

where:

$$\boldsymbol{A}_c = \begin{bmatrix} \boldsymbol{f}_x & \boldsymbol{f}_y \\ \boldsymbol{g}_x & \boldsymbol{g}_y \end{bmatrix}, \quad \text{and} \quad \boldsymbol{B}_c = \begin{bmatrix} \boldsymbol{\Gamma} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}.$$

The analysis on the performance and numerical features and issues of (33) and (34) is beyond the scope of this paper and will considered in future work.

III. Examples

As disclosed in the introduction, the semi-implicit formulation reduces the complexity and increases the sparsity of the Jacobian matrices of the right-hand side of the DAE. This statement is best explained through examples. Common power system devices and regulators are considered in this section to illustrate the advantages of (4) over (3).

A. Common Control Blocks

Some basic control blocks are considered in this subsection, namely the lag, the lead-lag and the PI control blocks, whose transfer functions are shown in Fig. 1.

1) Lag Block: The conventional time-domain formulation of the lag function is:

$$\dot{x} = (Ku - x)/T \tag{36}$$

Using the proposed semi-implicit form, one obtains:

$$T\dot{x} = Ku - x \tag{37}$$

In this case, the number of operations of (36) and (37) is the same, but (37) allows using $T = 0$ and is not prone to numerical issues for small values of T.

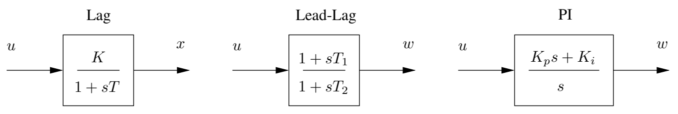

Fig. 1. Transfer functions of common control blocks.

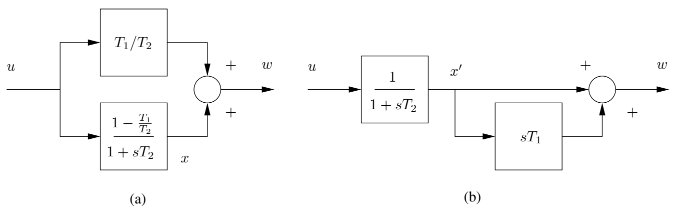

Fig. 2. Implementations of the Lead-lag diagram: (a) parallel model, and (b) series model.

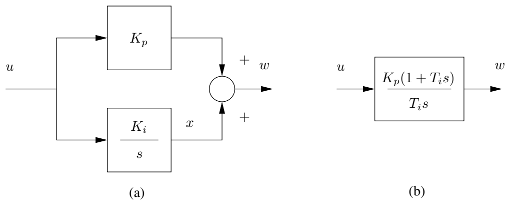

Fig. 3. PI controller: (a) conventional scheme; (b) modified scheme to exploit the semi-implicit formulation.

2) Lead-Lag Block: The non-uniqueness of the time-domain representation of a transfer function is well illustrated through the lead-lag block. Figure 2 shows two conventional representations of a lead-lag transfer function. The "parallel" model is described by:

$$\begin{aligned} \dot{x} &= ((1-T_1/T_2)u-x)/T_2 \\ w &= (T_1/T_2)u+x \end{aligned} \tag{38}$$

whereas the "series" model leads to the following DAE system:

$$\begin{aligned} \dot{x}' &= (u-x')/T_2 \\ w &= (T_1/T_2)(u-x')+x' \end{aligned} \tag{39}$$

The state variable x in (38) takes different values than x' in (39) but the output w is the same for the two models.

The semi-implicit formulation of the lead-lag block is:

$$\begin{aligned} T_2\dot{x} &= u-x \\ -T_1\dot{x} &= x-w \end{aligned} \tag{40}$$

Note that also the semi-implicit formulation is not unique. Another implementation is:

$$\begin{aligned} T_2\dot{x}-T_1\dot{u} &= u-x \\ 0 &= x-w \end{aligned} \tag{41}$$

which requires the same number of operations as (40) but can be defined only if u is a state variable or function of state variables. (40) and (41) show less operations and a sparser Jacobian matrix than (38) and (39). The latter property is best illustrated by the example on the PSS controller given in Subsection III-D.

3) PI Block: PIs are very common blocks found in many regulators of power system devices. The conventional PI implementation is depicted in Fig. 3.a, as follows:

$$\begin{aligned} \dot{x} &= K_iu \\ 0 &= K_pu+x-w \end{aligned} \tag{42}$$

Equations (42) are both explicit and semi-implicit. An alternative semi-implicit formulation, shown in Fig. 3.b, is:

$$\begin{aligned} T_i\dot{x} &= K_pu \\ -T_i\dot{x} &= x-w \end{aligned} \tag{43}$$

where $T_i = K_p/K_i$. Equations (43) lead to a sparser Jacobian matrix than (42), especially if the input u is a function of more than one variable, as it is often the case.

B. Synchronous machine model

A conventional synchronous machine model in explicit form is as follows [32]:

$$\begin{aligned} \dot{e}'_q &= (-e'_q-(x_d-x'_d)(i_d-\gamma_{d2}\psi''_d \\ &\quad -(1-\gamma_{d1})i_d+\gamma_{d2}e'_q)+v_f)/T'_{d0} \\ \dot{e}'_d &= (-e'_d+(x_q-x'_q)(i_q-\gamma_{q2}\psi''_q \\ &\quad -(1-\gamma_{q1})i_q-\gamma_{d2}e'_d))/T'_{q0} \\ \dot{\psi}''_d &= (-\psi''_d+e'_q-(x'_d-x_\ell)i_d)/T''_{d0} \\ \dot{\psi}''_q &= (-\psi''_q-e'_d-(x'_q-x_\ell)i_q)/T''_{q0} \end{aligned} \tag{44}$$

where:

$$\begin{aligned} \gamma_{d1} &= \frac{x''_d-x_\ell}{x'_d-x_\ell}\,, \quad \gamma_{q1}=\frac{x''_q-x_\ell}{x'_q-x_\ell} \\ \gamma_{d2} &= \frac{1-\gamma_{d1}}{x'_d-x_\ell}\,, \quad \gamma_{q2}=\frac{1-\gamma_{q1}}{x'_q-x_\ell} \end{aligned} \tag{45}$$

The proposed semi-implicit model is:

$$\begin{aligned} T'_{d0}\dot{e}'_q+\tilde{T}''_{d0}\dot{\psi}''_d &= -e'_q-(x_d-x'_d)i_d+v_f \\ T'_{q0}\dot{e}'_d-\tilde{T}''_{q0}\dot{\psi}''_q &= -e'_d+(x_q-x'_q)i_q \\ T''_{d0}\dot{\psi}''_d &= -\psi''_d+e'_q-(x'_d-x_\ell)i_d \\ T''_{q0}\dot{\psi}''_q &= -\psi''_q-e'_d-(x'_q-x_\ell)i_q \end{aligned} \tag{46}$$

where

$$\begin{aligned} \tilde{T}''_{d0} &= (x_d-x'_d)\gamma_{d2}T''_{d0} \\ \tilde{T}''_{q0} &= (x_q-x'_q)\gamma_{q2}T''_{q0} \end{aligned} \tag{47}$$

Several models of reduced order can be obtained based on the 6th order model. For example, a commonly-used 5th order model assumes $T'_{q0}=0$ [33]. Note that, while reduced order models require rewriting part of the equations of (44), to reduce the dynamic order of (46), it is sufficient to set to zero a time constant on the left-hand side of (46).

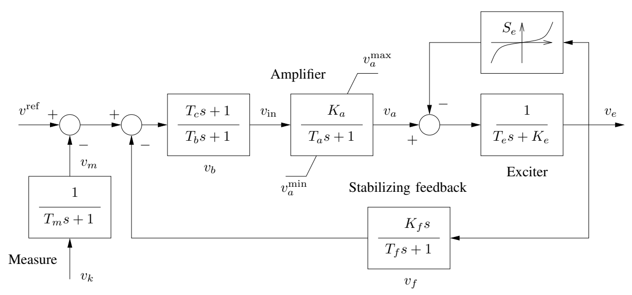

Fig. 4. Control diagram of the IEEE automatic voltage regulator Type DC1 [34].

C. IEEE Type DC1 Exciter and Controller

The control scheme of the IEEE Type DC1 exciter is shown in Fig. 4 [34]. An explicit formulation of this regulator is:

$$\begin{aligned}\dot{v}_m &= (v_k - v_m)/T_m \\ \dot{v}_b &= ((1 - \frac{T_c}{T_b})(v^{\mathrm{ref}} - v_m - v_f - \frac{K_f}{T_f}v_e) - v_b)/T_b \\ \dot{v}_a &= \frac{T_c}{T_b}(K_a(v^{\mathrm{ref}} - v_m - v_f - \frac{K_f}{T_f}v_e) + v_b - v_a)/T_a \\ \dot{v}_f &= -(\frac{K_f}{T_f}v_e + v_a)/T_f \\ \dot{v}_e &= -((K_e + S_e(v_e))v_e - v_a)/T_e\end{aligned} \tag{48}$$

where v_k is the regulated voltage at the generator terminal bus k. Details on the ceiling function S_e can be found in [32]. A possible semi-implicit formulation reads:

$$\begin{aligned}T_m\dot{v}_m &= v_k - v_m \\ T_b\dot{v}_b + K_a\dot{v}_f &= K_a(v^{\mathrm{ref}} - v_m) - v_b \\ T_a\dot{v}_a - T_c\dot{v}_b &= v_b - v_a \\ T_f\dot{v}_f &= K_f v_e - v_f \\ T_e\dot{v}_e &= v_a - (K_e + S_e(v_e))v_e\end{aligned} \tag{49}$$

The anti-windup limiter of the amplifier has to be handled carefully in the semi-implicit formulation. For anti-windup limiters, the sign of the time derivative of the state variable is part of the limiter logic, as follows:

$$\begin{aligned}\text{if } v_a \geq v_a^{\max} \text{ and } \dot{v}_a \geq 0 &\Rightarrow\ v_a = v_a^{\max} \text{ and } \dot{v}_a = 0 \\ \text{if } v_a \leq v_a^{\min} \text{ and } \dot{v}_a \leq 0 &\Rightarrow\ v_a = v_a^{\min} \text{ and } \dot{v}_a = 0 \\ \text{otherwise} &\Rightarrow\ T_a\dot{v}_a = v_{\mathrm{in}} - v_a\end{aligned} \tag{50}$$

where v_{in} is the amplifier input signal (see Fig. 4), whose expression depends on the DAE formulation. Hence, the sign of $\dot{v}_a$, not only the value of v_a, defines the behaviour of the anti-windup limiter. With the explicit formulation (48), the sign of $\dot{v}_a$ is the same as that of its right-hand-side differential equation. With the semi-implicit formulation (49), however, the right-hand-side might not have the same sign of $\dot{v}_a$ due to the term $K_a\dot{v}_f$. Thus, $\mathrm{sign}(\tilde{\boldsymbol{f}}) \neq \mathrm{sign}(\boldsymbol{f}) \equiv \mathrm{sign}(\dot{\boldsymbol{x}})$, in general.

Note that the input signal to the stabilizing feedback of the AVR is a state variable, namely, v_e. Hence, an alternative semi-implicit model can be defined by using the time derivative of the input rather than the output signal of the stabilizing feedback. The resulting model is:

$$\begin{aligned}T_m\dot{v}_m &= v_h - v_m \\ T_b\dot{v}_b &= K_a(v^{\mathrm{ref}} - v_m - v_f) - v_b \\ T_a\dot{v}_a - T_c\dot{v}_b &= v_b - v_a \\ T_f\dot{v}_f - K_f\dot{v}_e &= -v_f \\ T_e\dot{v}_e &= v_a - (K_e + S_e(v_e))v_e\ .\end{aligned} \tag{51}$$

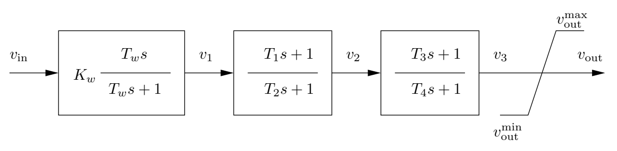

Fig. 5. Control scheme of a common power system stabilizer.

D. Power System Stabilizer

Figure 5 shows a common PSS model consisting of a cascade of washout and lead-lag blocks. An explicit form is:

$$\begin{aligned}\dot{v}_1 &= -(K_w v_{\mathrm{in}} + v_1)/T_w \\ \dot{v}_2 &= ((1 - \frac{T_1}{T_2})(K_w v_{\mathrm{in}} + v_1) - v_2)/T_2 \\ \dot{v}_3 &= ((1 - \frac{T_3}{T_4})(v_2 + (\frac{T_1}{T_2}(K_w v_{\mathrm{in}} + v_1))) - v_3)/T_4 \\ 0 &= v_3 + \frac{T_3}{T_4}(v_2 + \frac{T_1}{T_2}(K_w v_{\mathrm{in}} + v_1)) - v_{\mathrm{out}}\end{aligned} \tag{52}$$

Another set of explicit DAEs is as follows:

$$\begin{aligned}\dot{v}_1 &= (K_w v_{\mathrm{in}} - v_1)/T_w \\ \dot{v}_2 &= (K_w v_{\mathrm{in}} - v_1 - v_2)/T_2 \\ \dot{v}_3 &= (K_1 v_2 + \frac{T_1}{T_2}(K_w v_{\mathrm{in}} - v_1) - v_3)/T_4 \\ 0 &= (K_2 v_3 + \frac{T_3}{T_4}(K_1 v_2 + \frac{T_1}{T_2}(K_w v_{\mathrm{in}} - v_1)) - v_{\mathrm{out}}\end{aligned} \tag{53}$$

where $K_1 = 1 - T_1/T_2$ and $K_2 = 1 - T_3/T_4$. Note that the Jacobian matrices of the two explicit models (52) and (53) are relatively dense.

Also semi-implicit formulations are non-unique. The following is an implementation:

$$\begin{aligned}T_w\dot{v}_1 &= K_w v_{\mathrm{in}} - v_1 \\ T_2\dot{v}_2 - T_w\dot{v}_1 &= -v_2 \\ T_4\dot{v}_3 - T_1\dot{v}_2 &= v_2 - v_3 \\ -T_3\dot{v}_3 &= v_3 - v_{\mathrm{out}}\end{aligned} \tag{54}$$

Even just from a visual inspection, (54) looks simpler than (52) and (53). Note also that the input signal v_{in} *propagates* through (52) and (53), while it appears only in the first equation of (54).

E. Remarks on Explicit and Semi-Implicit Models

For illustration, Table II shows the number of non-zero (NNZ) elements of the Jacobian matrices of different DAE models for the synchronous machines, AVR and PSS devices presented above. The last column of Table II provides the relative reduction of NNZ of semi-explicit formulations versus implicit ones. The reduction is 23.5% in the worst case and 42.9% in the best one. The number of operations of semi-implicit models is sensibly smaller than that of explicit ones.

TABLE II
NUMBER OF NON-ZERO ELEMENTS OF THE JACOBIAN MATRIX OF DIFFERENT DAE MODELS

Device	Model	Eq.	NNZ	Red. %
Synch. Mach.	Ex.	(44)	17	-
	SI	(46)	13	23.5
AVR	Ex.	(48)	17	-
	SI	(49)	12	29.4
	SI	(51)	12	29.4
PSS	Ex.	(52)	14	-
	Ex.	(53)	14	-
	SI	(54)	8	42.9

It is important to note that the proposed semi-implicit approach allows replacing a set of – say – n differential equations with another set of n differential equations, but with sparser structure. However, improving the sparsity of the Jacobian matrix of DAEs can be obtained also by artificially adding algebraic constraints (together with the corresponding algebraic variables) to the original DAE. The latter set of DAEs results of larger size but, if the additional constraints are properly chosen, it can be also much sparser than the original DAE set. The use of "intermediate" constraints and variables is convenient to decompose a large system automatically into its sub-systems. Such consideration have been exploited in the literature. An example is the MANA approach proposed in [22] and, more recently, in [35].

The examples shown in this section are promising but are only particular cases. To apply such transformation to a large software tool, it would be interesting to have a systematic approach to pass from the standard formulation (3) to the proposed (4). Unfortunately, to the best of the knowledge of the author, there is no such a general methodology to transform explicit into (semi-)implicit DAEs. Further investigation on this point will be carried out in future work.

Another aspect that can discourage one from adopting the proposed semi-implicit formulation is the complexity and time associated with porting an existing software tool based on on (3) to (4). With this regard, it can be useful to note that the changes required to adapt the implicit integration schemes to the semi-implicit formulation are relatively small and can be implemented quickly. For example, the author was able to implement such changes in two weeks in his software tool Dome [27]. Of course, more time consuming is to rewrite device models. However, since the conventional implicit formulation is a special case of the semi-implicit one, conventional explicit models can coexist with the semi-implicit ones (this features has been exploited for the simulations included in the case study discussed in Section IV). Hence porting a software tool from the explicit to the semi-explicit formulation is relatively smooth operation that allows maintaining compatibility with existing explicit models.

IV. CASE STUDY

In this case study, the properties and the performance of the conventional and the proposed DAE models are compared through a dynamic model of the ENTSO-E transmission system.[2] The model includes 21,177 buses (1,212 off-line); 30,968 transmission lines and transformers (2,352 off-line); 1,144 coupling devices, i.e., zero-impedance connections (420 off-line); 15,756 loads (364 off-line); and 4,828 power plants. Of these power plants, 2,664 are modelled with 6th order synchronous machines models with AVRs and turbine governors. The remaining 2,168 machines are simulated with the classical 2nd order model. 1,160 power plants are off-line. The system also include 364 PSSs. The topology and the data of the transmission system are based on the actual real-world system provided by the ENTSO-E but dynamic data are guessed and based on the technology of power plants.

All simulations are obtained using Dome, a Python-based power system analysis toolbox [27]. The Dome version used for in this case study is based on Python 3.4.1; ATLAS 3.10.1 for dense vector and matrix operations; CVXOPT 1.1.7 for sparse matrix operations; and KLU 1.3.2 for sparse matrix factorization. All simulations were executed on a 64-bit Linux Fedora 21 operating system running on two Intel Xeon 4 Core 3.5 GHz CPUs, and 12 GB of RAM.

Table III shows the statistics for the ENTSO-E network modelled with the explicit formulation (3). This consists of 39,732 state variables and 106,432 algebraic variables. In particular, Table III shows the sparsity degree of Jacobian matrices of (3) including the full Jacobian matrix $\boldsymbol{A}$ as defined in (23).

TABLE III
STATISTICS FOR THE EXPLICIT MODEL OF THE ENTSO-E TRANSMISSION SYSTEM

Matrix	Rows	Cols	NNZ	NNZ %
$\boldsymbol{g_y}$	$106,432$	$106,432$	$487,599$	0.00430%
$\boldsymbol{g_x}$	$106,432$	$39,732$	$33,272$	0.00079%
$\boldsymbol{f_y}$	$39,732$	$106,432$	$54,828$	0.00130%
$\boldsymbol{f_x}$	$39,732$	$39,732$	$80,164$	0.00508%
$\boldsymbol{A}$	$146,164$	$146,164$	$655,863$	0.00307%

Table IV shows the statistics for the ENTSO-E network modelled with the semi-implicit formulation (4). In this case, there are 49,396 differential equations and 96,768 algebraic ones. The rank of the matrix $\boldsymbol{T}$ is 39,732, which means that 9,664 state variables have a null time constant. These are mainly stator fluxes of synchronous machines of power plants. Note that the total number of variables, i.e., $n + m$, is the same as for the explicit formulation of Table III. The last column of Table IV indicates the reduction, in percentage, of the number of non-zero elements passing from the explicit to the semi-implicit formulation.[3] As expected, the semi-implicit form allows a noticeable reduction of the NNZ of

[2]The data of the system has been licensed to the author by ENTSO-E. Data can be requested through an on-line application at `www.entsoe.eu`.

[3]This reduction is based on the NNZ of Jacobian matrices. Note that this comparison is somewhat "loose" as the number of state and algebraic variables of the explicit and semi-implicit models is not the same. The only exact comparison applies to full Jacobian matrices $\boldsymbol{A}$, $\tilde{\boldsymbol{A}}$ and $\tilde{\boldsymbol{A}}_{\rm QSS}$.

matrices $\tilde{\boldsymbol{f}}_{\boldsymbol{x}}$, $\tilde{\boldsymbol{f}}_{\boldsymbol{y}}$ and $\boldsymbol{g}_{\boldsymbol{x}}$. The algebraic Jacobian matrix $\tilde{\boldsymbol{g}}_{\boldsymbol{y}}$ is also slightly sparser than the explicit matrix $\boldsymbol{g}_{\boldsymbol{y}}$ because of the terms depending on $\dot{\boldsymbol{x}}$. These terms are included in $\boldsymbol{R}$, which, however, does not increase the sparsity of $\tilde{\boldsymbol{g}}_{\boldsymbol{x}}$. The full Jacobian matrix $\tilde{\boldsymbol{A}}$ is about 14% sparser than $\boldsymbol{A}$ and the full Jacobian matrix for QSS analysis, $\tilde{\boldsymbol{A}}_{QSS}$, is about 16% sparser than $\boldsymbol{A}$.

TABLE IV
STATISTICS FOR THE SEMI-IMPLICIT MODEL OF THE ENTSO-E TRANSMISSION SYSTEM

Matrix	Rows	Cols	NNZ	NNZ %	Red. %
$\tilde{\boldsymbol{g}}_{\boldsymbol{y}}$	96,768	96,768	395,362	0.00422%	1.87%
$\tilde{\boldsymbol{g}}_{\boldsymbol{x}}$	96,768	49,396	34,768	0.00073%	7.59%
$\boldsymbol{R}$	96,768	49,396	844	0.00002%	-
$\boldsymbol{R}+\tilde{\boldsymbol{g}}_{\boldsymbol{x}}$	96,768	49,396	34,768	0.00073%	7.59%
$\tilde{\boldsymbol{f}}_{\boldsymbol{y}}$	49,396	96,768	46,192	0.00097%	25.38%
$\tilde{\boldsymbol{f}}_{\boldsymbol{x}}$	49,396	49,396	77,356	0.00317%	37.60%
$\boldsymbol{T}$	49,396	49,396	50,948	0.00209%	-
$\boldsymbol{T}+\tilde{\boldsymbol{f}}_{\boldsymbol{x}}$	49,396	49,396	89,664	0.00367%	27.76%
$\tilde{\boldsymbol{A}}$	146,164	146,164	565,986	0.00265%	13.68%
$\tilde{\boldsymbol{A}}_{\rm QSS}$	146,164	146,164	553,678	0.00259%	15.64%

A time domain simulation is carried out to compare the performance of the explicit and semi-implicit formulations. The case study consists in simulating a three-phase fault at bus 129,218 occurring at $t = 1$ s and cleared after 200 ms. The total simulated time is 5 s. Table V shows the CPU times to solve the explicit and the semi-implicit DAE formulations using the full dynamic model and the integration schemes discussed in subsection II-C, namely, the BEM, ITM and a second order BDF method. The latter is restarted by means of the BEM after the short-circuit occurrence and clearance. A standard dishonest Newton-Raphson method to solve each point of the time domain simulation is used [11]. For comparison, the integration step length h and the convergence tolerance ϵ are constant and equal for all integration methods. In particular, $h = 0.02$ s and $\epsilon = 10^{-6}$ are used. Note that the values shown in Table V refers to the time required to complete the sole time integration routine, not the power flow analysis and the initialization of dynamic models. As expected, the semi-implicit formulation allows reducing the computational burden. The overall gain is about 15%.

CPU times obtained using the QSS model outlined in Subsection II-E are also shown in Table V. Times refer to a simulation of 60 s with a time step $h = 0.1$ s. Loads increase following a ramp rate of 10^{-5} pu/s. Also in this case, the semi-implicit formulation performs better than the conventional explicit one.

TABLE V
COMPUTATIONAL BURDEN OF TIME DOMAIN INTEGRATION FOR THE ENTSO-E TRANSMISSION SYSTEM

Method	Explicit	Semi-Implicit	Time Saving
BEM	24.67 s	21.15 s	14.26% s
ITM	38.38 s	32.72 s	16.26% s
BDF	39.04 s	32.80 s	15.98% s
QSS	27.52 s	23.76 s	13.66% s

A. Remarks on Simulation Results

From the results shown in Table V, it appears that, while more efficient than the standard explicit DAEs, the semi-implicit formulation does not lead to an outstanding improvement of simulation times. With this regard, the following remarks are relevant.

- The proposed semi-implicit DAE formulation can be coupled with any other technique to speed up time domain simulation software, e.g., parallelization and extra algebraic constraints (see also the discussion in Subsection III-E). Hence, even if the speed up provided by the semi-implicit formulation *per se* is not huge, such a speed up can be combined with and, possibly, amplified by other numerical techniques.
- Reducing the computational burden is not the sole benefit of the proposed semi-implicit formulation. As discussed in the paper, other relevant features are the intrinsic ability to accept zero time constants, thus leading to the possibility to seamlessly switch state variables to algebraic ones; and a consistent reduction of the sparsity of the Jacobian matrix. The latter is a feature that is expected to be beneficial to further speedup the time domain analysis if other techniques are used, as discussed in the point above.
- The case study considered in the paper as well as the many tests that the author has carried out to test the semi-implicit formulation, indicate that the behaviour, including convergence, i.e., number of iterations to complete each integration step, of the considered integration methods are effectively identical for both the explicit and semi-implicit formulations.

V. CONCLUSIONS AND FUTURE WORK

This paper proposes a semi-implicit formulation of DAEs for angle and voltage stability analysis of power systems. This formulation provides several advantages with respect to the conventional explicit one. The advantages are: (i) differential equations are simplified and require less operations; (ii) Jacobian matrices are sparser and, hence, require less time to factorize; and (iii) state variables can effortlessly switched to algebraic ones by simply imposing null time constants. The case study based on a 21,177-bus model the ENTSO-E transmission system confirms that the proposed formulation allows reducing the computational burden of time domain integration.

The author is currently investigating several applications of the proposed semi-implicit formulation of DAEs. For example, an aspect that will be considered is how (4) affects the computation of the state matrix of the DAE and, hence, its impact on small signal stability analysis. Future work will also focus on the definition, if possible, of a systematic methodology to transform explicit DAEs into semi-implicit ones. This will also involve the attempt to define in a mathematical way the *optimal* structure of DAEs in terms of number of operations and Jacobian matrix sparsity. Coupling the semi-implicit approach with techniques that increase the sparsity of Jacobian matrices through additional algebraic constraints as well as with techniques that exploits different approximations of the DAEs for time scales will be investigated. Future work will focus also on further exploring the potential of the semi-implicit DAE formulation model for other dynamic systems, such as electro-magnetic models of power systems and electronic circuits.

It appears that the possible developments of the proposed approach are several and diverse. Reviewers' comments on this work have been very stimulating and suggested most of the future work indicated above. It is the hope of the author that this work can pave the way to a new paradigm of power system modelling and time domain analysis.

References

[1] H. W. Dommel and N. Sato, "Fast Transient Stability Solutions," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-91, no. 4, pp. 1643–1650, 1972.

[2] B. Stott, "Power System Dynamic Response Calculations," *Proceedings of the IEEE*, vol. 67, no. 2, pp. 219–241, Feb. 1979.

[3] F. L. Alvarado, R. H. Lasseter, and J. J. Sanchez, "Testing of Trapezoidal Integration with Damping for the Solution of Power Transient Problems," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-102, no. 12, pp. 3783–3790, Dec. 1983.

[4] F. De Mello, J. Feltes, and T. Laskowski, "Simulating Fast and Slow Dynamic Effects in Power Systems," *IEEE Transactions on Power Apparatus and Systems*, vol. 5, no. 3, pp. 33–38, 1992.

[5] C. A. Cañizares, "Voltage Stability Assessment: Concepts, Practices and Tools," IEEE/PES Power System Stability Subcommittee, Final Document, Tech. Rep., Aug. 2002, available at http://www.power.uwaterloo.ca.

[6] IEEE/CIGRE Joint Task Force on stability Terms and Definitions, "Definition and Classification of Power System Stability," *IEEE Transactions on Power Systems*, vol. 19, no. 2, pp. 1387–1401, May 2004.

[7] I. A. Hiskens, "Power System Modeling for Inverse Problems," *IEEE Transactions on Circuits and Systems - I: Regular Papers*, vol. 51, no. 3, pp. 539–551, Mar. 2004.

[8] G. Gross and A. Bergen, "An Efficient Algorithm for Simulation on Transients in Large Power Systems," *IEEE Transactions on Circuits and Systems*, vol. 23, no. 12, pp. 791–799, Dec. 1976.

[9] D. Yang and V. Ajjarapu, "A Decoupled Time-domain Simulation Method via Invariant Subspace Partition for Power System Analysis," *IEEE Transactions on Power Systems*, vol. 21, no. 1, pp. 11–18, Feb. 2006.

[10] J. S. Chai, N. Zhu, and A. Bose, "Parallel Newton Type Methods for Power System Stability Analysis using Local and Shared Memory Multiprocessors," *IEEE Transactions on Power Systems*, vol. 6, no. 4, pp. 1539–1545, Nov. 1991.

[11] M. La Scala, G. Sblendorio, A. Bose, and J. Q. Wu, "Comparison of Algorithms for Transient Stability Simulations on Shared and Distributed Memory Multiprocessors," *IEEE Transactions on Power Systems*, vol. 11, no. 4, pp. 2045–2050, Nov. 1996.

[12] P. Aristidou, D. Fabozzi, and T. Van Cutsem, "Dynamic Simulation of Large-scale Power Systems Using a Parallel Schur-complement-based Decomposition Method," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 10, pp. 2561–2570, Oct. 2014.

[13] P. Aristidou and T. Van Cutsem, "Algorithmic and Computational Advances for Fast Power System Dynamic Simulations," in *IEEE PES General Meeting*, July 2014, pp. 1–5.

[14] J. Astic, A. Bihain, and M. Jerosolimski, "The Mixed Adams-BDF Variable Step Size Algorithm to Simulate Transient and Long Term Phenomena in Power Systems," *IEEE Transactions on Power Systems*, vol. 9, no. 2, pp. 929–935, May 1994.

[15] J. Sanchez-Gasca, R. D'Aquila, and W. Price, "Variable Time Step, Implicit Integration for Extended-term Power System Dynamic Simulation," in *Proc. IEEE Power Industry Computer Application Conf. 1995*, May 1995, pp. 183–189.

[16] R. K. Brayton, F. G. Gustavson, and G. D. Hachtel, "A New Efficient Algorithm for solving Differential-Algebraic Systems using Implicit Backward Differentitation Formulas," *Proceedings of the IEEE*, vol. 60, no. 1, pp. 98–108, Jan. 1972.

[17] C. Fu, J. D. McCalley, and J. Tong, "A Numerical Solver Design for Extended-Term Time-Domain Simulation," *IEEE Transactions on Power Systems*, vol. 28, no. 4, pp. 4926–4935, Nov 2013.

[18] K. Strunz and E. Carlson, "Nested Fast and Simultaneous Solution for Time-Domain Simulation of Integrative Power-Electric and Electronic Systems," *IEEE Transactions on Power Delivery*, vol. 22, no. 1, pp. 277–287, Jan. 2007.

[19] V. Jalili-Marandi, V. Dinavahi, K. Strunz, J. A. Martinez, and A. Ramirez, "Interfacing Techniques for Transient Stability and Electromagnetic Transient Programs," *IEEE Transactions on Power Delivery*, vol. 24, no. 4, pp. 2385–2395, Oct. 2009.

[20] T. Van Cutsem and C. D. Vournas, "Voltage Stability Analysis in Transient and Mid-term Time Scales," *IEEE Transactions on Power Systems*, vol. 11, no. 1, pp. 146–154, Jan. 1996.

[21] D. Fabozzi and T. Van Cutsem, "Simplified Time-domain Simulation of Detailed Long-term Dynamic Models," in *IEEE PES General Meeting*, July 2009, pp. 1–8.

[22] J. Mahseredjian, S. Dennetière, L. Dubé, B. Khodabakhchian, and L. Gérin-Lajoie, "On a new Approach for the Simulation of Transients in Power Systems," *Electric Power Systems Research*, vol. 77, no. 11, pp. 1514 – 1520, 2007, selected Topics in Power System Transients - Part {II6th} International Conference on Power System Transients.

[23] T. Van Cutsem, "Voltage Instability: Phenomena, Countermeasures, and Analysis Methods," *Proceedings of the IEEE*, vol. 88, no. 2, pp. 208–227, Feb. 1992.

[24] Q. Wang, H. Song, and V. Ajjarapu, "Continuation-Based Quasi-Steady-State Analysis," *IEEE Transactions on Power Systems*, vol. 21, no. 1, pp. 171–179, Feb. 2006.

[25] K. E. Brenan, S. L. Campbell, and L. R. Petzold, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*. New York, NY: North-Holland, 1989.

[26] C. W. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations*. Upper Saddle River, NJ: Prentice Hall, 1971.

[27] F. Milano, "A Python-based Software Tool for Power System Analysis," in *Procs. of the IEEE PES General Meeting*, Vancouver, BC, Jul. 2013.

[28] F. E. Cellier and E. Kofman, *Continuous System Simulation*. London, UK: Springer, 2006.

[29] J. W. Demmel and B. Kågström, "Computing stable eigendecompositions of matrix pencils," *Linear Algebra Applications*, vol. 88/89, no. 10, pp. 139–186, Oct. 1987.

[30] ——, " The Generalized Schur Decomposition of an Arbitrary Pencil $A - \lambda B$: Robust Software with Error Bounds and Applications, Part I: Theory and Algorithms," *ACM Transactions on Mathematical Software*, vol. 19, pp. 160–174, 1993.

[31] ——, " The Generalized Schur Decomposition of an Arbitrary Pencil $A - \lambda B$: Robust Software with Error Bounds and Applications, Part II: Software and Applications," *ACM Transactions on Mathematical Software*, vol. 19, pp. 175–201, 1993.

[32] P. W. Sauer and M. A. Pai, *Power System Dynamics and Stability*. Upper Saddle River, New Jersey: Prentice Hall, 1998.

[33] F. Milano, *Power System Modelling and Scripting*. London: Springer, 2010.

[34] IEEE Working Group on Computer Modelling of Excitation Systems, "Excitation System Models for Power System Stability Studies," *IEEE Transactions on Power Apparatus and Systems*, vol. 100, no. 2, pp. 494–509, Feb. 1981.

[35] I. Kocar, J. Mahseredjian, U. Karagaac, G. Soykan, and O. Saad, "Multiphase Load-Flow Solution for Large-Scale Distribution Systems Using MANA," *IEEE Transactions on Power Delivery*, vol. 29, no. 2, pp. 908–915, Apr. 2014.

Federico Milano (S'02, M'04, SM'09, F'16) received from the Univ. of Genoa, Italy, the ME and Ph.D. in Electrical Eng. in 1999 and 2003, respectively. From 2001 to 2002 he was with the Univ. of Waterloo, Canada, as a Visiting Scholar. From 2003 to 2013, he was with the Univ. of Castilla-La Mancha, Spain. In 2013, he joined the Univ. College Dublin, Ireland, where he is currently an associate professor. His research interests include power system modelling, stability analysis and control.