Highlights

Fast and Accurate Simulation of Smart Digital Controllers in Power System Dynamic Studies

Mehran Jafari, Gautier Bureau, Marco Chiaramello, Adrien Guironnet, Patrick Panciatici, Petros Aristidou

- A new, fast, interpolation-based method (IBM) for incorporating and analyzing multiple smart digital controller models in power system dynamic simulations while maintaining accuracy.
- A prototype solver with the proposed IBM and showcasing its accuracy and performance on three example systems with equation-based and smart, black-box, digital controllers.

Fast and Accurate Simulation of Smart Digital Controllers in Power System Dynamic Studies

Mehran Jafari[a,*], Gautier Bureau[b], Marco Chiaramello[b], Adrien Guironnet[b], Patrick Panciatici[b], Petros Aristidou[a]

[a] *Cyprus University of Technology, Limassol, Cyprus*
[b] *RTE, Paris, France*

Abstract

In recent decades, the number of smart digital controllers employed in electric power systems has increased drastically. Either from the modernization of existing analog ones or the introduction of new, data-driven, or even optimization-based controllers, they are now dominating the behavior of power systems. However, this introduces a challenge in the simulation of power system dynamics, as the existing numerical simulation methods are very time-consuming when tackling the resulting hybrid differential-algebraic systems. In this paper, a novel interpolation-based method is proposed for performing fast and accurate dynamic simulations of electric power systems equipped with smart digital controllers. This method fully exploits the potential of variable time-step integration methods without requiring a time-step reduction in the case of discrete events stemming from digital controllers. Therefore, it accelerates the numerical simulation of large-scale systems containing many non-equation-based smart digital controllers, while maintaining

*Corresponding author
Email address: mm.jafari@edu.cut.ac.cy (Mehran Jafari)

accuracy. The performance of the proposed method is showcased using both conventional and smart digital controllers.
Keywords: smart digital controllers, power system dynamic simulations, power system modeling, hybrid differential-algebraic equations, discontinuities.

---

1. Introduction

1.1. Background and Motivation

Electric power systems are continuously expanding, with new interconnections and components installed daily. Moreover, the introduction of Smart Grids and the widespread use of microgrids is accompanied by the introduction of smart digital controllers, incorporated to handle the optimal operation of local units, implement p2p communication and negotiations, optimize the wide-scale operation of the systems, etc. The majority of existing conventional controllers (e.g., PI-based) and all the modern controllers (e.g., Machine Learning-based, fuzzy logic, MPC-based, etc.) are digital and discrete in nature, thus leading to a mixture of continuous and discrete behavior of power systems. The dynamic response of such systems is usually characterized with the use of *large-scale, hybrid, stiff, differential-algebraic equations (DAEs)* [1].

The proliferation of smart digital controllers in power systems introduces significant challenges to their modeling and simulation. More specifically, the digital control actions, defined by their sampling and action time intervals, insert thousands of discrete changes [2] that can alter the DAEs describing the system and introduce discontinuities. Figure 1 shows the discrete time events

introduced in a small system equipped with three digital controllers, each with a different sampling rate. Each vertical bar in the figure represents the sampling of at least one controller and thus a discrete event. In this example, three digital controllers with different sampling times equal to 0.1 s, 0.12 s, and 0.15 s are considered.

When performing a dynamic study of the system, simulating the DAEs with a variable step numerical method, these discrete events define the time instances when the simulation needs to stop and restart. Therefore, it gives an indication of the computational burden imposed by these digital controllers due to the required time-step reduction and system initialization. In large-scale systems, addressing the multiple discontinuities introduced by the digital controllers remains an open challenge and requires special and time-consuming treatment.

Overall, discrete events in DAEs can be categorized as *state events* and *time events* [2]. The events happening at specific points in time, that are known in advance, are called "*time events*" while the events triggered by rules applied to the state variable values of the system are called "*state events*". The operation of smart digital controllers with specific sampling and action times is a classical example of time events. On the other hand, the operation of transformer tap changers due to a change in the system's voltage is considered a state event.

However, in the case of digital controllers used, many of the state events are transformed into time events. For instance, a digital AVR controller with overexcitation limit (OXL): while the OXL will trigger a state event when the field current is violated, the actual detection of the limit crossing and activa-

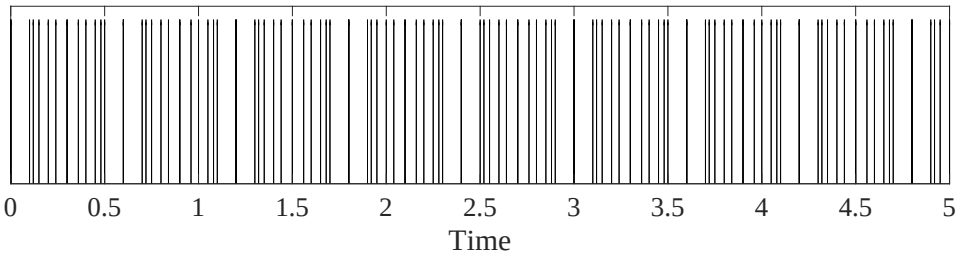

Figure 1: Discrete events in a system with three digital controllers with sampling time 0.1 s, 0.12 s, and 0.15 s, over a 5 s simulation.

tion of the control will take place at the next time sampling period depending on the digital controller frequency. Thus, this paper is concerned only with the treatment of "*time events*" stemming from digital control actions.

One of the simplest and most used methods of treating time events [2] relies on the preparation of an exhaustive time-event list at the beginning of the simulation. Then, the simulation proceeds to the next event, reduces the time-step to land on the exact event time, changes the equation set of the system, computes new initial values, and restarts the integration with the new DAEs and initial values [3]. While this method is the most accurate, it can become computationally intensive and handicaps variable time-step methods since the step size is limited to the intervals between time events (see Fig. 1).

A second approach shifts all the discontinuities taking place within a time-step to the end of the time-step [4]. This approach is fast and computationally inexpensive. However, it leads to simplified simulations, the artificial synchronization of events, and introduces inaccuracies. Moreover, the simulation can be stuck between two or more alternating states (also

called limit-cycle) due to the aggregation of events at the end of a time step. Ref. [4] introduces precautions to reduce such simulation failures.

Finally, another treatment employs Filippov theory [5] to make a continuous bridge between the vector fields on two sides of the discontinuity surface [6]. This method can especially be helpful for systems with chattering problems [7]. A solution based on Filippov theory is proposed for the chattering problem of PI controllers in [8, 9]. Although it can provide an effective way to treat a discontinuity, it still has a high computational burden, especially when treating hundreds or thousands of events, and requires extensive modifications to the existing DAE solvers.

Therefore, as more and more smart digital controllers are introduced in smart grids, there is an urgent need for a method that is accurate but at the same time has high computational performance. The existing methods are either accurate but computationally heavy or fast but with the cost of accuracy loss [10].

1.2. Highlights and Contributions

In this paper, we propose a novel method for handling discrete time events in a system containing multiple smart digital controllers. The method is based on an interpolating treatment of the discrete events, within each time step, without stopping the simulation method or reducing the time step. Thus, variable time-step methods are not hindered and the computational burden is significantly reduced.The proposed method is capable of incorporating phasor-domain dynamic studies of both equation-based and black-box smart digital controllers. This means that the digital controllers' sampling actions (time events) are not a burden for the simulation, and they can be

accurately yet quickly simulated without the need for any simplification, relaxation, or replacement by their analog counterparts.

The proposed method uses interpolation to estimate the state variables of the system (digital controller inputs). A similar approach is employed in modern EMT simulators to determine the on or off state of the switching devices [11, 12], the proposed method calculates the digital controller outputs and refines them in each Newton iteration of the non-linear solver by including them in the state variables vector of the system. This means that the digital controller outputs also affect the convergence test of the solver and the time step selection through the error estimate scheme.

The proposed method is developed considering a variable time step approach which means that the time step selection is affected by the error estimate calculations similar to the simulation of a system under control by analog controllers. In other words, the time steps are not limited nor reduced by the time events of the digital controller. This also means that the proposed method is able to solve the systems under control by non-equation-based smart digital controllers using a variable time step scheme.

The main contributions of the paper can be highlighted as follows:

- A new, fast, interpolation-based method (IBM) for incorporating and analyzing multiple smart digital controller models in power system dynamic simulations while maintaining accuracy.

- A prototype solver with the proposed IBM and showcasing its accuracy and performance on three example systems with equation-based and smart, black-box, digital controllers.

The rest of the paper is organized as follows. First, the modeling back-

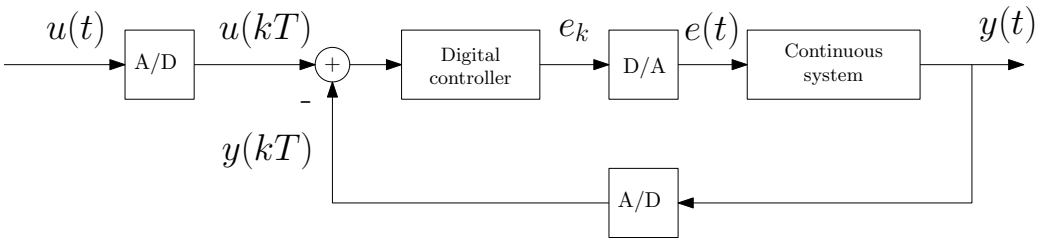

Figure 2: General schematic of a continuous system controlled by a digital controller that controls the error $u(t) - y(t)$

ground and solution methods are introduced in Section 2. Then, the proposed methodology is presented in Section 3. Section 4 introduces the test case systems, and a comparison between the proposed approach and other existing approaches is provided. Finally, the paper concludes in Section 5 with practical comments and future steps.

2. Smart Digital Controllers in Dynamic Studies

Let's consider a continuous system controlled by a digital controller, as shown in Fig. 2. For simplicity, the *continuous system* is described by an Initial Value Problem (IVP) of Ordinary Differential Equations (ODEs).

$$\begin{aligned} \dot{\boldsymbol{y}}(t) &= \boldsymbol{f}(\boldsymbol{y}(t), e(t)) \\ \boldsymbol{y}(0) &= \boldsymbol{y_0},\ e(0) = e_0 \end{aligned} \tag{1}$$

where $\boldsymbol{y}$ is the vector of differential states with $\boldsymbol{y}_0$ the initial value vector and $e(t)$ is the digital controller output at time t with e_0 its initial value.

As seen in Fig. 2, the digital controller takes as input the variables $\boldsymbol{y}$ at the time $t = kT$, where $k \in \mathbb{N}$ and T is the controller sample time. Then, the controller uses the historical controller actions and the control inputs to

compute the next digital control output e_k. If we assume a single historical value used in the controller algorithm, then the digital controller model for sampling time kT, is defined as follows:

$$e_k = \zeta(e_{k-1}, \boldsymbol{y}(kT)) \tag{2}$$

where ζ is the control function. While ζ can be a conventional equation-based function, in many recent applications it could also be a *black box*. That is either because the internal control function is unavailable for proprietary reasons or it cannot be formulated using DAEs (e.g., optimization-based controllers, AI-based controllers, etc.).

To interface with the continuous system, the digital signal e_k is then converted into a piecewise analog signal $e(t)$. For instance, a zero-order-hold approach can be used as shown below:

$$e(t) = e_k, \quad t \in [kT, (k+1)T[\tag{3}$$

In most applications, the signals e_k (controller's discrete output) and $y(kT)$ (discrete sample of output) are quantized. A q-bit uniform quantization method is used to quantize the digital signals. The quantization is performed after each Newton iteration in the corrector step. Let U_e and U_y be the measures of the intervals in which e and y take values, a q-bit uniform quantization can be performed as [13]:

$$\begin{aligned} e_{q,k} &= \text{round}\left(\frac{e_k 2^q}{U_e}\right) \cdot \frac{U_e}{2^q} \\ \boldsymbol{y}_q(kT) &= \text{round}\left(\frac{\boldsymbol{y}(kT) 2^q}{U_y}\right) \cdot \frac{U_y}{2^q} \end{aligned} \tag{4}$$

2.1. Integration Scheme

To perform a dynamic simulation of the system with the smart digital controllers, an appropriate integration scheme must be used. An example of such a scheme is presented below in three steps:

2.1.1. Integration Method

In this work, a variable time-step predictor-corrector integration method is used, where $\boldsymbol{y}_n$ is the solution of $\boldsymbol{y}(t_n)$ at time t_n ($n \in \mathbb{N}$) and $h_n = t_n - t_{n-1}$ is the selected time-step size. The predictor provides an initial estimate of the solution $\boldsymbol{y}_n^{(0)}$ using, in most cases, an explicit integration method.

Then, the corrector calculates the exact solution of $\boldsymbol{y}_n$ using an implicit integration method. This is usually obtained by solving an implicit set of discretized equations using an iterative Newton method.

In this work, the pair of second-order Adams-Bashforth and Adams-Moulton [14] methods are used in the predictor-corrector approach.

2.1.2. Convergence Test

In the corrector phase, after each Newton iteration, a convergence test is applied on $\boldsymbol{y}_n$ according to the user-specified threshold. The Newton iterations continue until the condition is satisfied. A maximum number of iterations is usually used to safeguard against non-converging situations.

2.1.3. Time-step Selection

For each time step solution, an error test is performed based on Milne's estimate approach to adjust the size of the next time step [14]. This approach states that in the case of predictor-corrector methods, the error estimate can

be calculated using the difference between the predictor and corrector multiplied by a coefficient that depends on the chosen pair of methods. For the pair of second-order Adams methods chosen in this work, it can be computed using:

$$\boldsymbol{d}_n = \left| \frac{\boldsymbol{y}_n - \boldsymbol{y}_n^0}{3(\frac{h_{n-1}}{h_n})} \right|_\infty \leq \text{Tolerance} \tag{5}$$

Based on the error estimated by (5) for each time step, three situations might occur:

1. If the estimated error is less than the Tolerance set by the user, the next time step sizes will increase until they reach the Tolerance.
2. If the estimated error is more than the Tolerance, the current time step will be rejected and the integration restarts for this time step with a smaller size h_n.
3. If the estimated error is more than the Tolerance and the time step is already at the minimum value, the current time step is accepted, but a warning is issued to the user.

2.2. Incorporating Smart Digital Controllers

When incorporating the Smart Digital Controllers in the dynamic simulation, several discrete events are introduced in the process described above. Figure 3 shows the "normal" time steps (h_n) and their solutions (y_{n-1} and y_n) based on the variable time-step method described above, and the smaller smart digital controller time sampling times and their solutions and outputs ($y_{n,i}$ and $e_{n,i}$). The black horizontal piece-wise lines show the ZOH controller signal $e(t)$, and the approximate solutions are shown with blue and red crosses for the main time steps and intermediate time steps, respectively.

Let p_n be the number of digital controller actions happening within the time step $h_n = t_n - t_{n-1}$. We denote $e_{n,g}$ with $g \in [1, p_n]$ the value of e at time $t_{n,g}$, where $t_{n-1} < t_{n,g} < t_n$. It should be noted that there is no reason that sample times kT should coincide with the integration time steps t_n as shown in Fig. 3.

As described in the introduction, solving this combined system in an accurate and fast way is extremely challenging. The traditional method would be to reduce the time-step h_n to coincide with the controller actions, and significantly limit the variable time-step method. However, a new method is proposed in the next section to address the computational and accuracy problems.

2.3. State Events

As mentioned before, a state event happens due to the conditions of the system. Changing a tap of a transformer due to a voltage violation or reaching a variable to its maximum value are examples of state events. In a system under control by continuous analog controllers, such events must be detected in the time steps and located accurately in a time step. Then, the time step must be reduced to land exactly on the event to ensure accuracy. This is the accurate approach for a continuous controller since it addresses the event immediately without delay.

However, in the case of a system with digital controllers, due to the discrete nature of the controllers, the state event happening in the system will not be addressed by the controller until the next sampling action. In other words, the state event is shifted to the next time event of the digital controller.

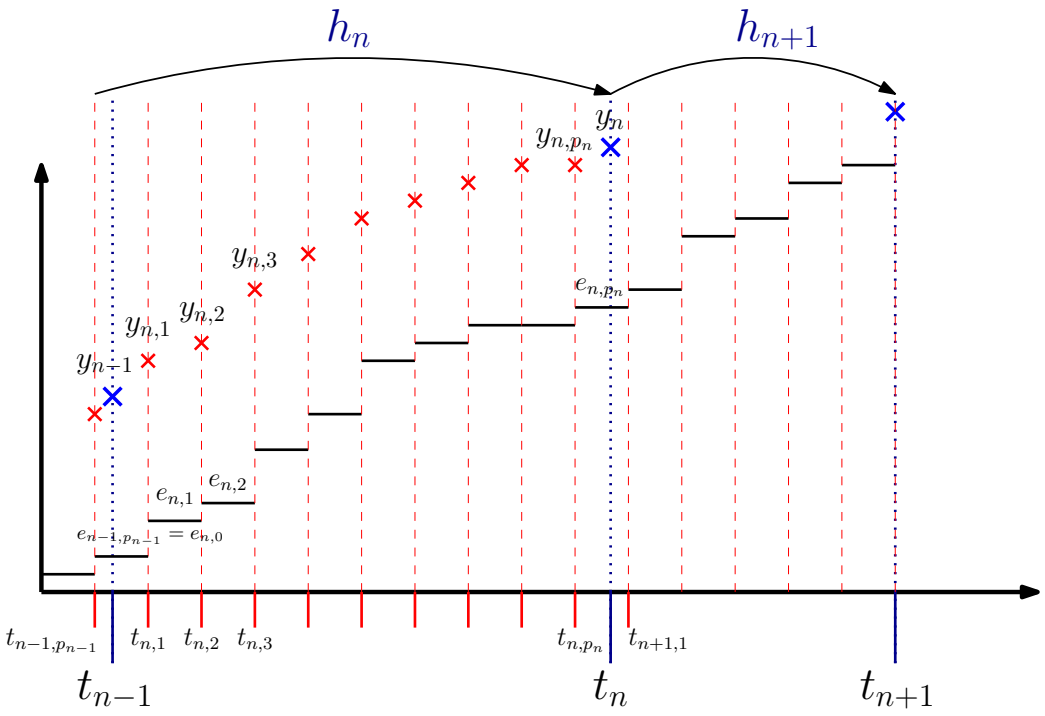

Figure 3: Example of the integration time steps and the intermediate control actions in a variable time-step simulation with smart digital controllers.

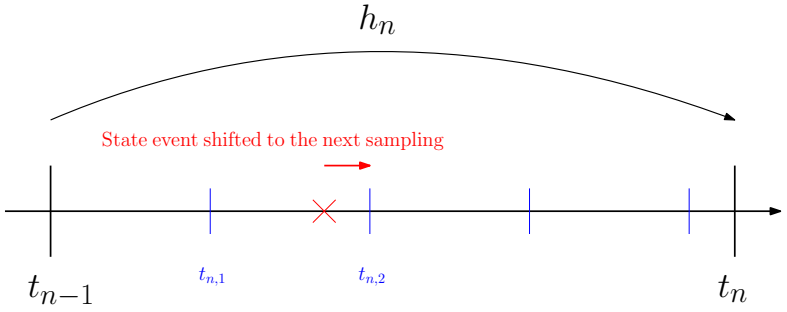

Figure 4: A scheme showing the state event shifted to the next time event in a system controller by digital controllers.

An example of this phenomenon is depicted in Fig. 4. The vertical lines indicate the sampling actions of the digital controller found in the time step

h_n, and the red cross shows the state event between two sampling actions. In other words, the red cross shows the time the conditions leading to the state event occurrence are satisfied. Nevertheless, due to its discrete nature, the digital controller is not able to act accordingly until the next sampling action. Therefore, the state event will be addressed with a delay. This means that while a system exclusively controlled with digital controllers is under simulation, the state events become time events, and no detection or time step reduction is required to land on state events.

Finally, it should be noted that, if there is a state event that is not going to be addressed by the digital controller, the only option is to typically reduce the time step to land on it. However, this will not be expensive for the solver since there are not many state events (compared to time events of digital controllers) happening during the simulation.

3. Proposed Interpolation-based Method

This section describes the proposed Interpolation-based Method (IBM) to handle the digital control actions. Instead of reducing the time step to land on the events defined by the digital control actions (noted as $t_{n,g}$ in Fig. 3), this method allows to take large time steps while including the impact of the controller's signal by interpolating the state variables at sampling times ($t_{n,g}$) of the digital controllers.

Let's consider the solution of a specific time-step $h_n = t_n - t_{n-1}$, as shown in blue in Fig. 3. The residual function of the implicit step to be solved for $t = t_n$ is denoted by $\boldsymbol{g}$:

$$\boldsymbol{g}(\boldsymbol{y}_n, e_{n,p_n}) = 0 \tag{6}$$

where only the last controller output before the time instant t_n is neces-

sary (e_{n,p_n}). However, the controller output e_{n,p_n} depends on the multiple controller actions between t_{n-1} and t_n. These are given by:

$$\begin{aligned} e_{n,1} &= \zeta(e_{n,0}, y_{n,1}, t_{n,1}) \\ &... \\ e_{n,g} &= \zeta(e_{n,g-1}, y_{n,g}, t_{n,g}) \\ &... \\ e_{n,p_n} &= \zeta(e_{n,p_{n-1}}, y_{n,p_n}, t_{n,p_n}) \end{aligned} \tag{7}$$

where $e_{n,0}$ is the controller output at the previous time-step and $\boldsymbol{y}_{n,g}$, $g \in [1, p_n]$, is the controlled system state variables at the intermediate time $t_{n,g}$ with an intermediate time step $h_{n,g} = t_{n,g} - t_{n-1}$.

In this work, we interpolate the intermediate state variables $\boldsymbol{y}_{n,g}$ using a polynomial function $\boldsymbol{w}_n$. Then, using the interpolated values, the controller's output signal is computed and the final value e_{n,p_n} is used in (6). One example of interpolation function is:

$$\begin{aligned} \boldsymbol{y}_{n,g} =& \boldsymbol{w}_n(t_{n,g}) = \boldsymbol{y}_{n-1} + h_{n,g}\dot{\boldsymbol{y}}_{n-1} \\ &+ \frac{h_{n,g}^2}{h_n^2}(\boldsymbol{y}_n - \boldsymbol{y}_{n-1} - h_n\dot{\boldsymbol{y}}_{n-1}) \end{aligned} \tag{8}$$

where a second-order Taylor expansion interpolator is used. It is noteworthy that to control and bound the integration method error, the integration method should have a higher order of local error than the interpolation polynomial [15]. Therefore, for most second-order integration methods, the Taylor interpolant polynomial with an order equal to the integration method (2nd order) has been found to be adequate.

The equations defined in (7) are included in the system of equations for

the solution of the time step t_n. Thus, the problem is extended by adding p_n new equations related to $\boldsymbol{e}_{n,g}$ ($g \in [1, p_n]$).

For the new extended problem, a new state variable vector $\boldsymbol{z}_n$ is defined combining the vector $\boldsymbol{y}_n$ and the controller's signals $e_{n,g}$ ($g \in [1, p_n]$):

$$\boldsymbol{z}_n = \begin{bmatrix} \boldsymbol{z}_{n,1} \\ \boldsymbol{z}_{n,2} \end{bmatrix} \tag{9}$$

with:

$$\boldsymbol{z}_{n,1} = \boldsymbol{y}_n \tag{10}$$

$$\boldsymbol{z}_{n,2} = \begin{bmatrix} e_{n,1} & e_{n,2} & \dots & e_{n,g} & \dots & e_{n,p_n} \end{bmatrix}^T \tag{11}$$

Also, a new residual vector is considered for the new state variable vector $\boldsymbol{z}_n$:

$$\tilde{\boldsymbol{g}} = \begin{bmatrix} \tilde{\boldsymbol{g}}_1 \\ \tilde{\boldsymbol{g}}_2 \end{bmatrix} \tag{12}$$

with:

$$\tilde{\boldsymbol{g}}_1(\boldsymbol{z}_n) = \boldsymbol{g}(\boldsymbol{y}_n, e_{n,p_n}) \tag{13}$$

$$\tilde{\boldsymbol{g}}_2(\boldsymbol{z}_n) = \begin{bmatrix} e_{n,1} - \zeta(e_{n,0}, \boldsymbol{y}_{n,1}) \\ \dots \\ e_{n,g} - \zeta(e_{n,g-1}, \boldsymbol{y}_{n,g}) \\ \dots \\ e_{n,p_n} - \zeta(e_{n,p_{n-1}}, \boldsymbol{y}_{n,p_n}) \end{bmatrix} \tag{14}$$

Consequently, and writing it in terms of interpolating polynomials:

$$\tilde{\boldsymbol{g}}(\boldsymbol{z}_n) = \begin{bmatrix} \boldsymbol{g}(\boldsymbol{y}_n, e_{n,p_n}) \\ e_{n,1} - \zeta(e_{n,0}, \boldsymbol{w}_n(t_{n,1})) \\ \cdots \\ e_{n,g} - \zeta(e_{n,g-1}, \boldsymbol{w}_n(t_{n,g})) \\ \cdots \\ e_{n,p_n} - \zeta(e_{n,p_{n-1}}, \boldsymbol{w}_n(t_{n,p_n})) \end{bmatrix} \tag{15}$$

Then, the new residual vector $\tilde{\boldsymbol{g}}$ can be solved for $\boldsymbol{z}_n$ to find $\boldsymbol{y}_n$.

3.1. Newton Iterations

At the m-th Newton iteration, the solution vector and interpolator function are defined as $\boldsymbol{y}_n^{(m)}$ and $\boldsymbol{w}_n^{(m)}$, respectively. Also, $y_{n,g}^{(m)}$ and $e_{n,g}^{(m)}$ are the value as at the interpolated steps. The IBM internal equations are thus expressed as:

$$\begin{aligned} \boldsymbol{y}_n^{(m)} &= \boldsymbol{w}_n^{(m)}(t_n) \\ \boldsymbol{y}_{n,g}^{(m)} &= \boldsymbol{w}_n^{(m)}(t_{n,g}), \quad \forall g \in [1, p_n] \\ e_{n,g}^{(m)} &= \zeta(e_{n,g-1}^{(m)}, \boldsymbol{y}_{n,g}^{(m)}), \quad \forall g \in [1, p_n] \end{aligned} \tag{16}$$

The Newton iteration is then solved with:

$$\boldsymbol{J}_n^{(m)}(\boldsymbol{z}_n^{(m+1)} - \boldsymbol{z}_n^{(m)}) = -\tilde{\boldsymbol{g}}(\boldsymbol{z}_n^{(m)}) \tag{17}$$

where $\boldsymbol{J}_n^{(m)}$ is the Jacobian matrix of the extended system (15) at the m-th Newton iteration, given by:

$$\boldsymbol{J}_n^{(m)} = \begin{bmatrix} \frac{\partial \tilde{\boldsymbol{g}}_1}{\partial \boldsymbol{z}_{n,1}} & \frac{\partial \tilde{\boldsymbol{g}}_1}{\partial \boldsymbol{z}_{n,2}} \\ \frac{\partial \tilde{\boldsymbol{g}}_2}{\partial \boldsymbol{z}_{n,1}} & \frac{\partial \tilde{\boldsymbol{g}}_2}{\partial \boldsymbol{z}_{n,2}} \end{bmatrix} \tag{18}$$

Computing $\boldsymbol{J}_n^{(m)}$ can be extremely challenging, especially when the smart digital controller is proprietary or black-box in nature. While there exist some automatic differentiation tools to compute the Jacobian in a numerical way [16], it should be noted that the equations and states at (14) relate to different time instances and the interpolation-based approximations can vary significantly between iterations.

To avoid these issues and accelerate the computations, an approximate Jacobian matrix can be used. For that purpose, the following approximation is employed:

$$\boldsymbol{J}_n^{(m)} = \begin{bmatrix} \frac{\partial \tilde{\boldsymbol{g}}_1}{\partial \boldsymbol{z}_{n,1}} & \frac{\partial \tilde{\boldsymbol{g}}_1}{\partial \boldsymbol{z}_{n,2}} \approx diag(\boldsymbol{0}_{p_n-1}, \frac{\partial \tilde{\boldsymbol{g}}_1}{\partial e_{n,pn}}) \\ \frac{\partial \tilde{\boldsymbol{g}}_2}{\partial \boldsymbol{z}_{n,1}} \approx \boldsymbol{0} & \frac{\partial \tilde{\boldsymbol{g}}_2}{\partial \boldsymbol{z}_{n,2}} \approx \boldsymbol{I}_{p_n} \end{bmatrix} \tag{19}$$

The cross derivative $\frac{\partial \tilde{\boldsymbol{g}}_2}{\partial \boldsymbol{z}_{n,1}}$ is set to zero to achieve a derivative-free method with respect to the controller input signal. The derivative of the controller to its own states is set to $\boldsymbol{I}_{p_n}$ in lack of more specific knowledge and to avoid rank deficiency of the matrix. In this way, the controller is treated as a black box with inputs and outputs. Therefore, this simplification may be necessary in the case of simulating non-equation-based controllers. However, the consequence of this simplification is that more Newton iterations may be needed to converge since we are ignoring the impact of state variables on the controller's output.

The flowchart of the proposed approach for one time step is depicted in Fig. 5. It should be noted that the new extended state variable vector z_n is used for the convergence test and the error estimation as it is reflected in the flowchart. Therefore, the controllers' outputs are also affecting the time step adjusting process.

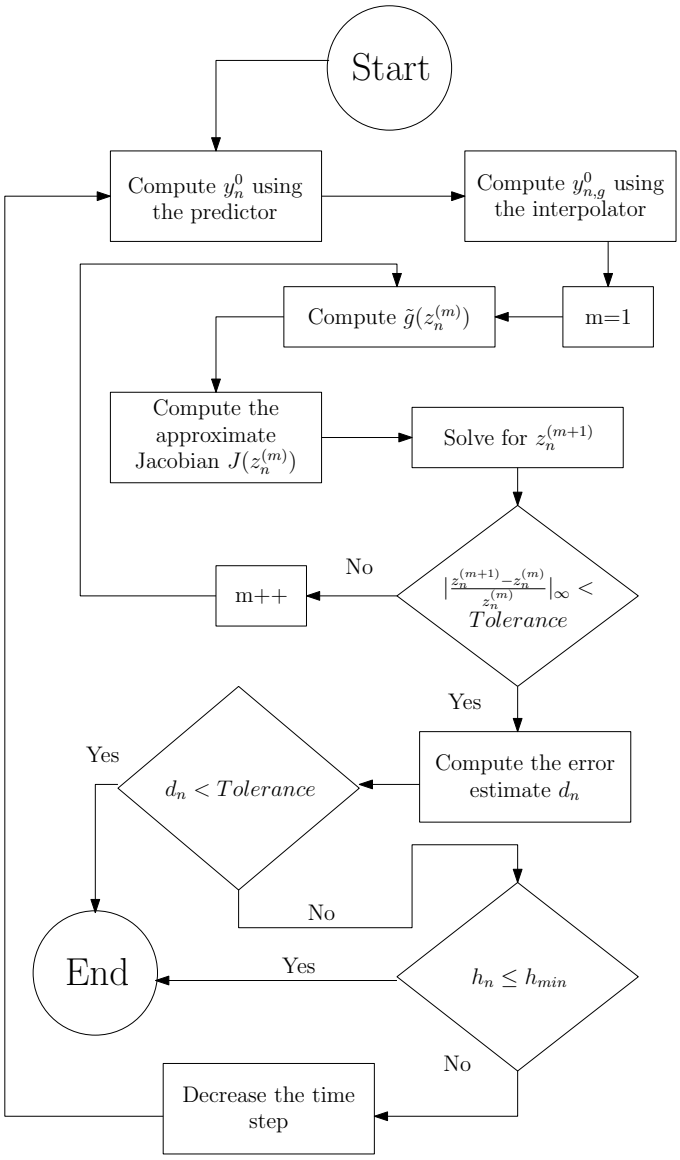

Figure 5: Flowchart of the interpolation-based method for one time step

4. Numerical Case Studies

In this section, the following three test cases are considered to evaluate the performance of IBM compared to the classical Step-size Reduction Method

(SRM), and the simplified simulation method (SSM) described briefly in Section 1:

1. A general system consisting of two ODEs controlled by an equation-based integral digital controller.
2. A single-machine infinite-bus (SMIB) system having a synchronous generator controlled by an equation-based governor digital controller and a black-box fuzzy AVR digital controller connected to an infinite bus.
3. A modified Kundur test system consisting of 4 synchronous generators and an HVDC link between two areas. The first two generators have equation-based AVRs while the rest have black-box fuzzy AVRs. The same equation-based governor digital controller is used for all of them. Furthermore, an equation-based power oscillations damping (POD) controller is considered for the HVDC link.

For all the case studies, a variable time-step predictor-corrector method using the pair of second-order Adams-Bashforth and Adams-Moulton [14] is used. Moreover, the number of bits considered for quantizing the controller's signal is equal to 18 and the minimum and maximum values for time steps are considered $h_{min} = 0.001$ s and $h_{max} = 1$ s, respectively. Also, the increasing and decreasing rates for altering the time steps are 1.25 and 0.5. Finally, it should be noted that in the case of SRM, lower time steps than h_{min} can be accepted if the distance between consecutive discrete events of digital controllers is smaller.

All the models and the solution algorithms are implemented in MATLAB and all methods involve the solution of the same DAEs.

4.1. A single integral controller

In this section, a test system controlled by a single integral controller is considered. Referring back to Fig. 2, for the continuous system a set of two differential equations formulated as (20) is considered:

$$\underbrace{\begin{bmatrix} \dot{y_1} \\ \dot{y_2} \end{bmatrix}}_{\dot{\boldsymbol{y}}} = \underbrace{\begin{bmatrix} a & b \\ -b & 0 \end{bmatrix}}_{\boldsymbol{A}} \underbrace{\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}}_{\boldsymbol{y}} + \underbrace{\begin{bmatrix} -b \\ 0 \end{bmatrix}}_{\boldsymbol{B}} e(t) \tag{20}$$

An integral controller with a gain $K_I = 0.07$ and sampling time $T = 0.1$ s is considered. The digital integral controller is defined as:

$$e_k = e_{k-1} + K_I T(u(kT) - y_2(kT)) \tag{21}$$

where $u(kT)$ is the set point (considered constant).

The simulated output for a step set point change is shown in Figs. 6 and 7 for one stable and one unstable system, respectively. As can be seen, no visible difference can be detected between the output trajectories of SRM and IBM. However, SSM is not able to find the correct steady-state nor follow fluctuations accurately. This is because SSM has been proposed for handling state events and they are usually scarce during a simulation. Therefore, shifting one event to the end of the time step can hardly cause a significant accuracy problem. On the contrary, in the case of digital controllers, multiple time events arising from the operation of multiple digital controllers in each time step are pushed to the end of the time step compromising the accuracy. Another reason causing the accuracy loss is due to considering only one time event per controller per time step shifted to the end of it. SSM cannot consider multiple interconnected time events arising from one digital

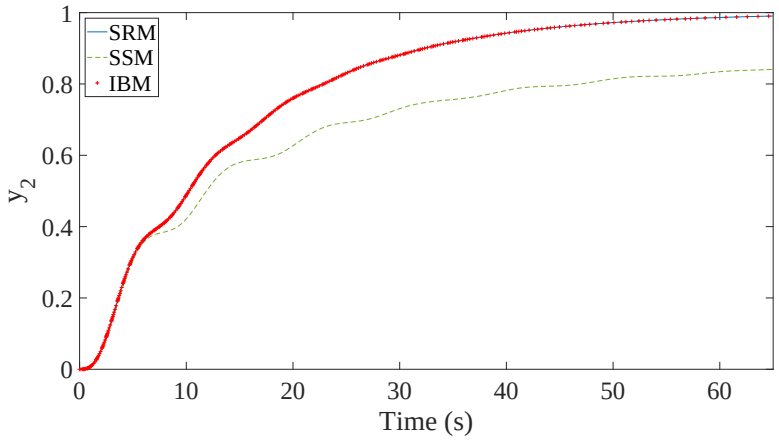

Figure 6: Output results for the stable system with the Integral controller

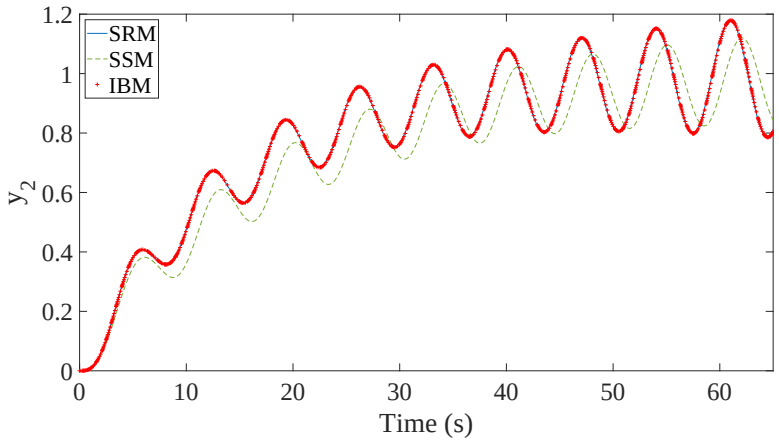

Figure 7: Output results for the unstable system with the Integral controller

controller since each controller output depends on the previous one with a different input coming from the system for a different point in time.

The total number of Newton iterations required to solve the system is summarized in Table 1 as a benchmark to assess the performance of IBM compared to SRM and SSM. It can be seen that, in both the stable and unstable scenarios, IBM is faster than SRM, having a performance similar to

Table 1: Performance comparison between IBM, SSM, and SRM for a system containing one Integral controller in terms of the number of Newton iterations

System parameters	SRM	SSM	IBM
Stable case	5473	810	1133
Unstable case	5333	2173	2403

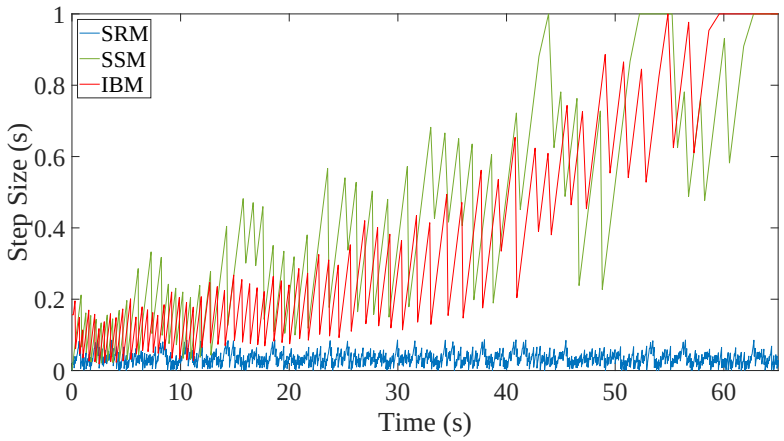

Figure 8: Step size results for the stable system with the Integral controller

SSM. This can be explained from Fig. 8 and Fig. 9, where the time-step size is plotted for all methods. While SRM is blocked by the digital controller sampling time, IBM manages to increase the time-step size without sacrificing accuracy.

4.2. Single machine model with excitation system and governor

In this section, a synchronous machine connected to an infinite bus (SMIB) by a tie of transmission lines is considered as shown in Fig. 10. The synchronous machine is controlled by a fuzzy-based excitation controller and a governor, as illustrated in Fig. 11. The DAEs describing the generator and

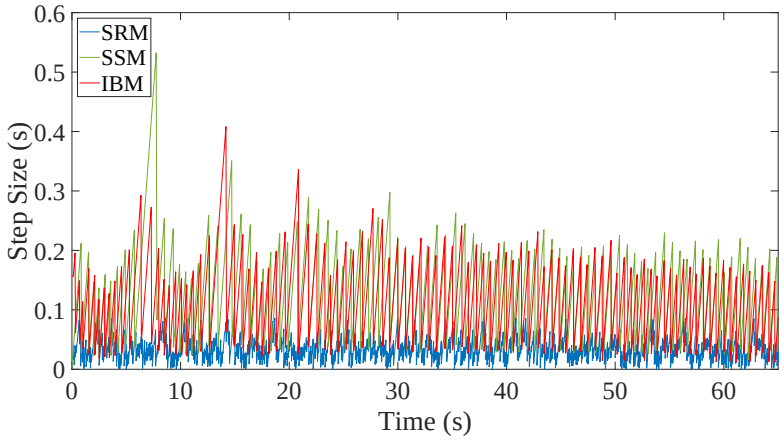

Figure 9: Step size results for the unstable system with the Integral controller

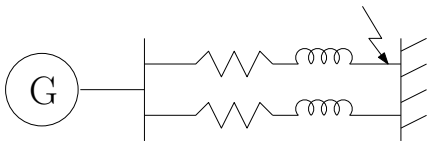

Figure 10: Schematic of the single machine infinite bus system

the schematics of the fuzzy-based exciter and governor systems are presented in Appendices Appendix A and Appendix B, respectively. It is noticeable that while the governor is an equation-based controller, the fuzzy AVR is non-equation-based and modeled as a black-box with an input and output [17]. The sampling time of the Governor and AVR are considered equal to 20 and 6 ms, respectively.

A short circuit with low impedance is applied at the end of the line connected to the infinite bus at time $t = 1$ s and cleared after 0.2 s by disconnecting the line. Figure 12 shows the output voltages of the generator. Furthermore, the output active and reactive power, and frequency of the generator are illustrated in Figs. 13, 14, and 15. In addition, the output

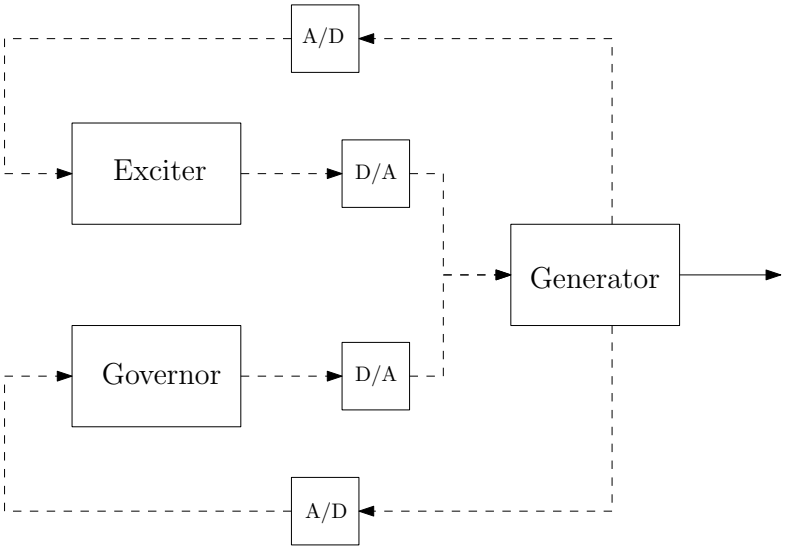

Figure 11: Overview of a synchronous machine with digital Governor and Exciter digital controllers

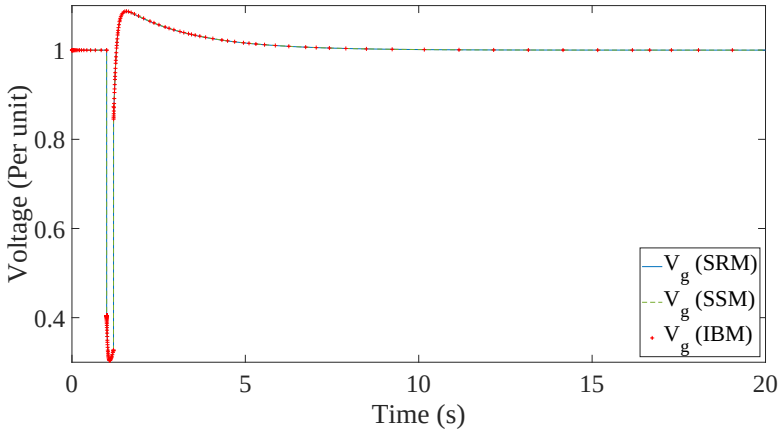

Figure 12: The generator's voltage output of SMIB system

signal for the excitation system and the governor is illustrated in Figs. 16 and 17. It can be seen that the simulation outputs are almost identical in all three methods for all the simulated signals.

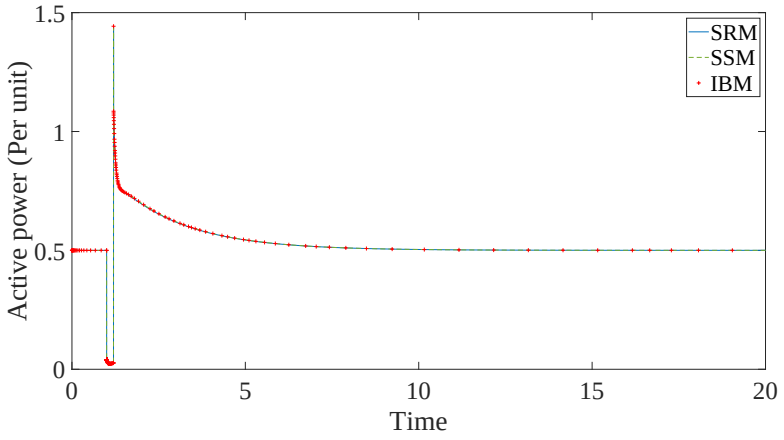

Figure 13: Active power of the generator of SMIB system

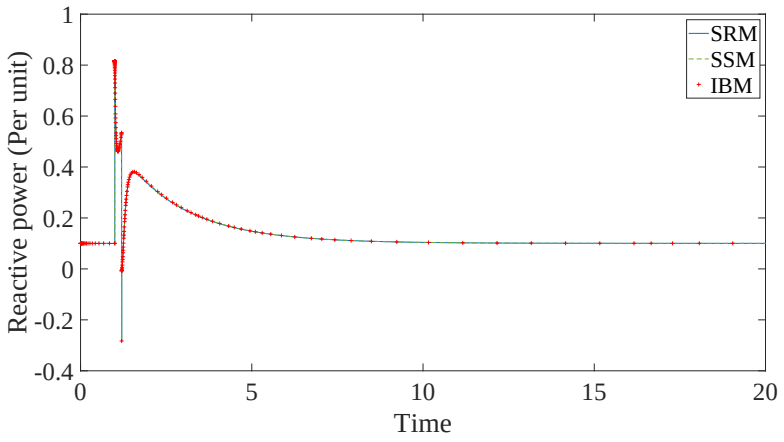

Figure 14: Reactive power of the generator of SMIB system

In addition, as can be seen in Fig. 17, the governor reaches its upper and lower limits equal to 0.9 and 0.1, showing the non-linear nature of the simulated model. This also shows the ability of IBM to catch the state events caused by the controller reaching its limits without the need to reduce the step size. As discussed in section 2.2, it is achieved by the fact that the state events translate to time events since they are not seen by the digital

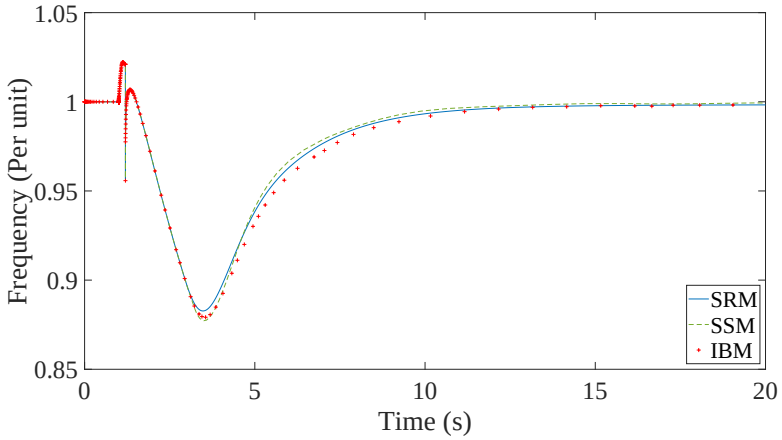

Figure 15: Frequency of the generator of SMIB system

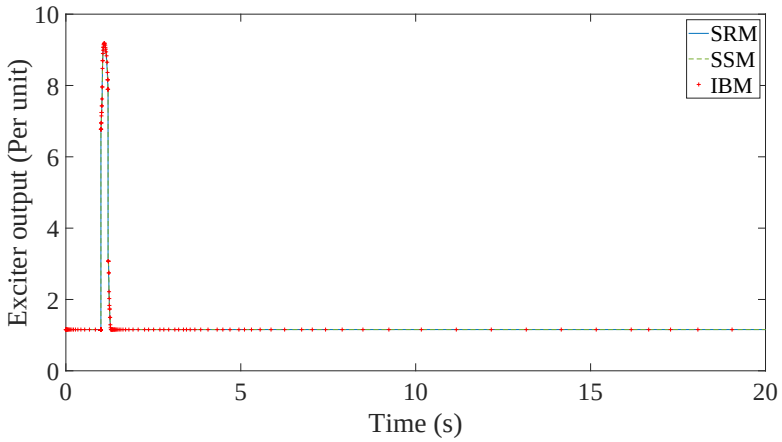

Figure 16: Output of the digital excitation system of the generator of SMIB system

controller before its next sampling. The same phenomenon happens for SRM and SSM as well since all the controllers are digital. However, SRM is forced to reduce the time step while SSM just shifts the sampling (therefore the state event) to the end of the time step. This delay can be seen in the same figure.

The number of Newton iterations and the execution time required to solve

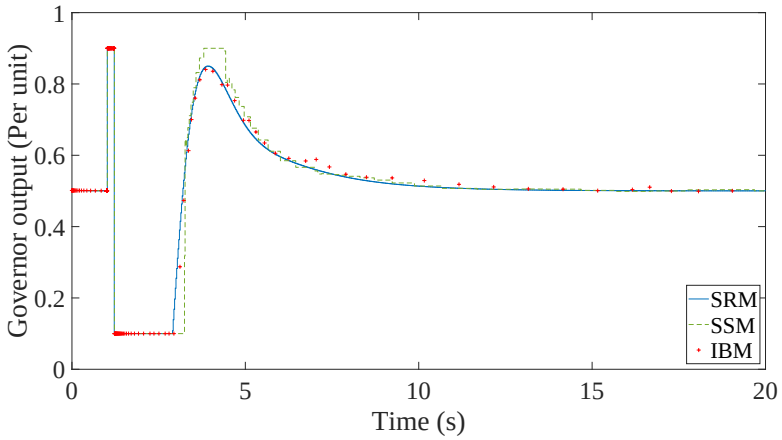

Figure 17: Output of the digital governor of the generator of SMIB system

Table 2: Performance comparison between IBM, SSM, and SRM for the SMIB system

Method	Number of Newton iterations	Run time (s)
SRM	25491	9.35
SSM	670	0.43
IBM	483	1.90

the described system are summarized in Table 2. It can be seen that IBM is about 52 and 5 times faster than traditional SRM in terms of Newton iterations and execution time, respectively. The step sizes taken to simulate the SMIB system are shown in Fig. 18, justifying the accelerated execution time. It is noticeable that SSM is the fastest while it still suffers from lack of accuracy as is evident in Fig. 17. However, it maintains better accuracy compared to the previous case study since the trajectories have fewer fluctuations.

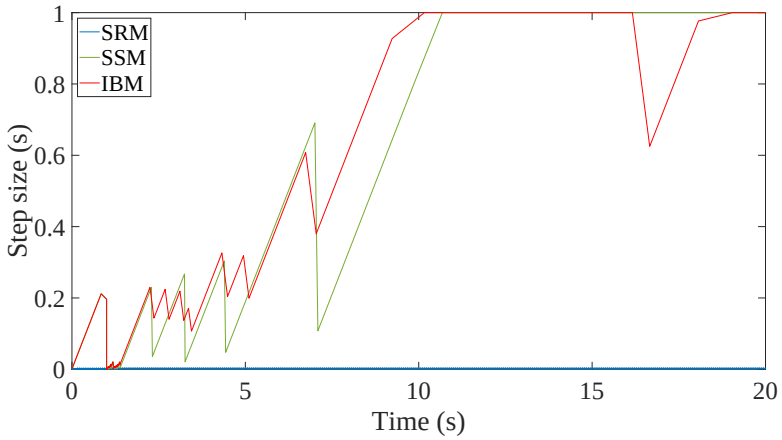

Figure 18: Step size results for SMIB system

4.3. Kundur test system

In this section, the Kundur test network [18], illustrated in Fig. 19 alongside its initial power flows, consisting of four synchronous generators, and two loads is considered as the test case. In addition, an HVDC link parallel with the tie between the two areas of the system is considered. Similar to the previous case study, each generator is controlled by a digital AVR and Governor. While the AVR model used for generators 1 and 2 is considered equation-based, the non-equation-based fuzzy AVR is utilized for generators 3 and 4. So, the system has 4 digital governors, 4 digital exciters for the generators, and a power oscillations damping (POD) controller for the HVDC link [19], therefore, 9 controllers in total. The sampling time of digital governors and digital exciters for generators 1 to 4 are considered 20 ms, 30 ms, 40 ms, 50 ms and 5 ms, 6 ms, 7 ms, 8 ms, respectively, and 90 ms for the POD controller. It should be noted that the sampling times were selected slightly different, and asynchronous to showcase the proposed method in a

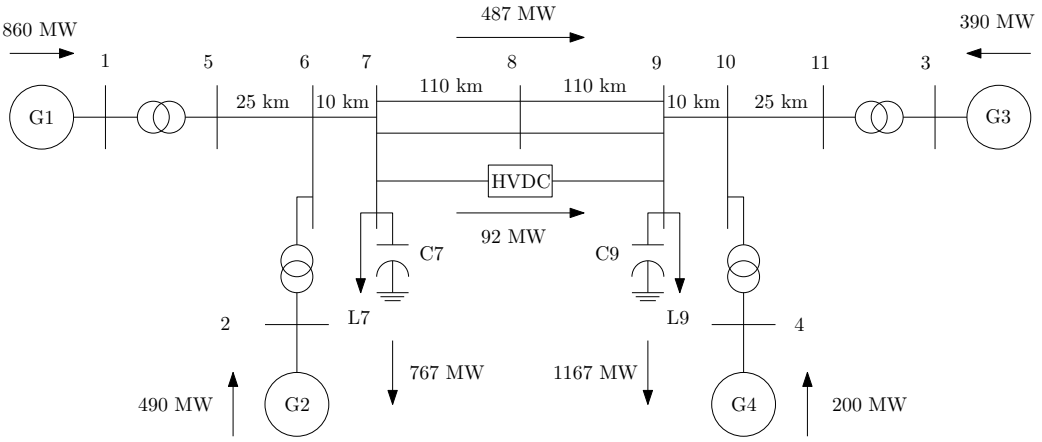

Figure 19: Schematic of modified Kundur network

more realistic setting. The DAEs describing the generator and the schematics of all the controllers are presented in Appendices Appendix A and Appendix B, respectively. For this case study, SSM fails to converge with the default setting of maximum time step size equal to 1 s since too many time events are neglected or shifted to the end of large time steps. Therefore, SSM's maximum time step size is limited to 0.1 s. This limits the number of time events falling in each time step.

A short circuit with a low impedance on bus 3 for 0.2 s is simulated as the disturbance. The voltage, speed deviation, AVR, governor, and POD output of the first generator are illustrated in Figs. 20, 21, 22, 23, and 24 respectively. It is shown that the simulated results are almost identical between SRM and IBM. Even though SSM's maximum time step size is limited, it still fails to provide an accurate solution, missing all the fluctuations from the point at which its maximum time step size is reached.

The performance of the methods is compared using the number of Newton

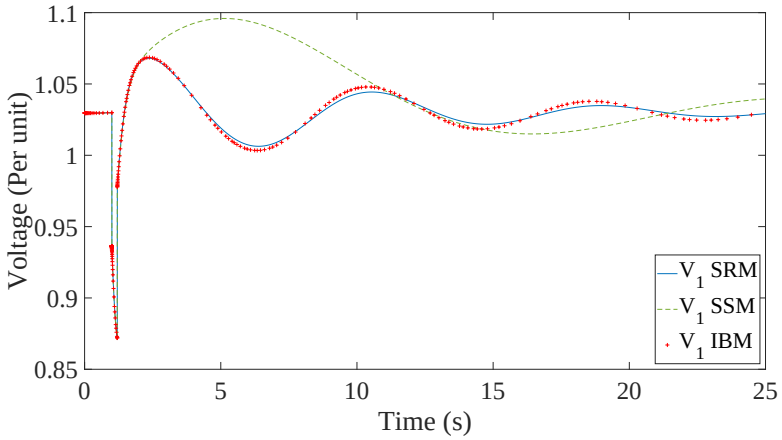

Figure 20: Bus 1 voltage magnitude of Kundur system

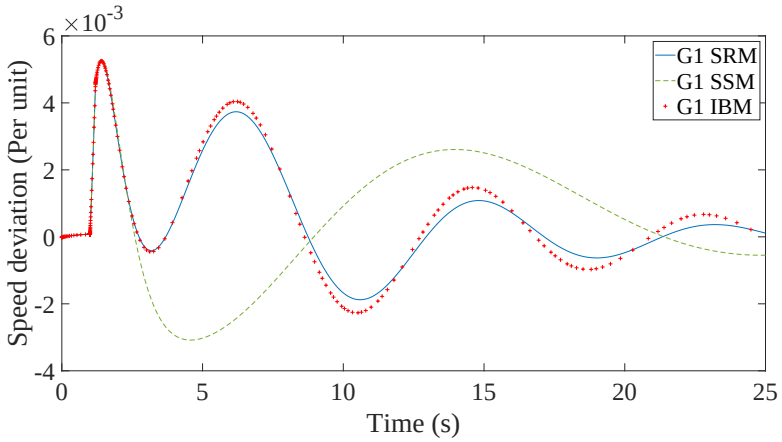

Figure 21: Speed deviation of generator 1 of Kundur system

iterations and the execution time required to solve the Kundur test network, and the results are summarized in Table 3. Again, it can be seen that IBM is much faster than SRM (about 16 times), as justified by the time-step size shown in Fig. 25. Also, by comparing all three case studies, it can be noted that, as the number of digital controllers of the system under simulation increases, SRM becomes slower at a much faster rate than IBM. The reason

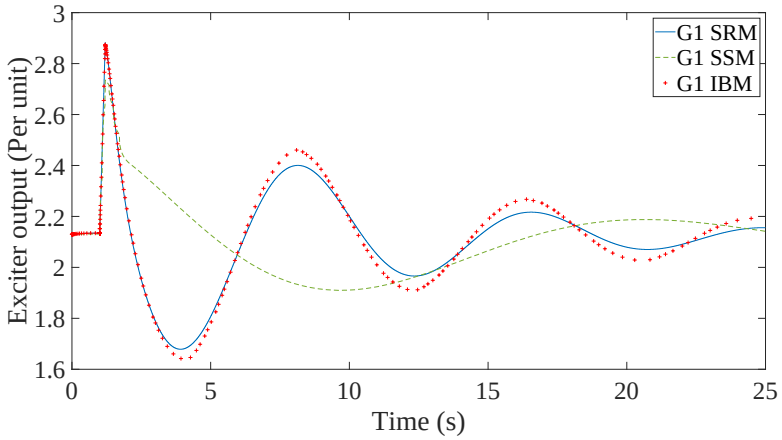

Figure 22: Output of the digital excitation system of generator 1 of Kundur system

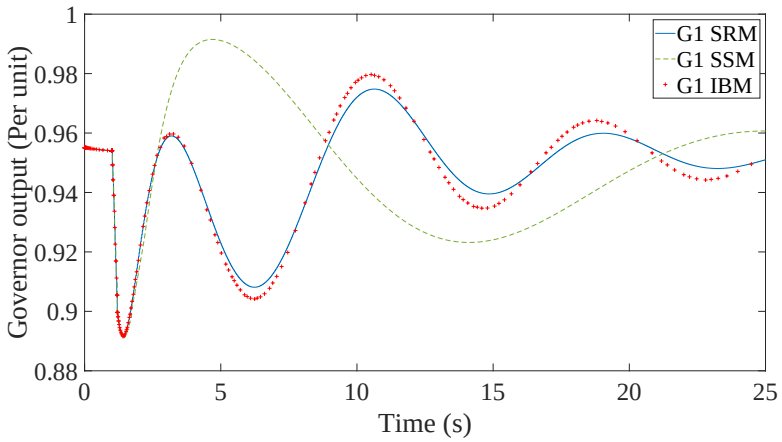

Figure 23: Output of the digital governor of generator 1 of Kundur system

is that as the number of controllers grows, the difference between two adjacent sampling times becomes smaller and limits the step size further while this is not the case in IBM. In the case of SSM, it can be seen that it is still the fastest method despite the limit on its time step size. Further limiting the maximum step size leads SSM to be slower than IBM, while its accuracy will not be improved much due to numerous numbers of events.

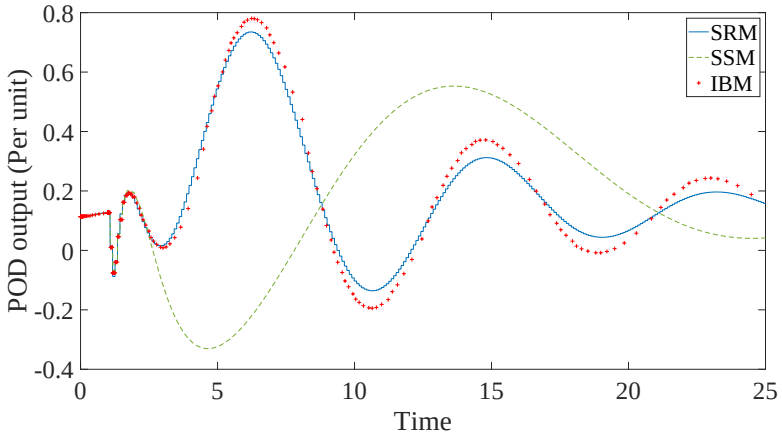

Figure 24: Output of the digital POD of Kundur system

Table 3: Performance comparison between IBM, SSM, and SRM for Kundur test system

Method	Number of Newton iterations	Run time (s)
SRM	64158	327.26
SSM	795	4.84
IBM	959	19.54

Table 4: Number of time events in every case study

Case study	Number of time events
A	650
B	4333
C	19347

Finally, the number of time events calculated based on the simulation time and controllers' sampling rate for all the case studies are summarized in Table. 4.

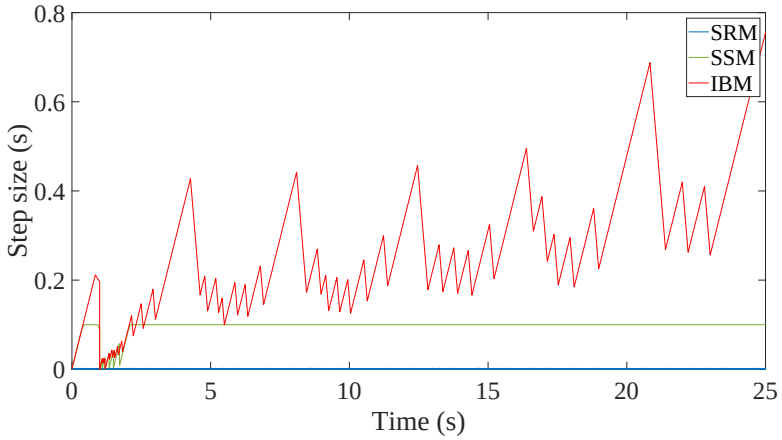

Figure 25: Step size results for Kundur system

5. Conclusion

In this paper, a novel, fast, accurate method is proposed for incorporating smart digital controllers in time-domain numerical simulations of power systems. Contrary to traditional methods, the proposed method does not impose step size reduction in case of discrete events imposed by the sampling of digital controllers. Instead, the impact of the digital controllers on the systems is handled internally to the time step, using an interpolation-based approach. Furthermore, the proposed method is able to integrate non-equation-based controllers modeled as a black box into the dynamic simulation of the system. It should be noted that IBM has been developed to capture the accurate dynamics of digital controllers in RMS simulations. The switching devices introducing time events in EMT simulations operate hundreds of times faster, forcing the EMT solvers to have time steps many times smaller than the digital controller sampling rate. This means for EMT simulations, the solver is already taking small enough time steps not to be

affected (further limited) by the sampling rate of digital controllers.

Three test cases are used to assess the performance and accuracy of the method compared to the classical step-reduction approach, and simplified simulation method. The results show good accuracy and high performance for IBM. Moreover, we showed that, as the number of controllers increases, the performance benefit becomes more noticeable. In addition, it was shown that the simplified simulation is not able to simulate systems containing multiple digital controllers effectively.

For future work, a large-scale test system study is under preparation, incorporating the proposed method in industrial-level simulation software. Moreover, an approach to incorporate the method in a special Modelica digital controller mock-up is made, to facilitate the use of the method without extensive changes to the system solvers. Also, the accuracy and performance of the method against non-smooth simulations such as cycling can be investigated.

Appendix A. Synchronous generator model

The DAEs of the second-order synchronous machine including the motion equations (fourth order in total) are formulated below [18].

- Differential equations:

$$\dot{\Delta\omega_r} = \frac{1}{2H}(T_m - T_e - K_D\Delta\omega_r) \tag{A.1}$$

$$\dot{\delta} = \omega_0\Delta\omega_r \tag{A.2}$$

$$\frac{1}{\omega_N}\dot{\psi_f} = v_f - R_f i_f \tag{A.3}$$

$$\frac{1}{\omega_N}\dot{\psi_{q1}} = -R_{q1} i_{q1} \tag{A.4}$$

- Algebraic equations:

$$0 = \psi_d - L_{dd} i_d - L_{df} i_f \tag{A.5}$$

$$0 = \psi_q - L_{qq} i_q - L_{qq1} i_{q1} \tag{A.6}$$

$$0 = \psi_f - L_{ff} i_f - L_{df} i_d \tag{A.7}$$

$$0 = \psi_{q1} - L_{qq1} i_q - L_{q1q1} i_{q1} \tag{A.8}$$

$$0 = v_d + R_a i_d + \psi_q \tag{A.9}$$

$$0 = v_q + R_a i_q - \psi_d \tag{A.10}$$

$$0 = v_d - \cos\theta_r^o v_x - \sin\theta_r^o v_y \tag{A.11}$$

$$0 = v_q - \sin\theta_r^o v_x + \cos\theta_r^o v_y \tag{A.12}$$

$$0 = i_d - \cos\theta_r^o i_x - \sin\theta_r^o i_y \tag{A.13}$$

$$0 = i_q - \sin\theta_r^o i_x + \cos\theta_r^o i_y \tag{A.14}$$

Appendix B. Controllers

The schematic of the AVR and the governor controlling the synchronous machine, and the POD controller is presented in Fig. B.26, Fig. B.27, and Fig. B.28, respectively. In addition, the values for the parameters of the

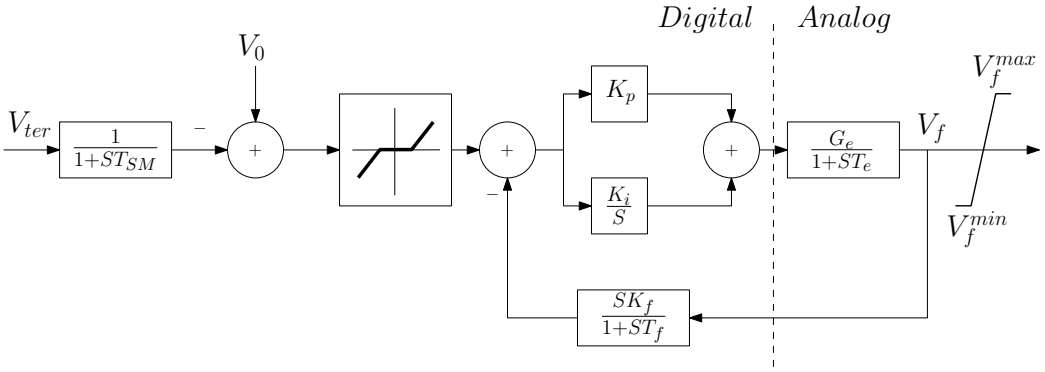

Figure B.26: Schematic of the AVR used to control the synchronous generator

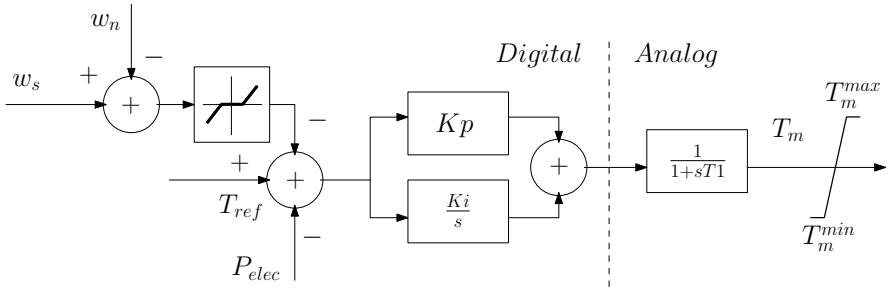

Figure B.27: Schematic of the governor used to control the synchronous generator

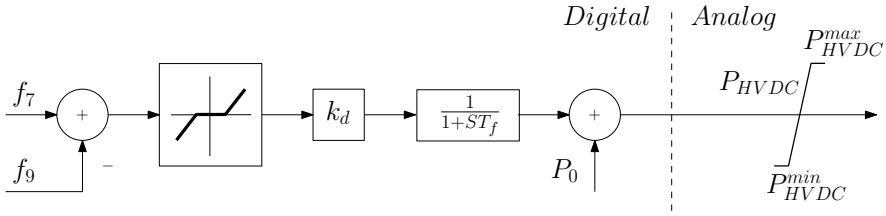

Figure B.28: Schematic of the POD controller used to control the HVDC line

exciter, the governor, and the POD controller are summarized in Table B.5, Table B.6, and Table B.7, respectively. Finally, the fuzzy AVR schematic is illustrated in Fig. B.29.

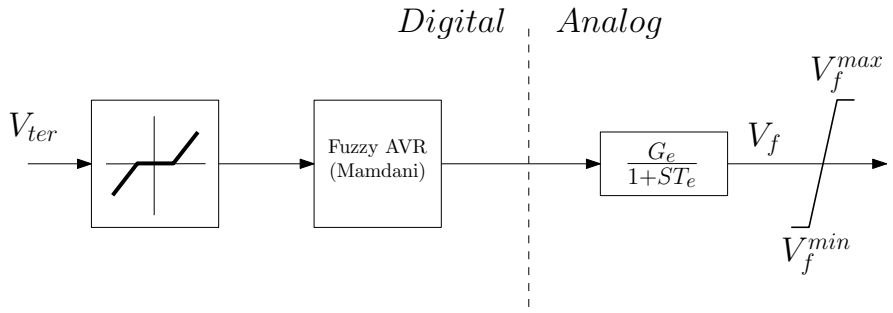

Figure B.29: Schematic of the fuzzy exciter used to control the synchronous generator

Table B.5: Parameter values for the AVR

Parameter	T_{SM}	K_p	K_i	G_e	T_e	K_f	T_f
Value	0.2	0.003	0.004	1	0.12	0.8	0.9

Table B.6: Parameter values for the governor

Parameter	K_p	K_i	$T1$
Value	12	0.2	0.3

Table B.7: Parameter values for the POD controller

Parameter	k_d	T_f
Value	8	0.1

References

[1] F. Milano, Power System Modelling and Scripting, Power Systems, Springer Berlin Heidelberg, 2012.

[2] D. Ellison, Efficient automatic integration of ordinary differential equa-

tions with discontinuities, Mathematics and Computers in Simulation 23 (1) (1981) 12–20.

[3] Y. Zhu, Z. Zhao, B. Shi, Z. Yu, Discrete state event-driven framework with a flexible adaptive algorithm for simulation of power electronic systems, IEEE Transactions on Power Electronics 34 (12) (2019) 11692–11705.

[4] D. Fabozzi, A. S. Chieh, P. Panciatici, T. Van Cutsem, On simplified handling of state events in time-domain simulation, Proceedings of the 17th PSCC (2011).

[5] A. F. Filippov, Differential equations with discontinuous righthand sides: control systems, Vol. 18, Springer Science & Business Media, 2013.

[6] L. Dieci, C. Elia, L. Lopez, On Filippov solutions of discontinuous DAEs of index 1, Communications in Nonlinear Science and Numerical Simulation 95 (2021) 105656.

[7] F. Zhang, M. Yeddanapudi, P. J. Mosterman, Zero-crossing location and detection algorithms for hybrid system simulation, IFAC Proceedings Volumes 41 (2) (2008) 7967–7972.

[8] M. A. A. Murad, F. Milano, Chattering-free modelling and simulation of power systems with inclusion of filippov theory, Electric Power Systems Research 189 (2020) 106727.

[9] M. A. A. Murad, B. Hayes, F. Milano, Application of Filippov theory to the IEEE Standard 421.5-2016 anti-windup PI controller, in: 2019 IEEE Milan PowerTech, 2019, pp. 1–6.

[10] M. Jafari, G. Bureau, M. Chiaramello, A. Guironnet, P. Panciatici, P. Aristidou, Modeling of Digital Controllers in Electric Power System Dynamic Simulations, in: 2023 IEEE Belgrade PowerTech, 2023, pp. 1–6.

[11] J. Na, H. Kim, H. Zhao, A. Gole, K. Hur, An improved high-accuracy interpolation method for switching devices in emt simulation programs, Electric Power Systems Research 223 (2023) 109630.

[12] M. Faruque, V. Dinavahi, W. Xu, Algorithms for the accounting of multiple switching events in digital simulation of power-electronic systems, IEEE Transactions on Power Delivery 20 (2) (2005) 1157–1167.

[13] K. Sayood, Introduction to data compression, Morgan Kaufmann, 2017.

[14] U. M. Ascher, L. R. Petzold, Computer methods for ordinary differential equations and differential-algebraic equations, Vol. 61, Siam, 1998.

[15] S. Glumac, Z. Kovačić, Defect Analysis of a Non-Iterative Co-Simulation, Mathematics 11 (6) (2023) 1342.

[16] H. M. Bucker, G. F. Corliss, A bibliography of automatic differentiation, LECTURE NOTES IN COMPUTATIONAL SCIENCE AND ENGINEERING 50 (2006) 321.

[17] M. Datta, T. Senjyu, Fuzzy control of distributed PV inverters/energy storage systems/electric vehicles for frequency regulation in a large power system, IEEE Transactions on Smart Grid 4 (1) (2013) 479–488.

[18] P. Kundur, N. J. Balu, M. G. Lauby, Power system stability and control, Vol. 7, McGraw-hill New York, 1994.

[19] H. F. Latorre, M. Ghandhari, L. Soder, Control of a VSC-HVDC operating in parallel with AC transmission lines, in: 2006 IEEE/PES Transmission & Distribution Conference and Exposition: Latin America, IEEE, 2006, pp. 1–5.