IEEE Open Access Journal of Power and Energy

Received 2 November 2022; revised 20 January 2023; accepted 8 February 2023.
Date of publication 20 February 2023; date of current version 24 February 2023.

Digital Object Identifier 10.1109/OAJPE.2023.3245040

PowerSAS.m—An Open-Source Power System Simulation Toolbox Based on Semi-Analytical Solution Technologies

JIANZHE LIU[1,2] (Member, IEEE), RUI YAO[3] (Senior Member, IEEE), FENG QIU[1] (Senior Member, IEEE), YANG LIU[1] (Member, IEEE), AND KAI SUN[4] (Senior Member, IEEE)

[1]Argonne National Laboratory, Energy Systems and Infrastructure Analysis Division, Lemont, IL 60439 USA
[2]Consortium for Advanced Science and Engineering, The University of Chicago, Chicago, IL 60637 USA
[3]Moonshot Factory, X Development LLC, Mountain View, CA 94043 USA
[4]Department of EECS, University of Tennessee, Knoxville, TN 37996 USA

CORRESPONDING AUTHOR: J. LIU (jianzhe.liu@ieee.org)

This work was supported in part by the Laboratory-Directed Research and Development (LDRD) Program of Argonne National Laboratory and in part by U.S. DOE Advanced Grid Modeling Program under Grant DE-OE0000875.

ABSTRACT In handling complex power system simulation tasks, semi-analytical solution (SAS) methods have proven to be numerically robust and computationally efficient. They provide a competitive alternative to traditional numerical approaches. Still, there is inadequate power system simulation software, especially the open-source tools, that implements this technology. This paper introduces PowerSAS.m, an open-source toolbox that closes this gap by providing SAS baseline simulation options for power system steady-state and dynamic simulations. At its core, it implements a novel SAS method and encloses various heuristics and simulation techniques to ensure enhanced computational performance. In case studies, we verify PowerSAS.m in benchmarking comparisons and demonstrate its functionalities in grid analysis scenarios.

INDEX TERMS Power system simulation software, computational modeling, power system stability, reliability and resiliency.

I. INTRODUCTION

POWER system simulations are fundamental and crucial for grid security and stability analysis [1], [2], [3]. In recent years, a pressing need has emerged to conduct power system simulations over an extended simulation horizon. On the one hand, a power system is evolving into a more complicated system with growing dynamical behaviors spanning multiple time-scales [4], [5]; on the other hand, it has witnessed a growing number of contingencies having rich transients as well as lasting duration [6].

In general, power system simulations involve solving a set of equations that describe the steady-state and dynamical behaviors of the power network and components along with their interactions [7]. These equations are usually nonlinear and complex. Their analytical solution is usually not available [8]. Many numerical methods have been developed to solve these equations. Differential equations representing the dynamical behaviors of power systems are usually solved by numerical integration methods [8], [9], [10], including both explicit methods (for example, the Modified Euler method and 4th order Runge-Kutta method) and implicit methods (for example, the Trapezoidal-rule method). Algebraic equations representing the steady-state behaviors of power systems are usually solved by numerical iteration methods [8], [9], [10], for example, the Newton-Raphson method and its variations. These methods have been widely used in commercial software and open-source power system simulation toolboxes [7]. However, these numerical methods may suffer from expensive computation burdens: numerical integration methods need to take a tiny time step to ensure accuracy and numerical stability; in some cases, even if the length of a time step is carefully chosen, numerical iteration methods can still fail to converge [11], [12].

The semi-analytical solution (SAS) method is an emerging simulation technology. The basic idea of the SAS method is decomposing the computation burden of solving power system differential equations or algebraic equations into two stages [13]. The offline stage derives an approximate and analytical solution of the power system differential or algebraic equations model, which is an explicit expression of symbolic variables of time, the initial state, and parameters. Such a solution is accurate for a certain time window whose length generally depends on the system model and the order of the SAS expression. In the online stage, the SAS is evaluated by substituting the numerical values obtained from real-time conditions into those symbolic variables. Such a process is conducted over consecutive time windows until the desired simulation length is reached. In the literature, a variety of SAS methods have been investigated, and they have shown promising results in power system simulations, including the Adomian decomposition method [13], [14], [15], a power series method [16], a holomorphic embedding method [11], [17], [18], [19], a differential transformation method [20], [12], [21], a Padé Approximation method [22], a continued fraction method [23], and a homotopy method [24], etc.

Despite the rich literature on SAS methods' applications in power system simulations, there is still inadequate open-source power system simulation software that implements this technology. Building on our continued and fruitful track record of fundamental research into power system simulation and stability analysis, for example, [11], [17], [18], [19], [25], [26], and [27], a new open-source power system simulation toolbox entitled PowerSAS.m based on novel SAS technologies is developed to close this gap.

In the current version, PowerSAS.m is developed in the MATLAB/Octave GNU environment and establishes a baseline SAS technology implementation in steady-state and dynamic simulations. At the core, its simulations are powered by a novel SAS method that has been developed recently. Simulation accuracy, numerical robustness, and computational efficiency have been shown in a variety of numerical testing and comparison studies. To better facilitate the SAS method's applications to power system simulations, the toolbox encloses many heuristic algorithms to enhance the computational performance further. For example, a multistage SAS computational algorithm is implemented to automatically adjust SAS coefficients to maintain simulation accuracy at a prescribed level; an integrated Padé approximation function can be called upon at the users' request to expand the region of approximation of the SAS method to enhance the numerical robustness, and a hybrid simulation scheme is developed to automatically switch the system model between dynamic and steady-state representations in order to improve scalability.

To facilitate power system security and stability assessment, PowerSAS.m accommodates a variety of built-in analysis functionalities and allows for flexible customization of grid analysis scenarios. Common analysis functionalities, including N-1 contingency analysis, continuation power flow analysis, and transient stability analysis, are directly provided by PowerSAS.m and can be easily prompted by using a simple Application Programming Interface (API). Besides, the toolbox supports an event-driven simulation scheme, where key simulation parameters, including the network topology, contingency events, or contingency duration, can be artificially specified to vary in a temporal sequence. This allows the users to create customizable and extensible grid analysis scenarios, for example, N-k contingency analysis and complicated event sequences in extended simulation terms.

The rest of the paper is organized as follows: Section II provides an overview of PowerSAS.m to summarize its key features; Section III discusses the SAS technology implemented in this toolbox; Section IV discusses some important simulation approaches enclosed in this toolbox; Section V introduces the grid analysis functionalities; Section VI uses numerical case studies to demonstrate the computational performance of the toolbox; Section VII concludes the paper and discusses future work.

II. OVERVIEW

This section aims to discuss the basics about PowerSAS.m, including the underlying technologies, its functionalities, and how the tool may be used and customized.

A. ABOUT PowerSAS.M

PowerSAS.m is a power system simulation toolbox based on semi-analytical solution (SAS) technologies. It is currently open-sourced under the Berkeley Source Distribution (BSD) license and is available on GitHub .[1]

Its current version is developed in MATLAB/Octave programming environment. It currently provides the following power system simulation functionalities:

1) power system steady-state simulations, including power flow (PF) calculations, extended PF analysis to find equilibria of models involving differential equations, and continuation power flow (CPF) analysis.
2) Power system dynamic simulations to support transient stability analysis and contingency analysis, etc.
3) Hybrid extended-term simulations to support large-scale dynamic power system simulations over an extended time.

B. UNDERLYING TECHNOLOGIES

PowerSAS.m are equipped with advanced simulation technologies.

The simulation engine of PowerSAS.m is based on a novel SAS method. Power system simulations always involve solving a set of nonlinear equations. In general, the analytical solution to such equations is unavailable. Traditional methods, such as numerical integration-based methods or Newton-Raphson methods, take iterative small steps to

[1] PowerSAS.m. Available: https://github.com/ANL-CEEESA/powersas.m

solve the equations. They may suffer from computational inefficiency or divergence issues. Unlike these traditional methods, an SAS method finds an approximated analytical form of solution to a nonlinear system, avoiding the use of iterations and small simulation steps. SAS methods have been widely used in complex scientific simulation tasks and have shown advanced computational performance. Nonetheless, there is still lacking power system toolboxes that incorporate SAS technologies for extended-term power system simulations. PowerSAS.m closes this gap. It encases a novel SAS method specially tailored for power system simulations. Depending on whether there exists a switching event in the simulation, two types of SAS formulations are developed. For the two types of formulations, a unified method is developed to calculate the SAS coefficients based on simple linear operations. A detailed discussion about the SAS method is provided in Section III.

In addition, PowerSAS.m encloses a variety of heuristics and simulation techniques that further enhance computational performance. For example, to improve numerical robustness, PowerSAS.m provides the option to use Padé approximation methods to process the SAS simulation results, and this approximation expands the region of convergence of the power-series-based solution. To ensure simulation accuracy, PowerSAS.m actively monitors the SAS simulation results and adaptively updates SAS coefficients to maintain simulation results fidelity in a given boundary. Moreover, to enhance computational efficiency, PowerSAS.m has a heuristic algorithm to simplify the system model and simulation tasks while ensuring simulation quality. A detailed introduction is given in Section IV.

C. GRID ANALYSIS FUNCTIONALITIES

PowerSAS.m supports a variety of grid analysis tasks.

In its current version, PowerSAS.m provides five basic functionalities, including power flow analysis, continuation power flow analysis, line outage contingency analysis, N-1 line outage screening, and transient stability analysis with three-phase grounding faults. Users can use a simple Application Programming Interface (API) to call these functionalities. Each functionality corresponds to one common power system analysis scenario to allow users to apply the tool to some standard analysis tasks quickly.

Aside from the basic functionalities, PowerSAS.m supports customizable grid simulation, expanding the breadth of the tool's applicability. PowerSAS.m has an event-driven simulation scheme where one can use atomic events to structure a user-defined simulation scenario with a temporal sequence of events. These atomic events include adding/tripping lines/synchronous generators/induction motors/loads, etc. Using this flexible and extensible simulation scheme, PowerSAS.m can support more sophisticated and flexible applications.

A detailed introduction of the PowerSAS.m functionalities is given in Section V.

D. USING PowerSAS.M

The users can easily install PowerSAS.m and customize power system simulations in various use cases.

In the directory of the source code, a user only needs to execute command `setup` to complete the installation of all the functions of PowerSAS.m. For testing, users can first execute command `initpowersas` to initiate the environment and then execute `test_powersas` to run a set of predefined test cases. The testing includes a steady-state load flow study of the IEEE 3-bus system and the ACTIVSg-70k test case, respectively, along with a dynamic simulation test case based on a 528-bus reduced U.S. Eastern Interconnection (EI) model.

Code Block 1 Customizable API to call PowerSAS.m functionalities

```
% call PowerSAS.m functionalities
res=runPowerSAS(simType,data,options,varargin);

% plot results
plotCurves(1,res.t,res.stateCurve,...
res.SysDataBase,'v');
```

A user can customize a simple high-level API to apply all the aforementioned PowerSAS.m functionalities in a user-defined test case. An example API is shown in Code Block 1. It is a two-line MATLAB script that calls only one function, `runPowerSAS`, to run the simulations and one function, `plotCurves`, for result visualization.

The function `runPowerSAS` has four input arguments that specify the simulation parameters. For example, the first input argument defines the simulation type. It can take values from six candidate options corresponding to the five basic functionalities and the extensible simulation scheme. The second input argument specifies the simulation data, which are structured according to the PSAT format. Note that a user may use the open-source PSAT software to convert data files from other formats to the PSAT format.

The function `plotCurves` show the simulation results. This function allows the users to specify the variables to show in the plot. For example, in the example, we choose to show the voltage magnitude. One can also choose to plot voltage angles, rotor angles of synchronous generators, deviations of synchronous generator rotor speed, induction motor slips, and frequency.

III. BASICS ABOUT SAS TECHNOLOGIES

Power system simulations involve finding solutions to nonlinear equations. SAS is a powerful simulation framework to solve these equations. PowerSAS.m is primarily based on SAS technologies. In this section, we first review the basics of the SAS methods, then we introduce the novel SAS technologies developed for and implemented in PowerSAS.m.

A. BASICS ABOUT SAS

1) THE NUMERICAL PROBLEM

In power system applications, computer-based simulations are instrumental in understanding the system's behaviors in different scenarios. The simulation results are often obtained by solving a mathematical model consisting of differential-algebraic equations (DAEs):

$$\dot{x}(t) = f(x(t), y(t), p(t)), \tag{1a}$$

$$0 = g(x(t), y(t), p(t)), \tag{1b}$$

where t represents time, x is the dynamic state variable, y is the algebraic state variable, p represents the system parameter, and f and g are some functions describing the system behaviors. Like many real-world applications, in most power system applications, f and g are both holomorphic functions, meaning that they are locally infinitely differentiable and equal to their Taylor expansions in the complex space.

Let x_0 and y_0 represent some initial values for x and y, respectively, and let T be the end time of a simulation time horizon of interest. Power system numerical simulations often need to solve the following initial-value problem (IVP):

Given the initial condition $x(0) = x_0$ and $y(0) = y_0$, and the system parameter $p(t)$ for $t \in [0, T]$, find $x(t)$ and $y(t)$ for $t \in [0, T]$ that satisfy equations (1).

For example, power system engineers routinely use numerical simulation methods to conduct the N-1 analysis, where, for instance, the power system states' deviation from a given operating point after a line outage event is calculated to understand whether there are voltage stability or operational constraint violation risks.

The fact that the analytical solution of system (1) is usually unavailable makes it necessary to use numerical methods to approximate the solutions [11].

2) NUMERICAL INTEGRATION METHODS

Conventional numerical integration methods solve this IVP problem by approximating the exact solution with a series of discrete "approximated solutions". For example, let Δt be the elapsed time of one time step, numerical integration methods seek to find a sequence $\{\hat{x}(k \cdot \Delta t)\}$, where $k = 0, \cdots, \lfloor T/\Delta t \rfloor$ [2] and $\hat{x}$ is the approximated solution. This sequence aims to provide a sufficiently close approximation to the continuous state variable $x(t)$ for $t \in [0, T]$. These methods need to take some computation sub-steps to calculate each $\hat{x}(k \cdot \Delta t)$ iteratively.

The limitations of numerical integration methods include: 1) Such a simulation method usually introduces a simulation error that increases drastically with the value of Δt. Thus, the simulation time step is usually chosen to be small to reduce errors, which in turn might make the iterative calculation of the sequence $\{\hat{x}(k \cdot \Delta t)\}$ computationally dragging; 2) the inclusion of algebraic equations (1b) may cause some convergence issues.

[2]Note that $\lfloor a \rfloor$ finds the largest integer that is smaller than a scalar a.

3) GENERAL SEMI-ANALYTICAL SOLUTIONS METHODS

Different from the numerical integration methods, SAS methods seek to find a continuous approximated solution of (1). The approximated solution has a predetermined "semi-analytical" expression, which is usually a high-order power series.

In general, a SAS method constructs an augmented system of (1) as follows:

$$0 = h(\dot{x}(s), x(s), y(s), p(s), s), \tag{2}$$

where s is a new variable in the complex space, and h is the augmented SAS model.

In this construction, the solution and parameter are usually assumed to be high-order power series:

$$\hat{x}(s) = \sum_{k=0}^{\infty} x[k]s^k, \hat{y}(s) = \sum_{k=0}^{\infty} y[k]s^k, \hat{p}(s) = \sum_{k=0}^{\infty} p[k]s^k, \tag{3}$$

where the variables accented by $\hat{}$ represent the SAS approximated solution, $x[k]$, $y[k]$, and $p[k]$ are the coefficients of the kth coefficients of $\hat{x}(s)$, $\hat{y}(s)$, and $\hat{p}(s)$, respectively.

It is common practice to construct h to ensure: 1) Its solution is much easier to obtain than (1), and 2) it is homotopic to (1), which means that by varying s the solution of (2) converges to that of (1).

The structure of h and where s is embedded are often ad hoc and case-sensitive. Naturally, their design is key to developing the SAS methods.

B. NOVEL SAS DESIGN OF PowerSAS.M

PowerSAS.m considers two general types of simulation scenarios, depending on whether there exists a switching event in the power systems. A switching event stands for the cases where there is a sudden adding/tripping of a component (e.g., a synchronous generator, a power line, etc.) that makes some system parameters and state variables suddenly switch to another value, causing discontinuity with respect to time.

We develop novel SAS methods to cope with simulation tasks with or without a switching event. In either case, we show that PowerSAS.m transforms the complex computational tasks of solving nonlinear DAE equations into solving only linear algebraic equations.

1) NO SWITCHING EVENT

If there is no switching event, the power system model (1) is usually holomorphic with respect to time. Consequently, we will structure the approximated solutions in terms of t instead of introducing a new variable. The format of the approximated solutions and parameters is given by:

$$\hat{x}(t) = \sum_{k=0}^{\infty} x[k]t^k, \hat{y}(t) = \sum_{k=0}^{\infty} y[k]t^k, \hat{p}(t) = \sum_{k=0}^{\infty} p[k]t^k, \tag{4}$$

Note that we need to truncate these power series in PowerSAS.m for computational tractability, and the users can customize the order that the series is truncated. Note that the

TABLE 1. General rules for obtaining Equation (5).

Operator	kth order LHS	Lower-Order RHS
$x+y$	$x[k]+y[k]$	none
xy	$x[0]y[k]+y[0]x[k]$	$-\sum_{m=1}^{k-1} x[m]y[k-m]$
$f_{\text{div}}=\frac{x}{y}$	$\frac{1}{y[0]}x[k]-\frac{f_{\text{div}}[0]}{y[0]}x[k]$	$\frac{1}{y[0]}\sum_{m=1}^{k-1} f_{\text{div}}[k]y[k-m]$
$f_{\text{der}}=\frac{dx}{dt}$	$(k+1)x[k+1]$	none
$f_{\ln}=\ln x$	$\frac{1}{x[0]}x[k]$	$-\frac{1}{kx[0]}\sum_{m=1}^{k-1} mx[m]f_{\ln}[k-m]$
$f_{pow}=x^a$	$a\frac{x[k]}{x[0]}f_{\text{pow}}[0]$	$-\frac{a}{kx[0]}\sum_{m=1}^{k-1} mx[m]f_{\text{pow}}[k-m]+\frac{1}{kx[0]}\sum_{m=1}^{k-1} mf_{\text{pow}}[m]x[k-m]$
$f_{\text{exp}}=^x$	$x[k]f_{\text{exp}}[0]$	$-\frac{1}{k}\sum_{m=1}^{k-1} mx[m]f_{\text{exp}}[k-m]$
$f_{\ln}=\ln(x)$	$\frac{1}{x[0]}x[k]$	$-\frac{1}{kx[0]}\sum_{m=1}^{k-1} mx[m]f_{\ln}[k-m]$
$f_{\sin}=\sin(x)$	$x[k]f_{\cos}[0]$	$-\frac{1}{k}\sum_{m=1}^{k-1} mx[m]f_{\cos}[k-m]$
$f_{\cos}=\sin(x)$	$-x[k]f_{\sin}[0]$	$\frac{1}{k}\sum_{m=1}^{k-1} mx[m]f_{\sin}[k-m]$

coefficient of the order-0 term is known, which is the given initial condition.

Then, substituting these SAS approximated solutions into (1) yields a new set of equations, which describes the coupling between the coefficients. These equations are then equivalently transformed into polynomials of the coefficients, following the rules summarized in Table 1. After combining terms in the same order, the following linear equation is obtained:

$$A(z[0])z[k]=b(z[0],\cdots,z[k-1]), \tag{5}$$

where $z[k]$ encapsulates all the coefficients at kth order, $A(z[0])$ is a constant matrix concerning only the initial condition, and $b(z[0],\cdots,z[k-1])$ is a function of the coefficients at all lower orders.

Finding each $z[k]$ is cast into a classic "$Ax=b$" problem. In solving an IVP, the initial condition $z[0]$ is known. Hence, a recursive solution algorithm is designed to find the coefficients from order 1 until the desired order. It is worth noting that it is only necessary to find $A(z[0])$ once at the beginning of the recursive iteration, as it concerns only the initial condition. At the kth order, coefficients of all lower orders are known. Subsequently, $b(z[0],\cdots,z[k-1])$ is a known vector as well in (5). With given "$A(z[0])$" and "$b(z[0],\cdots,z[k-1])$", the major computational tasks of finding $z[k]$ involves solving a classic "$Ax=b$" linear problem.

2) WITH SWITCHING EVENTS

PowerSAS.m considers a comprehensive set of switching events, including adding/tripping power lines, synchronous generators, induction motors, static loads, etc. In the following, we show the general method PowerSAS.m adopts to cope with the switching events. We use the subscript "$-$" (resp., "$+$") to represent a pre-switching (resp., post-switching) function or variable.

We will construct the following SAS models connecting pre-switching and post-switching systems. Due to the underlying continuity of (1a), $x_+(t)=x_-(t)$. Then based on different types of events, the SAS models for solving the post-switching algebraic variables $y_+(t)$ can be constructed as (here we omit the time t because the instant has been set) an s-embedded system $g_s(x_+,y_s(s),p_s(s))=0$ such that when $s=0$, the equation is the pre-switching algebraic equation $g(x_-,y_-,p_-)=0$ and when $s=1$, the equation is the post-switch equation $g(x_+,y_+,p_+)=0$. Note that the pre-switching and post-switching systems form a homotopy with the construction of the s-embedded system, meaning that the latter solution can be obtained from the former through varying s. By solving the s-embedded system using SAS and we can get the post-switch algebraic variable $y_+(t)=y_s(s=1)$.

The entire simulation horizon is divided into several non-switching periods, separated by switching events. Consequently, the pre-switching condition is always available as it is either the initial condition of the simulation or the terminal condition of a non-switching simulation period. As a result, if $s=0$, we have a trivial solution to the s-embedded system.

As for solving for the coefficients, we use (3) as the form of the semi-analytical solutions. By substituting the SAS approximated solutions, we can again find a similar "$Ax=b$" expression of the coefficients at order-k as shown in (5). The coefficients can then be obtained through a recursive algorithm, as discussed in the previous subsection.

IV. SIMULATION TECHNIQUES

In this section, we discuss some key simulation techniques adopted by PowerSAS.m, including heuristics aiming to improve the simulation accuracy, numerical robustness, and computational efficiency.

A. MULTI-STAGE SIMULATIONS

To ensure simulation accuracy, a multi-stage simulation method is implemented. As discussed in Section III-B, SAS methods find an approximated analytical solution of the nonlinear equations. The accuracy of the approximated solution, i.e., the imbalances of the equations, may increase along with

the simulation time. To reduce such imbalances, multi-stage simulation heuristics are developed.

Specifically, PowerSAS.m will compare the imbalances with a predefined threshold. If the imbalances surpass the threshold at a time instant, the SAS coefficients will be derived again using the methods detailed in Section III-B. The initial condition will be set at a previous time instant where the imbalances are smaller than the threshold. All the original simulation results before that selected time instant will be kept, while the simulation results after will be recalculated using the updated coefficients.

B. PADÉ APPROXIMATION

For computational efficiency and robustness improvement, the Padé approximation method is applied. PowerSAS.m uses truncated power series to approximate the nonlinear system solutions. The Padé approximant is a well-performed post-processing method to enhance the numerical robustness for a power series [2]. For SAS methods, the use of Padé approximation can further expand the region of approximation as proven in numerous numerical testing [18], [22]. Specifically, let N be the desired order to which a power series is truncated at. For $\hat{x}(t) = \sum_{k=1}^{N} x[k]t^k$, the Padé approximant has the following form:

$$\hat{x}(t) = \frac{a[0] + a[1]t + \cdots + a[m]t^m}{b[0] + b[1]t + \cdots + b[m]t^m}, \tag{6}$$

where $a[\cdot]$ and $b[\cdot]$ are both coefficients to be determined, m and n are positive integers satisfying $m + n = N$. PowerSAS.m allows the users to specify whether to apply the Padé approximant method. It can quickly obtain the Padé coefficients using a vectorized calculation technique. For the sake of brevity, the detailed explanation is omitted. Interested readers can find detailed derivations in [18].

C. HYBRID SIMULATION SCHEME

For computational efficiency enhancement, a hybrid simulation scheme is available in PowerSAS.m. The tool can support both (quasi-)steady-state and dynamic simulations. Furthermore, it provides the option to conduct a hybrid simulation where it switches between the two simulation schemes. When conducting an extended-term dynamic simulation on a large-scale power system, this hybrid simulation scheme enhances the computational efficiency while capturing the transient performance. In this subsection, we briefly introduce this simulation scheme.

1) MODEL TO FIND STEADY-STATE

While the original DAE system can be directly used in the dynamic simulations, PowerSAS.m still needs to derive an algebraic model from representing the steady state of the DAE system. This is because the algebraic equation (1b) may not fully capture the steady state of the differential equations (1a). PowerSAS.m automatically derives the full-scale algebraic equation that fully represents the steady-state physics of the original system.

2) SWITCHING HEURISTICS

In this hybrid simulation scheme, a switching criterion is implemented to govern the switching action. Intuitively, this switching criteria identifies whether the system is operating in a steady state. Unlike traditional methods that design switching criteria based on the simulated fluctuation level of the system trajectories, PowerSAS.m designs this switching criterion prior to simulations.

Recall that the SAS approximated solution or the Padé approximant involves finding power series that have the following properties: 1) the power series is truncated; hence there are only finitely many terms in the power series; 2) the power series have finitely many coefficients, and they are known; 3) the variable of the power series satisfies a box constraint (e.g., $t \in [0, T]$). Therefore, the power series is refrained by a box constraint. PowerSAS.m has a heuristic algorithm to find a tight box constraint of such a power series. The algorithm has a computational complexity that is linear to the system scale; hence it only introduces a trivial computational burden. Using the obtained box constraint, one can find the maximum fluctuation of the solution without conducting the simulations. If the maximum fluctuation is smaller than a threshold, the system is considered to be entering a steady state. PowerSAS.m will subsequently switch to steady-state simulation mode.

V. SOFTWARE PACKAGE FUNCTIONALITIES

PowerSAS.m supports various power system analysis functionalities. As shown in Fig. 1, it has five built-in basic functionalities and an extensible simulation scheme to fit different application scenarios.

A. BASIC FUNCTIONALITIES

There are five built-in grid analysis functionalities in PowerSAS.m. They are ''power flow analysis'', ''continuation power flow analysis'', ''line outage contingency analysis'', ''N-1 line outage screening'', and ''transient stability analysis after three-phase faults''. These are analysis tasks commonly conducted to assist in power system operation and planning decision-making process.

Code Block 2 Example: PowerSAS.m N-1 Analysis

```
% EXAMPLE: N-1 Analysis with PowerSAS.m

% call PowerSAS.m basic functionality
res=runPowerSAS('n-1','d_ei_528.m');
```

Users can use a simple API to use the basic functionalities and view the simulation results. For example, we show how the N-1 screening functionality is called upon in Code Block 2. The script uses the function `runPowerSAS` to specify the simulation type (`'n-1'`) and the test system (`'d_ei_528.m'` which stands for the reduced 528-bus EI system).

In testing, one can select one from the following five values in the first input argument to specify each of the five basic functionalities:

- `'pf'`: Power flow analysis. In addition to the conventional power flow model, PowerSAS.m supports an extended power flow to solve the steady state of dynamic models as well. For example, it will calculate the rotor angles of synchronous generators and slips of induction motors in addition to the network equations.
- `'cpf'`: Continuation power flow analysis.
- `'ctg'`: Contingency analysis. PowerSAS.m computes the system states immediately after removing power lines.
- `'N-1'`: N-1 screening. PowerSAS.m performs a series of contingency analyses, each removing a line from the base state.
- `'tsa'`: Transient stability analysis. PowerSAS.m assesses the system's dynamic behavior and stability after a given disturbance(s). In this current version, PowerSAS.m supports simulation with three-phase balanced fault(s), which are the most common disturbances. PowerSAS supports the analysis of the combinations of multiple faults happening simultaneously or sequentially.

B. EXTENSIBLE ANALYSIS SCENARIOS

PowerSAS.m has an event-driven simulation scheme where users can use some atomic events to build a customized simulation scenario.

The atomic events include applying/clearing faults and adding/tripping power lines, synchronous generators, induction motors, and loads, among others. These atomic events can be arbitrarily arranged in a temporal sequence to form a customized scenario. PowerSAS.m will follow this scenario to conduct simulations. It is worth mentioning that multiple events happening at the same time are allowed.

Code Block 3 Example: PowerSAS.m DSA

```
% EXAMPLE: DSA during restoration with
PowerSAS.m

% PowerSAS.m event-driven simulation
res=runPowerSAS('dyn','d_014_mod.m',[],...
'event');

% plot results
plotCurves(1,res.t,res.stateCurve,...
res.SysDataBase,'v');
```

By making use of this functionality, users can apply PowerSAS.m to more sophisticated analysis scenarios. For example, one can use the tool to conduct dynamic security assessment (DSA) during grid restoration. As shown in Code Block 3, the API is similar to that shown in Code Block 2. The major difference is 1) we use `'dyn'` in the input argument instead of the five basic functions, and 2) we need to specify the event information, such as the event list. An example event list is given in Code Block 4, where we specify four events (adding a line, a synchronous generator, an induction motor, and a static load, respectively). Note that we use a matrix `eventList` to streamline the basic scenario: the first column corresponds to the sequence of the events; the second and third columns represent the event start and end time (note that for a switching event, the start and end time is the same); the fourth column specifies the event type (for example, `0` stands for simulation start, and `1` stands for adding line); and the fifth column points to the events' specifications (for example, in the second row of `eventList`, `1` is an index about which line is to add to the system). In addition to the event list, the `'dyn'` functionality can specify a variety of other simulation details. These settings are omitted here due to the paucity of space. Interested readers are referred to a detailed example, `ex_restoration.m`, that is included in the software package .[3]

Code Block 4 Example: DSA Event List

```
% EXAMPLE: Event list during restoration

eventList=[...
% -simulation starts-
1 0.0000 0.0000 0 0;
% -dynamic simulations-
2 0.0000 0.5000 50 0;
% -add line at time 0.5s-
3 0.5000 0.5000 1 1;
% -dynamic simulations-
4 0.5000 1.5000 50 0;
% -add synchronous generator at time 1.5s-
5 1.5000 1.5000 4 1;
% -dynamic simulations-
6 1.5000 4.5000 50 0;
% -simulation ends-
7 4.5000 4.5000 99 0;
];
```

VI. CASE STUDIES

In this section, we demonstrate the computational performance of PowerSAS.m. In comparison to existing methods, we show the competitive numerical robustness, simulation accuracy, and computational efficiency of PowerSAS.m.

A. IEEE 39-BUS SYSTEM: STATIC SECURITY REGION APPROXIMATION

Static security region (SSR) is important for power system monitoring and operational decision-making. An SSR is often a set in the parameter space of the steady-state power system model. Each point lying in the SSR specifies a combination of parameters and entails an operating scenario that corresponds to a stable operating point. Since analytical methods to characterize an SSR is generally unavailable, numerical approximation methods are often employed. Convergence issues usually emerge during the SSR approximation computations,

[3]The example `ex_restoration.m` is available at: "https://github.com/ANL-CEEESA/powersas.m/tree/release/example"

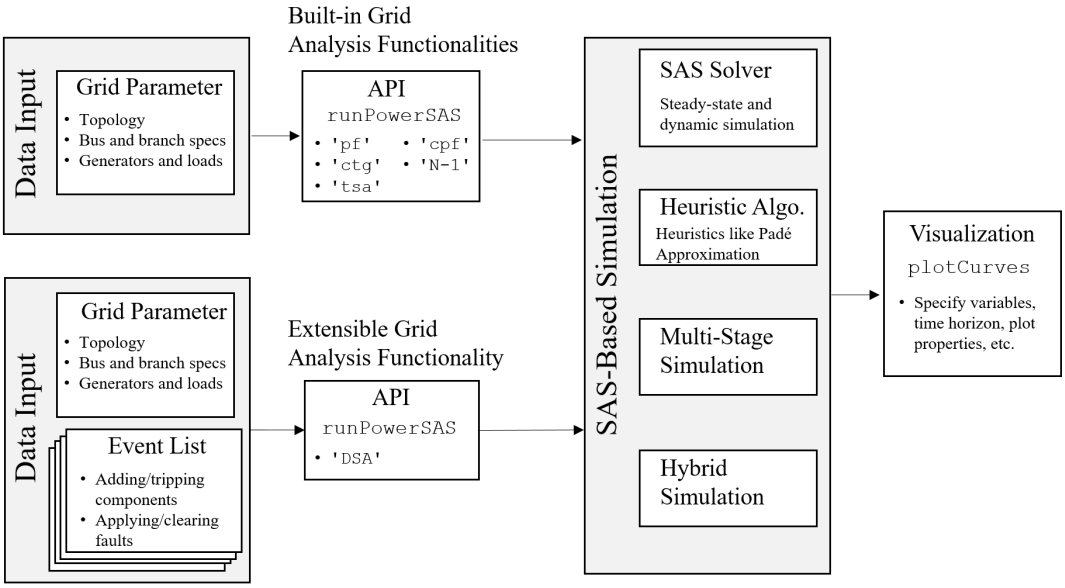


FIGURE 1. Illustration of PowerSAS.m's fundamental design and main functionalities.

especially when the scenario under study is near the SSR boundary.

Based on the above discussion, approximating the SSR of a power system is suitable for testing the numerical robustness of a computational method. We approximate the SSR of a modified IEEE 39-bus system. The nonlinear power flow equations determine the system's steady-state operating point. We sample the parameter space and use numerical methods to examine whether a sample corresponds to a stable operating point. The sample-examination tasks can be performed by PowerSAS.m using the steady-state power system simulation functionalities. In this case study, we focus on the power flow equations and utilize the power flow analysis functionality by calling `'pf'` in `runPowerSAS`. The simulation results are compared with PSS/E version 33, which is the benchmark solver. PSS/E version 33 is widely used for power system analysis, which is suitable for benchmarking purposes.

We approximate the SSR of the IEEE-39 system, supposing that the active power injection of Buses 3 and 4 have large variations. The active power is sampled uniformly over the interval of [-4000, 4000] MW. Fig. 2 and Fig. 3 show the SSR approximated by PSS/E and PowerSAS.m, respectively, where a blue dot represents a sample associated with a stable operating point, and a gray point represents the opposite. One can see that both tools identify a big oval-shaped SSR region. However, the PSS/E results find some outliers (about 0.1% of the samples) scattered around the oval to be part of the SSR as well. The PowerSAS.m results do not agree with the PSS/E's regarding these points. Through numerical testing, these outliers lead to unstable solutions and are thus not part of the SSR. Compared to this benchmark, PowerSAS.m can more accurately identify the SSR due to its strong numerical robustness.

B. REDUCED EI 458-BUS SYSTEM: N-25 CONTINGENCY ANALYSIS

With the event-driven simulation scheme, PowerSAS.m can perform various power system analysis tasks aside from the built-in functionalities. Here, we show a case study about N-25 contingency analysis.

In general, contingency analysis also has convergence challenges, mainly caused by large disturbances. Here we compare PowerSAS.m with PSS/E in an N-25 contingency analysis for a reduced eastern interconnection (EI) system with 458 buses. We increase the load and generation level in three testing cases by 15%, 20%, and 20.7%, respectively. In each case, we randomly choose 5000 different N-25 contingency scenarios. Once the scenarios are sampled, one only needs to update the event list enclosed in simulation settings to use PowerSAS.m to conduct the testing.

The simulation results are shown in Fig. 4. The benchmark results encompass cases where the PSS/E "non-divergence" (ND) functionality is turned on and off. It can be observed if ND is turned on, the PSS/E convergence result will significantly improve. Nevertheless, the PowerSAS.m still outperforms the benchmarks in terms of numerical robustness.

Then, we compare the computation efficiency between PowerSAS.m and PSS/E. Fig. 5 shows the average contingency analysis computation time in the N-25 contingency analysis. The results show that PowerSAS.m's computational speed is comparable to and even faster than that of PSS/E.

C. POLISH 2383-BUS SYSTEM: TRANSIENT STABILITY ANALYSIS

We then study the transient stability of a modified Polish 2383-bus system using PowerSAS.m. We compare the computational performance of the proposed SAS methods

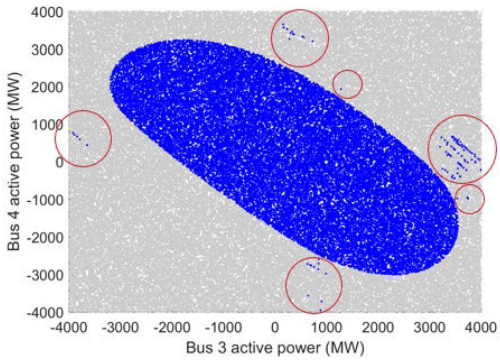


FIGURE 2. SSR characterized by PSS/E.

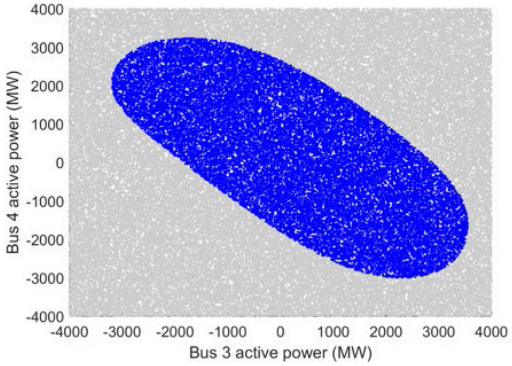


FIGURE 3. SSR characterized by PowerSAS.m.

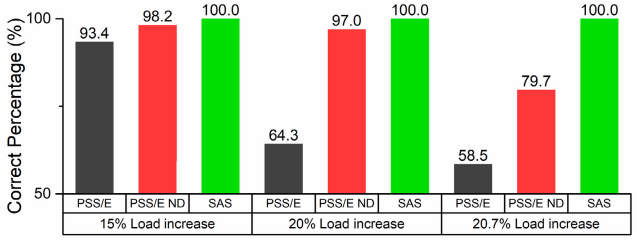


FIGURE 4. Comparison of PowerSAS.m and PSS/E in terms of convergence rates.

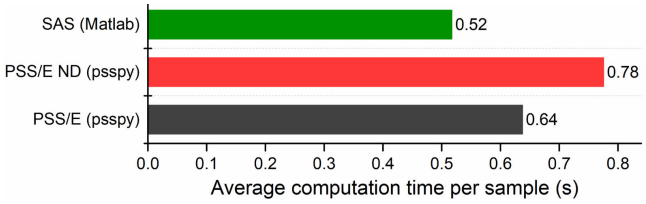


FIGURE 5. Comparison of PowerSAS.m and PSS/E in terms of computational speed.

against traditional methods with numerical integration (NI) and Newton-Raphson (NR) methods.

We include 1827 ZIP loads, 327 dynamical synchronous generator models, and 1542 dynamical induction motor models in the test case. Due to the page limit, the system specification is omitted here. Interested readers can find them in detail from a data file in our GitHub repository .[4]

[4]The file is named "d_2383wp_mod2_ind_zip_syn.m" and is available at: "https://github.com/ANL-CEEESA/powersas.m/tree/release/data"

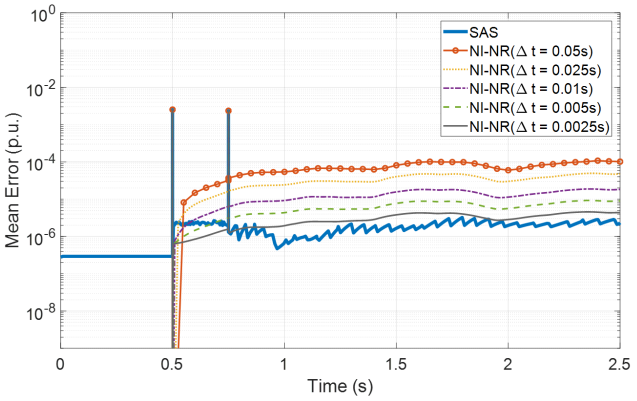


FIGURE 6. Comparison between SAS and numerical solvers regarding simulation accuracy.

TABLE 2. Comparison between SAS and numerical solvers regarding simulation speed.

Method	Simulation Time (s)
SAS	7.15
NI-NR ($\Delta t = 0.05$)	8.24
NI-NR ($\Delta t = 0.025$)	19.15
NI-NR ($\Delta t = 0.01$)	39.70
NI-NR ($\Delta t = 0.005$)	76.65
NI-NR ($\Delta t = 0.0025$)	149.46
NI-NR ($\Delta t = 0.0001$)	2088.66

We create a test scenario using the event-driven simulation scheme where two three-phase grounding faults happened to Lines 42-41 and 540-23. The fault event starts from 0.5 s and clears at 0.75 s.

We first use the NI-NR method with a small time-step of 0.0001 s to simulate the case. The simulation result is considered the "ground truth" (GT). Then, we use the NI-NR method with different time steps and the SAS method to simulate the case. The simulation results are compared with the GT to find the mean simulation error, shown in Fig. 6. The simulation time of all these cases is shown in Table 2. Note that for the SAS methods, we specify the order to be 10.

From Fig. 6, one can observe that the SAS method generally achieves the highest level of simulation accuracy. Although the NI-NR case with a time step of 0.0025 s can achieve similar performance with SAS in terms of accuracy, the simulation time is over 20 times greater than the latter. Moreover, from Table 2, other than the NI-NR case with a time step of 0.05 s, the simulation time of the SAS method is significantly smaller despite the better simulation accuracy. Still, Fig. 6 shows that the NI-NR case with 0.05 s time step has a mean simulation error of approximately two magnitudes greater than that of the SAS method.

D. SYSTEMS WITH 458 TO 210k BUSES: POWER FLOW STUDY

To further demonstrate the scalability of PowerSAS.m, we apply the tool to solve the power flow problems of multiple

TABLE 3. PowerSAS.m scalability testing.

System Size	Simulation Time (s)
210k-bus	77.65
EI 70k-bus	24.84
21447-bus	4.87
Polish 2383-bus	2.35
EI 458-bus	0.52

power systems, including a synthetic 210k-bus system. The test system models are available on our GitHub page.[5] The computational time to obtain the power flow solution is reported in Table 3. It is worth mentioning that PSS/E can only simulate power systems with up to 200k buses.

It can be observed that PowerSAS.m is reasonably scalable and can perform some grid analysis tasks that PSS/E cannot support.

VII. CONCLUSION

This paper introduces a recently open-sourced power system simulation toolbox, PowerSAS.m. This toolbox implements novel semi-analytical simulation (SAS) technologies that have proven advantageous computational performance in both steady-state and dynamic simulations. In addition, various simulation techniques, including simulation heuristic algorithms and hybrid simulation schemes, are developed to improve the numerical robustness and computational efficiency further. Benchmarking case studies are presented in the paper to verify the validity and value of the toolbox. For future work, we will 1) continue investigating the fundamentals of SAS technologies, 2) keep maintaining and updating PowerSAS.m, and 3) develop SAS-based toolboxes in other programming languages.

REFERENCES

[1] S. K. Khaitan, J. D. McCalley, and Q. Chen, "Multifrontal solver for online power system time-domain simulation," *IEEE Trans. Power Syst.*, vol. 23, no. 4, pp. 1727–1737, Nov. 2008.

[2] S. Rao, Y. Feng, D. J. Tylavsky, and M. K. Subramanian, "The holomorphic embedding method applied to the power-flow problem," *IEEE Trans. Power Syst.*, vol. 31, no. 5, pp. 3816–3828, Sep. 2016.

[3] S. Jin, Z. Huang, R. Diao, D. Wu, and Y. Chen, "Comparative implementation of high performance computing for power system dynamic simulations," *IEEE Trans. Smart Grid*, vol. 8, no. 3, pp. 1387–1395, May 2017.

[4] S.-K. Joo, C.-C. Liu, L. E. Jones, and J.-W. Choe, "Coherency and aggregation techniques incorporating rotor and voltage dynamics," *IEEE Trans. Power Syst.*, vol. 19, no. 2, pp. 1068–1075, May 2004.

[5] S. K. Mazumder et al., "A review of current research trends in power-electronic innovations in cyber–physical systems," *IEEE J. Emerg. Sel. Topics Power Electron.*, vol. 9, no. 5, pp. 5146–5163, Oct. 2021.

[6] Z. Bie, Y. Lin, G. Li, and F. Li, "Battling the extreme: A study on the power system resilience," *Proc. IEEE*, vol. 105, no. 7, pp. 1253–1266, Jul. 2017.

[7] R. Zimmerman, C. E. Murillo-Sanchez, and R. J. Thomas, "MATPOWER: Steady-state operations, planning, and analysis tools for power systems research and education," *IEEE Trans. Power Syst.*, vol. 26, no. 1, pp. 12–19, Feb. 2010.

[8] P. Kundur, *Power System Stability and Control*. New York, NY, USA: McGraw-Hill, 1994.

[9] F. Milano, *Power System Modelling and Scripting*. Berlin, Germany: Springer, 2010.

[5]PowerSAS.m/data. Available: https://github.com/ANL-CEEESA/powersas.m/tree/release/data

[10] B. Stott, "Power system dynamic response calculations," *Proc. IEEE*, vol. 67, no. 2, pp. 219–241, Feb. 1979.

[11] R. Yao, Y. Liu, K. Sun, F. Qiu, and J. Wang, "Efficient and robust dynamic simulation of power systems with holomorphic embedding," *IEEE Trans. Power Syst.*, vol. 35, no. 2, pp. 938–949, Mar. 2020.

[12] Y. Liu and K. Sun, "Solving power system differential algebraic equations using differential transformation," *IEEE Trans. Power Syst.*, vol. 35, no. 3, pp. 2289–2299, May 2019.

[13] N. Duan and K. Sun, "Power system simulation using the multistage Adomian decomposition method," *IEEE Trans. Power Syst.*, vol. 32, no. 1, pp. 430–441, Jan. 2017.

[14] G. Gurrala, D. L. Dinesha, A. Dimitrovski, P. Sreekanth, S. Simunovic, and M. Starke, "Large multi-machine power system simulations using multi-stage Adomian decomposition," *IEEE Trans. Power Syst.*, vol. 32, no. 5, pp. 3594–3606, Sep. 2017.

[15] B. Park, K. Sun, A. Dimitrovski, Y. Liu, and S. Simunovic, "Examination of semi-analytical solution methods in the coarse operator of parareal algorithm for power system simulation," *IEEE Trans. Power Syst.*, vol. 36, no. 6, pp. 5068–5080, Nov. 2021.

[16] B. Wang, N. Duan, and K. Sun, "A time–power series-based semi-analytical approach for power system simulation," *IEEE Trans. Power Syst.*, vol. 34, no. 2, pp. 841–851, Mar. 2018.

[17] R. Yao, K. Sun, D. Shi, and X. Zhang, "Voltage stability analysis of power systems with induction motors based on holomorphic embedding," *IEEE Trans. Power Syst.*, vol. 34, no. 2, pp. 1278–1288, Mar. 2019.

[18] R. Yao, K. Sun, and F. Qiu, "Vectorized efficient computation of Padé approximation for semi-analytical simulation of large-scale power systems," *IEEE Trans. Power Syst.*, vol. 34, no. 5, pp. 3957–3959, Sep. 2019.

[19] R. Yao, F. Qiu, and K. Sun, "Contingency analysis based on partitioned and parallel holomorphic embedding," *IEEE Trans. Power Syst.*, vol. 37, no. 1, pp. 565–575, Jan. 2021.

[20] Y. Liu, K. Sun, R. Yao, and B. Wang, "Power system time domain simulation using a differential transformation method," *IEEE Trans. Power Syst.*, vol. 34, no. 5, pp. 3739–3748, Sep. 2019.

[21] Y. Liu, K. Sun, and J. Dong, "A dynamized power flow method based on differential transformation," *IEEE Access*, vol. 8, pp. 182441–182450, 2020.

[22] C. Liu, B. Wang, and K. Sun, "Fast power system simulation using semi-analytical solutions based on Pade approximants," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, Jul. 2017, pp. 1–5.

[23] C. Liu, B. Wang, and K. Sun, "Fast power system dynamic simulation using continued fractions," *IEEE Access*, vol. 6, pp. 62687–62698, 2018.

[24] D. L. Dinesha and G. Gurrala, "Application of multi-stage homotopy analysis method for power system dynamic simulations," *IEEE Trans. Power Syst.*, vol. 34, no. 3, pp. 2251–2260, May 2019.

[25] F. Qiu and P. Li, "An integrated approach for power system restoration planning," *Proc. IEEE*, vol. 105, no. 7, pp. 1234–1252, Jul. 2017.

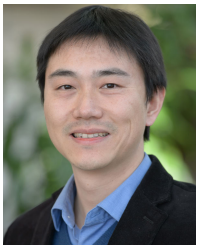
[26] Y. Zhang et al., "Encoding frequency constraints in preventive unit commitment using deep learning with region-of-interest active sampling," *IEEE Trans. Power Syst.*, vol. 37, no. 3, pp. 1942–1955, May 2021.

[27] J. Liu, B. Cui, D. K. Molzahn, C. Chen, X. Lu, and F. Qiu, "Optimal power flow in DC networks with robust feasibility and stability guarantees," *IEEE Trans. Control Netw. Syst.*, vol. 9, no. 2, pp. 904–916, Jun. 2019.

JIANZHE LIU (Member, IEEE) received the B.E. degree in electrical engineering from the Huazhong University of Science and Technology, China, in 2012, and the Ph.D. degree in electrical and computer engineering from The Ohio State University, USA, in 2017. He was a Visiting Scholar with the Department of Energy Technology, Aalborg University, Denmark, in 2017.

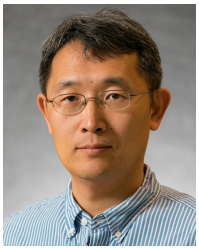
He is currently an Energy Systems Scientist with the Argonne National Laboratory. He also holds a joint appointment position with The University of Chicago, working as a Researcher and an independent PI. His research interests include power system dynamics analysis and operations, including stability analysis and control of power systems with high penetrations of renewable energy and power electronics devices, grid optimization with system dynamics constraints, and cyber-physical resilient operations. He is also an Associate Editor of the IEEE Open Access Journal of Power and Energy (OAJPE) and *Power Engineering Letters* (PES Letters).

RUI YAO (Senior Member, IEEE) received the B.S. (Hons.) and Ph.D. degrees in electrical engineering from Tsinghua University, Beijing, China, in 2011 and 2016, respectively. From 2016 to 2018, he was a Post-Doctoral Research Associate with the University of Tennessee, Knoxville, TN, USA. From 2018 to 2022, he was with the Argonne National Laboratory, Lemont, IL, USA. Currently, he is a Senior Engineer with Google LLC. His research interests include power system modeling and stability analysis, resilience modeling and assessment, and high-performance computational methodologies. He is also an Editor of the IEEE TRANSACTIONS ON POWER SYSTEMS, IEEE POWER ENGINEERING LETTERS, and *International Transactions on Electrical Energy Systems*.



FENG QIU (Senior Member, IEEE) received the Ph.D. degree from the School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, USA, in 2013. He is currently a Principal Computational Scientist and the Section Leader of the Argonne National Laboratory, Energy Systems Division. His research interests include power system modeling and optimization, electricity markets, power grid resilience, machine learning, and data analytics.

YANG LIU (Member, IEEE) received the B.S. degree in energy and power engineering from Xi'an Jiaotong University, China, in 2013, the M.S. degree in power engineering from Tsinghua University, China, in 2016, and the Ph.D. degree in electrical engineering from the University of Tennessee, Knoxville, USA, in 2021. He was a Post-Doctoral Research Associate with the University of Tennessee, in 2022. Currently, he is a Post-Doctoral Appointee with the Argonne National Laboratory. His research interests include power system simulation, dynamics, stability, and control.



KAI SUN (Senior Member, IEEE) received the B.S. degree in automation and the Ph.D. degree in control science and engineering from Tsinghua University, Beijing, China, in 1999 and 2004, respectively. He was the Project Manager of Grid Operations and Planning with the Electric Power Research Institute (EPRI), Palo Alto, CA, USA, from 2007 to 2012. He is currently a Professor with the Department of EECS, University of Tennessee, Knoxville, USA. He is also an Associate Editor of IEEE TRANSACTIONS ON POWER SYSTEMS and IEEE OPEN ACCESS JOURNAL OF POWER AND ENERGY.