

# A Frequency Domain Approach to Predict Power System Transients

Wenqi Cui , Graduate Student Member, IEEE, Weiwei Yang , and Baosen Zhang , Member, IEEE

**Abstract**—The dynamics of power grids are governed by a large number of nonlinear differential and algebraic equations (DAEs). To safely operate the system, operators need to check that the states described by these DAEs stay within prescribed limits after various potential faults. However, current numerical solvers of DAEs are often too slow for real-time system operations. In addition, detailed system parameters are often not exactly known. Machine learning approaches have been proposed to reduce the computational efforts, but existing methods generally suffer from overfitting and failures to predict unstable behaviors. This paper proposes a novel framework to predict power system transients by learning in the frequency domain. The intuition is that although the system behavior is complex in the time domain, there are relatively few dominant modes in the frequency domain. Therefore, we learn to predict by constructing neural networks with Fourier transform and filtering layers. System topology and fault information are encoded by taking a multi-dimensional Fourier transform, allowing us to leverage the fact that the trajectories are sparse both in time and spatial frequencies. We show that the proposed approach does not need detailed system parameters, greatly speeds up prediction computations and is highly accurate for different fault types.

**Index Terms**—Power system transients, prediction methods, learning systems, frequency domain analysis.

## I. INTRODUCTION

INCREASING the amount of renewable resources integrated in the electric grid is fundamental to reducing carbon emissions and mitigating climate change. Many governments and companies have set ambitious goals to generate their electricity with close to 100% renewables by 2050 [1]. So far, much of the attention has been paid to increasing the aggregate generation capacities. However, increased renewable generation capacities also lead to challenging problems in dynamic stability of the grid [2]. A grid can be thought as a large interconnected system of generators, loads and power electronic components, governed by nonlinear differential and algebraic equations (DAEs) [3], [4]. This system also undergoes constant disturbances, from load

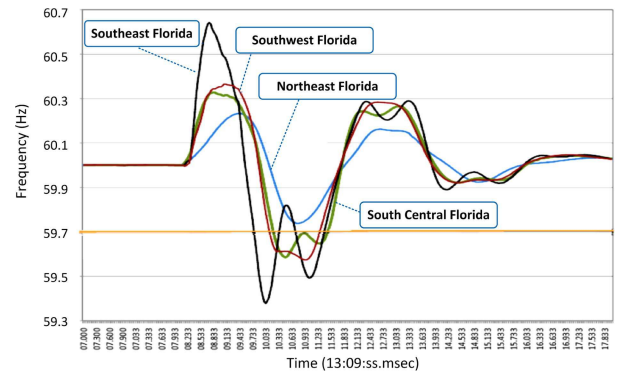


Fig. 1. Frequency oscillations during the Florida blackout resulted from a tripped line event in 2008 [6].

changes to line outages [5]. The predominant goal of power system operations is to make sure that the system stays within acceptable limits under these disturbances.

The system states are governed by different DAEs during the transient across pre-fault, fault-on and post-fault stages [4]. Fig. 1 shows a tripped line in Florida that led to rolling blackouts that impacted the lower two-thirds of state [6]. Ideally, a system should withstand these types of single events without performance degradation ( $N - 1$  security) [5], but this was not the case in Fig. 1. A fundamental reason is that solving the governing DAEs is extremely computationally challenging and not all contingencies can be checked. These DAEs are highly nonlinear and numerical methods (e.g., implicit and explicit integration) are used to solve them [7]. To study transient stability, numerical algorithms need to use fairly fine discretizations. Even for a moderately sized system, existing numerical solvers may take minutes to simulate only seconds of system trajectories [8].

Consequently, only a limited number of scenarios are studied offline and operators tend to restrict the system to operate close to these scenarios. As shown in Fig. 2, for certain given conditions of power generation, demand and topology, system operators simulate the system trajectories after disturbances through solving the shifted DAEs. By studying scenarios of system conditions offline, system operators will have a checklist about what action needs to be taken in real time to ensure that the systems' states are within the permissible range after critical contingencies. Since renewables have much larger uncertainties than conventional resources, operators often curtail them to artificially limit their generation to avoid operating at "unknown" regions [3]. For example, some European grids are not allowed to

Manuscript received 19 May 2022; revised 5 October 2022 and 30 January 2023; accepted 5 March 2023. Date of publication 21 March 2023; date of current version 26 December 2023. This work was supported in part by the Climate Change AI and in part by the National Science Foundation under Award ECCS-1942326. Paper no. TPWRS-00654-2022. (Corresponding author: Wenqi Cui.)

Wenqi Cui and Baosen Zhang are with the Department of Electrical and Computer Engineering, University of Washington, Seattle, WA 98195 USA (e-mail: wenqicui@uw.edu; zhangbao@uw.edu).

Weiwei Yang is with the Microsoft Research, Redmond, WA 98052 USA (e-mail: weiywa@microsoft.com).

Color versions of one or more figures in this article are available at <https://doi.org/10.1109/TPWRS.2023.3259960>.

Digital Object Identifier 10.1109/TPWRS.2023.3259960

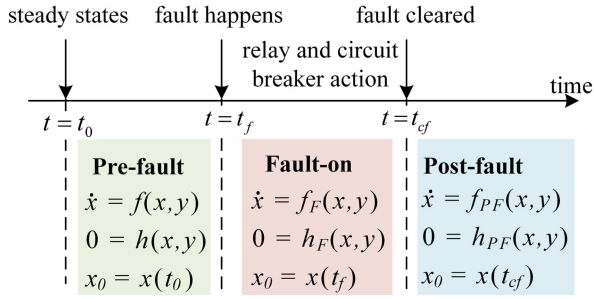


Fig. 2. Power system operators conduct dynamic power system transient prediction by solving DAEs [4], [10], where  $x$  and  $y$  represent the set of state variables and algebraic variables, respectively. The dynamics are described by differential equations  $f(\cdot)$  and algebraic equations  $h(\cdot)$ , with subscript  $F$  and  $PF$  indicating the fault-on and post-fault period divided by the fault time  $t_f$  and the fault-clear time  $t_{cf}$ , respectively. Under certain conditions of power generation, demand and topology, the system trajectory after disturbances is simulated through solving the shifted DAEs.

operate at above 40% wind, no matter how much wind is actually blowing [9]. Therefore, fast and accurate dynamic simulations would greatly increase the actual utilization of renewables in the system.

Recently, machine learning (ML) approaches have been proposed for dynamical simulation instead of DAE solvers and can reduce the computation time by orders of magnitude. Given the fault information and present measurement of states, ML methods learn feed-forward neural networks to predict future trajectories. Most works focus on a binary classification to identify whether a system is stable. Long short-term memory-based recurrent neural networks are commonly utilized to process sequential data [11], [12]. A shaplet learning approach is proposed to extract spatial-temporal correlations [13]. Convolutional neural networks are adopted in [14], [15] to process data from multiple sources. However, a binary prediction may not provide sufficient information for real-time decision making. For example, to prevent cascading outages, operators need to know the magnitude of the states and the actual trajectories [4], but binary predictions are too coarse for making these types of assessments.

To alleviate the limitation of binary prediction, some recent works provide finer trajectory prediction by learning the time domain solutions to the governing DAEs. Polynomial basis are used in [16] to approximate the solutions, but the number of basis functions grows exponentially with the system size. Extreme learning machine is utilized in [17] for online voltage stability margin prediction, but the traces of the system variables are not provided. Deep neural networks are used in [18], [19] to directly learn to predict the future trajectory from past and current measurements. However, since power systems are large and sampled sparsely in time, direct regression on time-domain data does not perform very well. A rolling prediction is used in [18] to improve accuracy, but with a high computational cost. Instead of the time-domain approach, eigenstates of linear swing equations after eigenspace transformation are utilized in [20] to infer system dynamics. However, the assumptions on the linearized system model and uniform damping ratios may not hold for realistic and large-scale systems.

In addition, since the majority of trajectories used in training are stable, the learned networks may fail to predict unstable behaviors. Physics-informed neural networks that directly attempt to solve the DAEs have been proposed as an alternative [21], but it does not currently scale beyond small networks.

In this paper, we propose a novel framework for predicting power system transients by learning and making predictions in the frequency domain, which provides a computation speed up of more than 400 times compared to existing power system tools. This approach follows the intuition that the system tends to undergo oscillations that have a few dominant temporal-spatial modes. We adopt and extend the structure of Fourier Neural Operator in [22] to learn in the frequency domain and recover the time domain trajectories through the inverse Fourier transform. Specifically, we design the dataframe to encode the power system topology and fault information, which leads to a 3D Fourier transform. This method is able to make smooth and accurate predictions, capturing both stable and unstable behaviors without the need to manually tune the training data. It improves the MSE prediction error by more than 70% compared with state-of-the-art AI methods, and vastly improves the detection of unstable behavior. Code and data are available at <https://github.com/Wenqi-Cui/Predict-Power-System-Dynamics-Frequency-Domain>.

In summary, the main contributions of the paper are:

- 1) We propose a novel machine learning approach to predict transient dynamics in the frequency domain, which can accurately predict state trajectories based on a few measurements.
- 2) We develop a dataframe that encodes spatial-temporal information about the system topology, which greatly reduces the computational complexity in multi-dimensional Fourier transforms.
- 3) The time-varying active/reactive power injection and fault-on/clear actions are incorporated in the proposed framework, enabling the prediction of the transients subject to different net power injections and actions.

The remaining of this paper is organized as follows. Section II introduces the problem formulation for predicting power system dynamics and transients. Section III provides the setup and intuition of learning in the frequency domain. Section IV shows the proposed framework for dynamic transient prediction and Section V illustrates the construction of dataframe to encode spatial-temporal relationships. Section VI shows the simulation results. Section VII concludes the paper.

## II. MODEL AND PROBLEM FORMULATION

### A. Power System Swing Equations

The dynamics of power systems depend on the interactions of a myriad of components including governors, exciters, stabilizers, etc., as illustrated in Fig. 3 [5]. Let  $\mathbf{x} \in \mathbb{R}^n$ ,  $\mathbf{y} \in \mathbb{R}^m$ ,  $\mathbf{a} \in \mathbb{R}^d$  be all the state variables, algebraic variables and external input variables, respectively.<sup>1</sup> The complete power system

<sup>1</sup>Throughout this paper, vectors are denoted in lower case bold and matrices are denoted in upper case bold, while scalars are unbolded. If not specified, all the vectors are time-varying. The vectors followed by  $(t)$  denote the value at the time  $t$ .

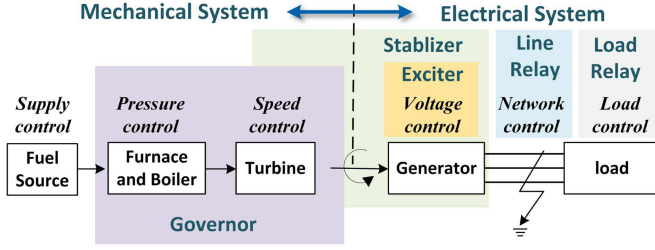


Fig. 3. Different components contributing to power system transient dynamics [5].

model for calculating system dynamic response relative to a disturbance can be described by a set of DAEs as follows [4], [16]:

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{y}, \mathbf{a}) \\ \mathbf{0} = \mathbf{h}(\mathbf{x}, \mathbf{y}, \mathbf{a}) \end{cases} \quad (1)$$

where the differential equation  $\mathbf{f} : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^d \rightarrow \mathbb{R}^n$  typically describes the internal dynamics of devices such as the speed and angle of generator rotors, the response of generator control systems (e.g., excitation systems, turbines, governors), the dynamics of equipment including DC lines, dynamically modeled loads and their control systems. Correspondingly,  $\mathbf{x} \in \mathbb{R}^n$  is the state variables such as generator rotor angles, generator velocity deviations (speeds), electromagnetic flux, various control system internal variables, etc. The set of algebraic equations  $\mathbf{h} : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^d \rightarrow \mathbb{R}^m$  describes the electrical transmission system and interface equations. Correspondingly,  $\mathbf{y} \in \mathbb{R}^m$  is the algebraic variable such as voltage magnitude and angles.

The external input variables  $\mathbf{a} \in \mathbb{R}^d$  acting on the equations are power injection from generators, automatic generation control systems, fault-response actions and so on [4], [23]. In this paper, we mainly consider the uncertainties in power generation and demand, as well as the fault-response actions  $\mathbf{u}$ . Let the active and reactive net power injection be  $\mathbf{p}$  and  $\mathbf{q}$ , then the external input variables are sometimes written as the tuple  $\mathbf{a} = (\mathbf{p}, \mathbf{q}, \mathbf{u})$ .

Under changing generator and load conditions, power systems are operated to withstand the occurrence of certain contingencies. To ensure that cascading outages will not occur for the set of critical disturbances, the state variables need to stay within permissible ranges during the transient process of power system after disturbances [4], [24]. For even a moderate power system with tens or hundreds of buses, it may be governed by hundreds or thousands of DAEs.

## B. Transient Dynamics

Disturbances lead to deviations in the states  $\mathbf{x}$  and variables  $\mathbf{y}$  through a step change of parameters in (1). For example, a short circuit on a transmission line will result in a sudden change of the susceptance and conductance in the set of algebraic equations, depending on the specific fault types (e.g., single-phase-to-ground, two-phase-to-ground, line-to-line, three-phase-to-ground, etc.).

Suppose a fault happens at the time  $t_f$  and is cleared at the time  $t_{cl}$ .<sup>2</sup> The pre-fault stage is defined as the period before the fault happens at  $t_f$ . The system evolves from the initial state  $\mathbf{x}(t_0)$  as:

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{y}, \mathbf{a}; \mathbf{x}(t_0)) \\ \mathbf{0} = \mathbf{h}(\mathbf{x}, \mathbf{y}, \mathbf{a}) \end{cases}, \quad t_0 \leq t < t_f \quad (2)$$

The sudden parameter changes after disturbances will lead to a shift of the swing equations. The fault-on system evolves with  $k$  subsequent actions from system relays and circuit breakers. Suppose the  $j$ -th action is taken at  $t_{F,j}$ , the fault-on system is described by several set of equations [4]

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{f}_{F,1}(\mathbf{x}, \mathbf{y}, \mathbf{a}; \mathbf{x}(t_f)) \\ \mathbf{0} = \mathbf{h}_{F,1}(\mathbf{x}, \mathbf{y}, \mathbf{a}) \end{cases}, \quad t_f \leq t < t_{F,1} \\ \vdots \\ \begin{cases} \dot{\mathbf{x}} = \mathbf{f}_{F,k}(\mathbf{x}, \mathbf{y}, \mathbf{a}; \mathbf{x}(t_{F,k})) \\ \mathbf{0} = \mathbf{h}_{F,k}(\mathbf{x}, \mathbf{y}, \mathbf{a}) \end{cases}, \quad t_{F,k-1} \leq t < t_{cl} \end{cases} \quad (3)$$

The post-fault stage refers to the system after the fault is cleared. The post-fault system evolves with the differential equation starting from the post-fault initial state  $\mathbf{x}(t_{cl})$ , written as [4]

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{f}_{PF}(\mathbf{x}, \mathbf{y}, \mathbf{a}; \mathbf{x}(t_{cl})) \\ \mathbf{0} = \mathbf{h}_{PF}(\mathbf{x}, \mathbf{y}, \mathbf{a}) \end{cases}, \quad t \geq t_{cl}. \quad (4)$$

Despite large numbers of variables  $\mathbf{x}$  and  $\mathbf{y}$ , not all of them are observable to system operators. For a system with  $N$  buses, typically the main variables of interest are the angle  $\delta_i$ , rotor angle speeds (frequency) deviation  $\omega_i$  and voltage  $v_i$  at each bus  $i$ . Denote  $[N] := \{1, \dots, N\}$ ,  $\boldsymbol{\delta} := (\delta_i, i \in [N]) \in \mathbb{R}^N$ ,  $\boldsymbol{\omega} := (\omega_i, i \in [N]) \in \mathbb{R}^N$ ,  $\mathbf{v} := (v_i, i \in [N]) \in \mathbb{R}^N$ . These variables of interest are described by a three-tuple, denoted by  $\mathbf{s} = (\boldsymbol{\delta}, \boldsymbol{\omega}, \mathbf{v}) \in \mathbb{R}^{3N}$ . Note that other variables can also be included in  $\mathbf{s}$  if they are observable. The key to the safe dynamic operation of power systems is to predict the future of the system trajectories, given the fault information, some observations  $\mathbf{s}$  and the expected clearing actions  $\mathbf{u}$ . Based on these trajectories, interim actions like load shedding or emergency generation can be taken to reduce the impact of the faults [5], [7].

## C. Current Approaches and Challenges

Current approaches in power system dynamic prediction are based on solving (2)–(4), which are highly nonlinear equations. System operators typically rely on numerical integration, such as Runge-Kutta (RK) methods or trapezoidal rule, to iteratively approximate the solution of (2)–(4) in small time intervals [7]. However, because of the highly nonlinear nature of the DAEs, very fine discretization steps are required for these numerical methods. As a result, these approaches may be too slow for real-time decision-making. Some solvers use reduced-order models and convert DAEs to ordinary differential equations (ODEs) to simulate the dynamic response of generators [25]. For a moderately sized system, existing numerical solvers take

<sup>2</sup>The disturbance such as a short circuit on a transmission line is automatically cleared by protective relay operation after a certain amount of time.

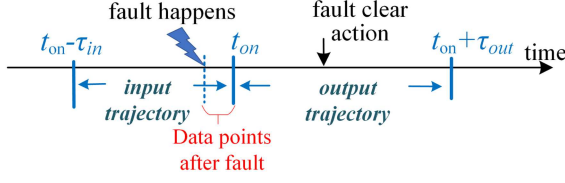


Fig. 4. Illustration of the trajectory prediction starting at the time stamp  $t_{on}$  for  $\tau_{out}$  number of time steps, using  $\tau_{in}$  observations.

minutes to simulate only seconds of system trajectories [8]. As an alternative, system operators also use manual heuristics to take action, but this strategy is becoming less tenable as renewables introduce distinctly different operating scenarios.

### III. LEARNING FOR DYNAMIC TRANSIENT PREDICTIONS

#### A. Problem Setup

Learning-based approaches try to find a mapping from present measurements to future trajectories. The predictions are then obtained through function evaluations, which significantly reduce the computational time compared to the conventional numerical approaches.

The problem we are interested in is to predict the trajectory of the states  $s$  starting at the time stamp  $t_{on}$  for  $\tau_{out}$  number of time steps with the sampling interval  $\Delta t$ , as illustrated in Fig. 4. The input are  $\tau_{in}$  observations of the states from  $s(t_{on} - \tau_{in})$  to  $s(t_{on} - 1)$ , the external inputs from  $a(t_{on} - \tau_{in})$  to  $a(t_{on} - 1)$ , and the expected fault-clear actions from  $u(t_{on})$  to  $u(t_{on} + \tau_{out} - 1)$ . We write  $s_{in} = (s(t_{on} - \tau_{in}), \dots, s(t_{on} - 1))$ ,  $a = (a(t_{on} - \tau_{in}), \dots, a(t_{on} - 1))$ ,  $u_{out} = (u(t_{on}), \dots, u(t_{on} + \tau_{out} - 1))$  and  $s_{out} = (s(t_{on}), \dots, s(t_{on} + \tau_{out} - 1))$ .

Our goal is to find a mapping  $G$  from the space of input  $(s_{in}, a, u_{out})$  to output trajectories  $s_{out}$ . In this paper we consider the mapping realized through (deep) neural networks with parameters  $\Phi$ . The prediction is then given by

$$\hat{s}_{out} = G_{\Phi}(s_{in}, a, u_{out}). \quad (5)$$

The exact form of  $\Phi$  depends on the structure of the neural network used. In this paper, we adopt the Fourier neural operator to learn in the frequency domain, and details will be specified later in Section IV.

Let  $H$  be the batch size and  $\hat{s}_{out}^i$  be the prediction of the  $i$ -th sample for  $i = 1, \dots, H$ . The weights of neural network  $\Phi$  are updated by back-propagation to minimize loss function  $L(\Phi)$  defined by the mean absolute percentage error (MAPE) between predicted trajectory and the actual trajectory  $L(\Phi) = \frac{1}{H} \sum_{i=1}^H \frac{\|\hat{s}_{out}^i - s_{out}^i\|_1}{\|s_{out}^i\|_1}$  [22].

#### B. Current ML Approaches and Limitations

Learning the power system transient dynamics is not trivial because states undergo nonlinear oscillations. Predictions with three existing approaches are illustrated in Fig. 5. The blue line is the trajectory of the frequency deviation on a bus before and after a fault. The blue squares are the true trajectory sampled at discrete times and the grey area is the prediction horizon.

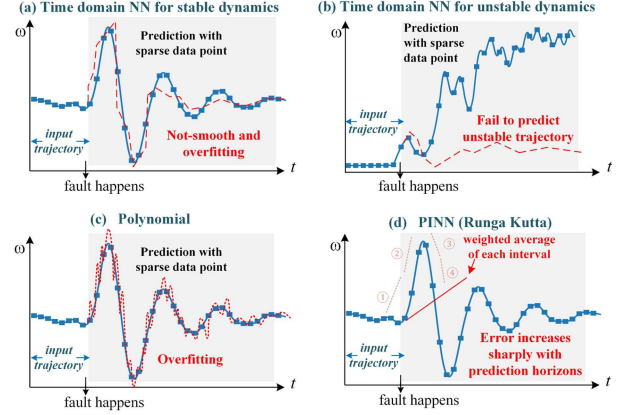


Fig. 5. Illustration of power system transient prediction with existing machine learning approaches. (a) Purely learning in the time domain tends to overfit. (b) Since the vast majority of historical data are stable, a ML method tends to not predict unstable trajectories. (c) Fitting the nonlinear dynamics with polynomial basis also easily leads to over-fitting. (d) The accuracy of PINN decreases sharply and fails to provide meaningful results for longer horizons.

A standard approach is to use a neural network to learn the time-domain mapping from the input to the output. As illustrated in Fig. 5(a) and Fig. 5(b), purely learning in the time domain will easily overfit and cannot learn a smooth curve like the true trajectories. More importantly, generic machine learning approaches are prone to false negative errors. Since the vast majority of historical trajectories are stable, a ML method tends to not predict unstable trajectories. This would lead to catastrophic consequences if the system operator does not take action to mitigate instabilities.

Similarly, fitting the nonlinear dynamics with polynomial basis will also easily lead to over-fitting, as illustrated in Fig. 5(c). Recently, Physics-Informed Neural Networks have been proposed in [26] to learn solutions that satisfy equations from implicit Runge-Kutta (RK) integration. This approach has been applied to power system swing dynamics in [21]. Since RK method is the weighted sum of ODE solutions in discretized intervals, its accuracy decreases sharply when predicting trajectories with large oscillations for a longer horizon (e.g., larger than 1 s), as illustrated in Fig. 5(d) (the prediction errors are larger than the limit of the y-axis).

#### C. The Proposed Approach: System Dynamics in the Frequency Domain

Because of the above challenges when learning in the time domain, we propose a new approach for learning power system transient dynamics in the frequency domain. Here we use a simple swing equation model for transient dynamics to illustrate the intuition for this approach [7], [27]:

$$\dot{\delta}_i = \omega_i \quad (6a)$$

$$M_i \dot{\omega}_i = p_i - D_i \omega_i - \sum_{j=1, j \neq i}^N B_{ij} (\delta_i - \delta_j) \quad (6b)$$

where  $i \in [N] := \{1, \dots, N\}$  is the index of buses,  $M := \text{diag}(M_i, i \in [N]) \in \mathbb{R}^{N \times N}$  are the generator inertia constants,

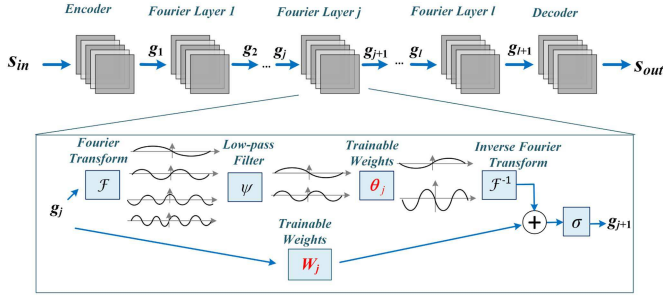


Fig. 6. The structure of neural networks for power system transient prediction using frequency-domain information. The input trajectory is first encoded using the framework introduced later in Section V. The encoded data is then passed through several Fourier Layers [22], where the input and output of the  $j$ -th layer is  $g_j$  and  $g_{j+1}$ , respectively. The  $j$ -th layer consists of trainable weights  $\theta_j$  for learning in the Fourier domain, and trainable weights  $W_j$  to keep the track of aperiodic and distorted waveform.

$D := \text{diag}(D_i, i \in [N]) \in \mathbb{R}^{N \times N}$  are damping coefficients,  $p := (p_i, i \in [N]) \in \mathbb{R}^N$  are the net power injections,  $B := [B_{ij}] \in \mathbb{R}^{N \times N}$  is the susceptance matrix.

With the assumption that the inertia and damping of the buses are proportional to their power ratings (i.e.,  $D_i/M_i = \gamma$  for all  $i \in [N]$ ), (6) can be explicitly solved [27]. Let  $C$  be the incidence matrix. The (scaled) graph Laplacian matrix is  $\Gamma = M^{-1/2} C B C^T M^{-1/2}$ , where  $0 = \lambda_1 < \lambda_2 \leq \dots \leq \lambda_N$  are the eigenvalues with corresponding orthonormal eigenvectors  $r_1, r_2, \dots, r_N$ . Suppose there is a step change  $\Delta p$  in the net power injection, and its decomposition along the eigenvectors is  $\Delta p = \sum_i \hat{p}_i M^{1/2} r_i$ . Then, the solution of equations (6) is [27]

$$\omega(t) = \sum_{i=1}^N \frac{\hat{p}_i}{\sqrt{\gamma^2 - 4\lambda_i}} (e^{\phi_{i,+} t} - e^{\phi_{i,-} t}) M^{-1/2} r_i \quad (7)$$

where  $\phi_{i,+} := \frac{-\gamma + \sqrt{\gamma^2 - 4\lambda_i}}{2}$ ,  $\phi_{i,-} := \frac{-\gamma - \sqrt{\gamma^2 - 4\lambda_i}}{2}$ .

Note that  $\phi_{i,+}$  are complex numbers with non-zero imaginary part if  $\gamma^2 - 4\lambda_i < 0$ , which results in sinusoidal oscillations. Thus, the sinusoidal basis in Fourier transform (and inverse Fourier transform) is a natural fit for power system transient dynamics. Because of the finite set of eigenvalues, there is a finite number of modes in the frequency domain. As a result, the trajectories are sparse in the frequency domain, making it easier to learn after Fourier transform.

However, the analysis based on the linear model (6) cannot be applied to more realistic systems as illustrated in Fig. 3, where high-order nonlinear differential equations are involved. This is the reason why we need learning to predict the transient dynamics. In the next sections, we will show the framework of learning in the frequency domain and conduct numerical verification using the full-order model for system dynamics.

#### IV. LEARNING IN THE FREQUENCY DOMAIN

##### A. Structure of the Neural Network

We construct the structure of neural network shown in Fig. 6, which consists of several Fourier layers for learning in the frequency domain. The input trajectory is first passed through an encoder to integrate the spatial-temporal relationships and

the fault information. The encoded data is then passed through  $l$  level of Fourier Layers [22], where the input of the  $j$ -th layer is  $g_j$  and the output is  $g_{j+1}$  for  $j = 1, \dots, l$ . Each Fourier layer consists of one path with trainable weights  $\theta_j$  that learns periodic components in the frequency domain, and another path with trainable weights  $W_j$  that directly operate on the time-domain data. Intuitively, the second path (often called the pass-through layer in Machine Learning literature) with weights  $W_j$  helps to keep the track of aperiodic and high-frequency components. In the following, we illustrate the structure of each Fourier Layer. The dimension for each tensor will be specified later after we elaborate the encoder in Section V.

##### B. Fourier Layer

For the input of each layer, we conduct discrete Fourier transform  $\mathcal{F}$  to convert the input trajectory into the frequency domain [28]. Inspired by the work in [22], we use neural networks parameterized by  $\theta_j$  to learn in the frequency domain in each layer  $j$ , and then recover the time-domain sequences by inverse Fourier transform  $\mathcal{F}^{-1}$ . This process is defined as Fourier neural operator  $\mathcal{K}_{\theta_j}(\cdot)$  represented by

$$\mathcal{K}_{\theta_j}(g_j) = \mathcal{F}^{-1}(\theta_j \cdot \psi(\mathcal{F}g_j)), \quad (8)$$

where the function  $\psi(\cdot)$  is a low-pass filter that truncates the Fourier series at a maximum number of modes  $k_{\max}$  for efficient computation [22]. Then  $\theta_j$  is the weight tensor that conducts a linear combination of the modes in the frequency domain.

The output of the  $j$ -th layer adds up Fourier neural operator with the initial time-domain sequence weighted by  $W_j$  to recover aperiodic and high-frequency components

$$g_{j+1} = \sigma(W_j g_j + \mathcal{K}_{\theta_j}(g_j)), \quad (9)$$

where  $\sigma$  is a nonlinear activation function whose action is defined component-wise. We use ReLU in this paper.

Even though we limit at most  $k_{\max}$  Fourier modes after the low-pass filter  $\psi$  after the Fourier transform, the linear transform  $W_j$  maintains high-frequency modes. The cut-off frequency  $k_{\max}$  of the low-pass filter is a tradeoff between the number of frequency components that are kept versus the computational complexity. If too few frequencies are selected, there is not enough information in the frequency domain to learn well. If too many frequencies are selected, then we need to learn a high dimensional set of weights, negating the benefit of learning in the frequency domain. The trade-off we adopted is to keep a small number of modes in the frequency domain and pass them through nonlinear layers, while still using a direct path as shown in Fig. 6. Intuitively, this means that the low-frequency modes should be learned in the frequency domain, while the higher-frequency modes can be directly handled using the time domain signal. The exact value of  $k_{\max}$  involves some trial and error. We will show the numerical study about the effect of  $k_{\max}$  in Section VI.

##### C. Multi-Dimensional Fourier Transforms

The above approach of learning the weights in the frequency domain and recovering the trajectory with inverse Fourier transform provides the advantage in fitting oscillatory functions, by learning smooth curvatures and avoiding over-fitting. For

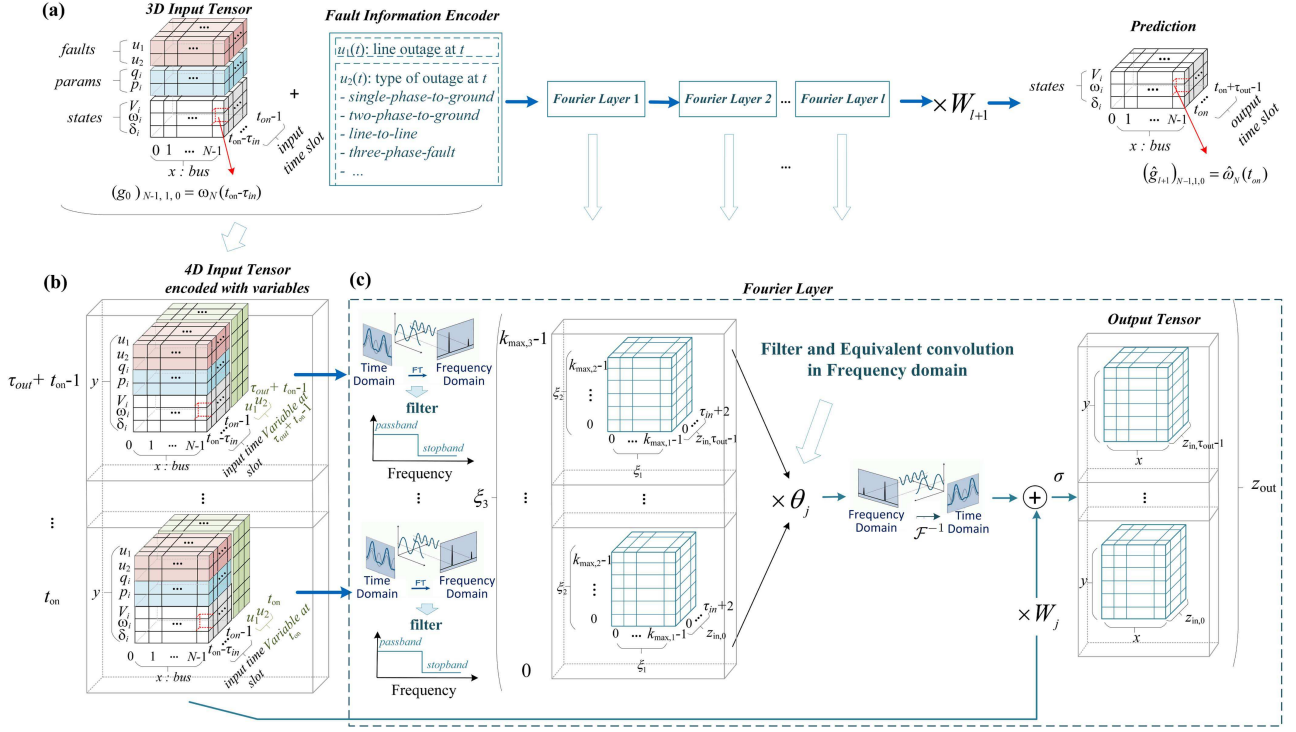


Fig. 7. Structure of power system transient prediction using Fourier Neural Operator (a) The data frame that incorporate time-varying parameters and fault-on/clear actions within input time slots. (b) Incorporation of future fault-clear actions in the output time slots. (c) Fourier layer for Learning in the frequency domain [22].

a system with  $N$  buses, there are  $3N$  state variables we are interested in: the voltage, angle, and frequency at each bus. However, conducting Fourier transform with  $3N$  dimensions is time-consuming even for moderately sized power systems.

Another choice is to neglect the dependence and purely conduct 1D Fourier transform on the time dimension. However, this will degrade the prediction performance since the networked structure is an important cause of the oscillations. Moreover, the time-varying parameters and the fault-clear actions should be considered as well.

To overcome these challenges, we design a novel dataframe that encodes time-varying parameters, fault information, and spatial-temporal relationships in transient dynamics in the next section.

## V. ENCODING SPATIAL-TEMPORAL RELATIONSHIPS

### A. Spatial-Temporal Relationship in Transient Dynamics

We construct 3D tensors to encode the input trajectories such that the spatial-temporal relationships in the power system can be included. In addition, computation complexities of Fourier transforms are also reduced. The proposed framework is shown in Fig. 7(a). Each data point is indexed by three dimensions and written as  $(g_0)_{x,y,z_{in}}$ . The  $x$  axis indexes the buses from 0 to  $N-1$ . The  $y$  axis indexes different state variables for each bus with  $y = 0, 1, 2$  stands for  $\delta_x, \omega_x, V_x$ , respectively. The  $z_{in}$  axis is for input time interval and  $z_{in} = t_{on} - \tau_{in}, \dots, t_{on} - 1$ . For example,  $(g_0)_{5,1,10}$  is the frequency at bus 5 at the time step 10, and  $(g_0)_{6,2,10}$  is the voltage at bus 6 at the time step 10.

We would like to highlight that the spatial information in this manuscript can be understood as the inherent spatial similarity

of signals. For example, it is well known that the angle speed of generators tend to appear as groups that swing in similar patterns [29]. This indicates the correlation along the axis of bus indexes. Moreover, the angle, angle speed and voltage also typically oscillate at similar frequency. This indicates the correlation along the axis of the type of signals. The Fourier transform is most commonly defined along the axis of time, namely, the 1D Fourier transform along the time series. In this paper, we conduct 3D Fourier transform along the dimension of  $x, y$  and the time horizon. This way, the correlations along different buses, along type of signals and along the time horizon are extracted.

### B. Encoding On-Fault and Fault-Clear Information

Importantly, we aim to predict the trajectories under changing net injections and faults. This is different from most previous works that learn a static mapping from input to output time sequences for fixed parameters. Therefore, we encode parameters and fault-clear actions explicitly in the input tensor as shown in Fig. 7(a). Note that the fault-clear action may not be known in advance, and we set up an expected relay time to predict the dynamic behaviors. The aim is to use the learning-based method to reproduce the results from solvers. That is, given the relay actions after the fault happens, the solvers can compute the trajectories after the fault. Likewise, we envision the relay actions as an extra input encoded in the data frame.

The fault information is encoded in  $u_1(t)$  and  $u_2(t)$ , which are variables that contain the location of the fault and the type of fault at the time step  $t$ , respectively. For example, suppose the fault at the time step  $z_{out} + t_{on}$  includes the trip of line 100 and

the fault type is a line-to-line fault. We encode the fault location as 100 and this fault type as 20. Then,  $u_1(z_{\text{out}} + t_{\text{on}}) = 100$  and  $u_2(z_{\text{out}} + t_{\text{on}}) = 20$ , respectively. For the time step  $z_{\text{out}} + t_{\text{on}}$  between the line tripping and line relay,  $u_1(z_{\text{out}} + t_{\text{on}}) \neq 0$  and  $u_2(z_{\text{out}} + t_{\text{on}}) \neq 0$ . If there is no fault happening at the time step  $z_{\text{out}} + t_{\text{on}}$ , then  $u_1(z_{\text{out}} + t_{\text{on}}) = 0$  and  $u_2(z_{\text{out}} + t_{\text{on}}) = 0$ . Hence, the information of line tripping and relay is inherently included when we add  $u_1(t)$  and  $u_2(t)$  to the data frame for  $t = t_{\text{on}} - \tau_{\text{in}}, \dots, t_{\text{on}} + \tau_{\text{out}} - 1$ . Next, we show how to attach  $\mathbf{u}$  and net injections to the dataframe.

### C. Encoding Time-Varying Parameters and Actions

Time-varying parameters include the net active power injection  $\mathbf{p}(t) = (p_1(t), \dots, p_N(t))$  and reactive power  $\mathbf{q}(t) = (q_1(t), \dots, q_N(t))$ . We stack them on the  $y$  axis on  $y = 3$  and  $y = 4$  as shown in blue part of Fig. 7(a). The benefit of this design is that the variance of  $\mathbf{p}(t)$  and  $\mathbf{q}(t)$  through time is naturally incorporated in the  $z_{\text{in}}$  axis. The fault information  $u_1(t)$  and  $u_2(t)$  for  $t = t_{\text{on}} - \tau_{\text{in}}, \dots, t_{\text{on}} - 1$  are stacked to the  $y$ -axis as  $y = 5$  and  $y = 6$ , shown in red part of Fig. 7(a).

Fault-clear actions may happen in the predicted time horizon  $[t_{\text{on}}, t_{\text{on}} + \tau_{\text{out}} - 1]$ . To incorporate future actions and temporal dependence in the prediction time steps, we expand the 3D input tensor in Fig. 7(b) along the output time sequence, with the new axis  $z_{\text{out}} = 0, \dots, \tau_{\text{out}} - 1$  correspond to the time stamp from  $t_{\text{on}}$  to  $t_{\text{on}} + \tau_{\text{out}} - 1$ . This is visualized in the green part of the Tensor in Fig. 7(b), where the axis of  $z_{\text{in}}$  is attached with  $z_{\text{out}} + t_{\text{on}}$ ,  $u_1(z_{\text{out}} + t_{\text{on}})$  and  $u_2(z_{\text{out}} + t_{\text{on}})$ , respectively. Each data point in the 4D tensor is written as  $(\mathbf{g}_1)_{z_{\text{out}}, x, y, z_{\text{in}}}$ , where  $x$  indexes bus,  $y$  indexes the type of signals (e.g.,  $y = 0, 1, 2$  corresponds to the angle, angle speed, and voltage, respectively), and  $z_{\text{out}}$  indexes the output time steps. The index  $z_{\text{in}} = 0, \dots, t_{\text{on}} - 1$  corresponds to the time stamp of the input trajectory. The index  $z_{\text{in}} = t_{\text{on}}$  stands for the location of the fault,  $z_{\text{in}} = t_{\text{on}} + 1$  stands for the type of the fault, and  $z_{\text{in}} = t_{\text{on}} + 2$  stands for the time stamp, respectively. For example, suppose the fault at the time stamp  $z_{\text{out}} + t_{\text{on}}$  includes the trip of line 100 and the fault type is a line-to-line fault. We encode this fault type as number 20 and the fault location as number 100. Then  $(\mathbf{g}_1)_{z_{\text{out}}, x, y, t_{\text{on}}} = 100$ ,  $(\mathbf{g}_1)_{z_{\text{out}}, x, y, t_{\text{on}}+1} = 20$  and  $(\mathbf{g}_1)_{z_{\text{out}}, x, y, t_{\text{on}}+2} = (z_{\text{out}} + t_{\text{on}})\Delta t$  for all  $x$  and  $y$ . Namely, the fault type as number 20, the fault location as number 100, and the time  $(z_{\text{out}} + t_{\text{on}})\Delta t$  are duplicated along the dimension of  $x$  and  $y$ . This way, fault-clearing actions and the output time stamps are encoded in the dataframe without adding extra complexity for batch operation.

After such an encoder, the data frame is converted from a 3D tensor with dimension  $\mathbb{R}^{N \times Y \times \tau_{\text{in}}}$  to a 4D tensor with dimension  $\mathbb{R}^{\tau_{\text{out}} \times N \times Y \times (\tau_{\text{in}}+3)}$ . We train the neural network on a batch of trajectories and the number of trajectories is  $B$ . Hence, the full size of input data is with the dimension  $\mathbb{R}^{B \times \tau_{\text{out}} \times N \times Y \times (\tau_{\text{in}}+3)}$ . We use the Great Britain transmission network with 2224 nodes that will appear later in the case study to give an impression of the size of the data. For the time steps  $\tau_{\text{in}} = 20$  and  $\tau_{\text{out}} = 150$ , the memory space for one input trajectory ( $B = 1$ ) in the Great Britain transmission network is 0.016 GB, and the memory space for 200 input trajectory ( $B = 200$ ) in training is 3.2 GB. The

memory space for other sizes of systems will scale linearly with the number of buses.

The faults of interest in the paper are mainly transmission line faults, and therefore we envision that the  $\mathbf{p}$  and  $\mathbf{q}$  during the prediction horizon of less than 10 s will not deviate too much from the values in the input trajectory. Hence, the encoding of  $\mathbf{p}$  and  $\mathbf{q}$  in horizon of the input trajectory already provides enough information to guide a good prediction. The goal is to use neural networks to map the input tensor to the output tensor with dimension  $\mathbb{R}^{\tau_{\text{out}} \times N \times 3}$  for the predicted dynamics of  $\delta$ ,  $\omega$  and  $V$  along  $\tau_{\text{out}}$  time steps for  $N$  buses.

### D. 3D Fourier Transform

After the encoder, the Fourier transform in (8) is reduced to 3D Fourier transform along the axis of  $x$ ,  $y$  and  $z_{\text{out}}$  computed as

$$(\mathcal{F}\mathbf{g}_j)_{\xi_1, \xi_2, \xi_3, z_{\text{in}}} = \sum_{z_{\text{out}}=0}^{\tau_{\text{out}}-1} \sum_{x=0}^{N-1} \sum_{y=0}^{Y-1} e^{-2\pi i \left( \frac{z_{\text{out}}\xi_1}{\tau_{\text{out}}} + \frac{x\xi_2}{N} + \frac{y\xi_3}{Y} \right)} \cdot (\mathbf{g}_j)_{z_{\text{out}}, x, y, z_{\text{in}}}, \quad (10)$$

where  $\xi_1$ ,  $\xi_2$  and  $\xi_3$  are modes in the frequency domain in the three dimensions after the discrete Fourier transform. Importantly, 3D FFT has been supported by most machine learning frameworks (e.g., Pytorch), which is computationally efficient for both backward propagation in training and forward propagation in prediction.

The structure of the Fourier layer with the 3D Fourier Transform is visualized in Fig. 7(c). After truncating the Fourier series at a maximum number of modes  $k_{\text{max}, i}$  for the  $i$ -th dimension, an equivalent convolution in the frequency domain is conducted using dot-product with weights  $\boldsymbol{\theta}_j \in \mathbb{R}^{k_{\text{max}, 1} \times k_{\text{max}, 2} \times k_{\text{max}, 3} \times (\tau_{\text{in}}+3) \times (\tau_{\text{in}}+3)}$  defined by

$$(\boldsymbol{\theta}_j \cdot \psi(\mathcal{F}\mathbf{g}_j))_{\xi_1, \xi_2, \xi_3, z_{\text{in}}} = \sum_{v=0}^{\tau_{\text{in}}+2} (\boldsymbol{\theta}_j)_{\xi_1, \xi_2, \xi_3, z_{\text{in}}, v} (\mathcal{F}\mathbf{g}_j)_{\xi_1, \xi_2, \xi_3, v} \quad (11)$$

for  $\xi_1 = 0, \dots, k_{\text{max}, 1} - 1$ ,  $\xi_2 = 0, \dots, k_{\text{max}, 2} - 1$ ,  $\xi_3 = 0, \dots, k_{\text{max}, 3} - 1$ ,  $z_{\text{in}} = 0, \dots, \tau_{\text{in}} + 2$  and  $j = 1, \dots, l$ .

The time domain signal is recovered by inverse Fourier transform as follows:

$$\begin{aligned} (\mathcal{K}_{\boldsymbol{\theta}_j}(\mathbf{g}_j))_{z_{\text{out}}, x, y, z_{\text{in}}} &= (\mathcal{F}^{-1}(\boldsymbol{\theta}_j \cdot \psi(\mathcal{F}\mathbf{g}_j)))_{z_{\text{out}}, x, y, z_{\text{in}}} \\ &= \sum_{\xi_1=0}^{k_{\text{max}, 1}-1} \sum_{\xi_2=0}^{k_{\text{max}, 2}-1} \sum_{\xi_3=0}^{k_{\text{max}, 3}-1} e^{2\pi i \left( \frac{z_{\text{out}}\xi_1}{\tau_{\text{out}}} + \frac{x\xi_2}{N} + \frac{y\xi_3}{Y} \right)} \\ &\quad \cdot (\boldsymbol{\theta}_j \cdot \psi(\mathcal{F}\mathbf{g}_j))_{\xi_1, \xi_2, \xi_3, z_{\text{in}}}, \end{aligned} \quad (12)$$

for  $z_{\text{out}} = 0, \dots, \tau_{\text{out}} - 1$ ,  $x = 0, \dots, N - 1$ ,  $y = 0, \dots, Y - 1$ ,  $z_{\text{in}} = 0, \dots, \tau_{\text{in}} + 2$  and  $j = 1, \dots, l$ .

Plugging (12) into (9) gives the output of the  $j$ -th layer  $\mathbf{g}_{j+1} \in \mathbb{R}^{\tau_{\text{out}} \times N \times Y \times (\tau_{\text{in}}+3)}$ . The predicted trajectory  $\hat{\mathbf{s}}_{\text{out}} \in \mathbb{R}^{\tau_{\text{out}} \times N \times 3}$  are obtained from the output of the last layer after a dense

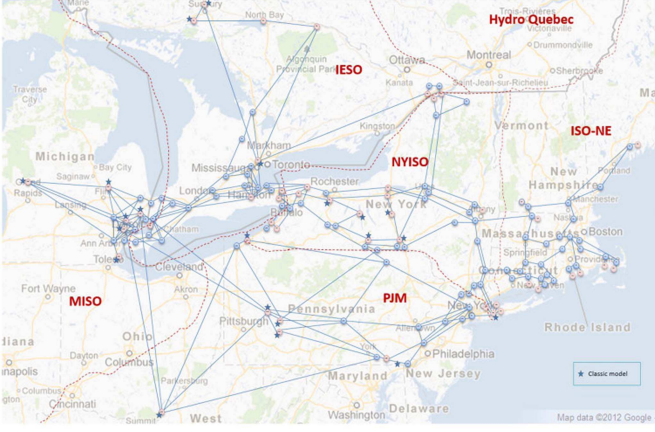


Fig. 8. Topology of Northeastern Power Coordinating Council (NPCC) 48-machine, 140-bus power system [25], [30].

combination weight  $\mathbf{W}_{l+1} \in \mathbb{R}^{(\tau_{in}+3) \times 1}$ .

$$(\hat{\mathbf{s}}_{out})_{z_{out},x,y} = \sum_{z_{in}=0}^{\tau_{in}+2} (g_{l+1})_{z_{out},x,y,z_{in}} (\mathbf{W}_{l+1})_{z_{in}} \quad (13)$$

for  $z_{out} = 0, \dots, \tau_{out} - 1$ ,  $x = 0, \dots, N - 1$ ,  $y = 0, 1, 2$ . The index  $y = 0, 1, 2$  corresponds to the prediction of angle, angle speed, and voltage, respectively. Typically, increased layer  $l$  enables the structure to learn more complex dynamic patterns. In the simulation, we found that four layers are sufficient. In practice, the encoder may also conduct a linear combination on the 4D tensor along the dimension of  $z_{in}$  to increase the representation capability of the neural networks [22]. In that case, the dimension of  $\mathbf{W}_{l+1}$  needs to be adjusted accordingly and all the other computations still remain the same.

### E. Algorithm

The pseudo-code for our proposed method is given in Algorithm 1. The variables to be trained are weights  $\Phi = \{\theta, \mathbf{W}\}$  shown in Fig. 6. Adam algorithm is adopted to update weights in each episode. The main practical benefit is that the learned neural networks are feedforward functions, which can be evaluated orders-of-magnitude faster than conventional iterative solvers. To be clear, these neural networks are not replacements for conventional solvers. Rather, they can be used by system operators to study a much larger set of scenarios of how the system would behave under various types of disturbances. They would be valuable tools that would enable better characterization of the dynamic behavior of the systems, and complement existing tools such as high-fidelity simulators.

## VI. CASE STUDY

In this section, we conduct several case studies to illustrate the effectiveness of the proposed method. We validate the performance of the proposed approach on a realistic power grid by the case studies with the Northeastern Power Coordinating Council (NPCC) 48-machine, 140-bus test system as shown in Fig. 8 [25], [30]. Then, we verify the performance on large-scale

### Algorithm 1: Training and Predicting Power System Transients.

**Training:** Learning rate  $\alpha$ , batch size  $H$ , number of episodes  $I$ , dataset for training

**Initialisation:** Initial weights  $\Phi$  of the neural network

- 1: **for**  $episode = 1$  to  $I$  **do**
- 2: Encode the input trajectory  $\mathbf{s}_{in}^i$  and the output trajectory  $\mathbf{s}_{out}^i$  for  $i = 1, \dots, H$  with fault information in the dataset
- 3: Using the current weights  $\Phi$  of the neural network, compute the predicted trajectory  $\hat{\mathbf{s}}_{out}^i$  for  $i = 1, \dots, H$
- 4: Calculate total loss of all the batches  
 $Loss = \frac{1}{H} \sum_{i=1}^H \frac{\|\hat{\mathbf{s}}_{out}^i - \mathbf{s}_{out}^i\|_1}{\|\mathbf{s}_{out}^i\|_1}$ .
- 5: Update weights in the neural network by passing  $Loss$  to Adam optimizer:  $\Phi \leftarrow \Phi - \alpha \text{Adam}(Loss)$
- 6: **end for**
- Predicting:** Pre-trained weights  $\Phi$  of the neural network
- 7: Encode the input trajectory  $\mathbf{s}_{in}$  with the setup of fault-clear actions
- 8: Using the pre-trained weights  $\Phi$  of the neural network, compute the predicted trajectory  $\hat{\mathbf{s}}_{out}$

power systems using the Great Britain (GB) transmission network, which consists of 2224 nodes, 3207 branches and 394 generators [31]. In Appendix-A of the longer online version of this paper [32], a simple single-machine infinite bus system is used to show the benefit of learning in the frequency domain compared to the time domain.

### A. Simulation and Hyper-Parameter Setup

We construct the network in Fig. 7 with four Fourier layers. We normalize the data of different physical meanings and thus eliminate the effect of the magnitude of features brought by different units. The maximum number of modes in the frequency domain is set to be  $k_{\max,1} = 6$ ,  $k_{\max,2} = 3$  and  $k_{\max,3} = 3$ . The episode number and the batch size are 4000 and 800, respectively. Weights of neural networks are updated using Adam with the learning rate initializes at 0.02 and decays every 100 steps with a base of 0.85. We use Pytorch and a single Nvidia Tesla P100 GPU with 16 GB memory. We use the generic deep neural network (DNN) as a benchmark to compare the performance with the proposed method (labeled as FNO). The DNN has a dense structure and seven layers with ReLU activation, where the width of each layer is 20. The hyper-parameters of DNN are also tuned to achieve their best performances for different tasks. The episode number and the batch size are set the same as FNO.

### B. Performance on NPCC Test System

The performance of the proposed method on a practical power system is verified by simulations on Northeastern Power Coordinating Council (NPCC) test system, which represents the power grid of the northeastern United States and Canada

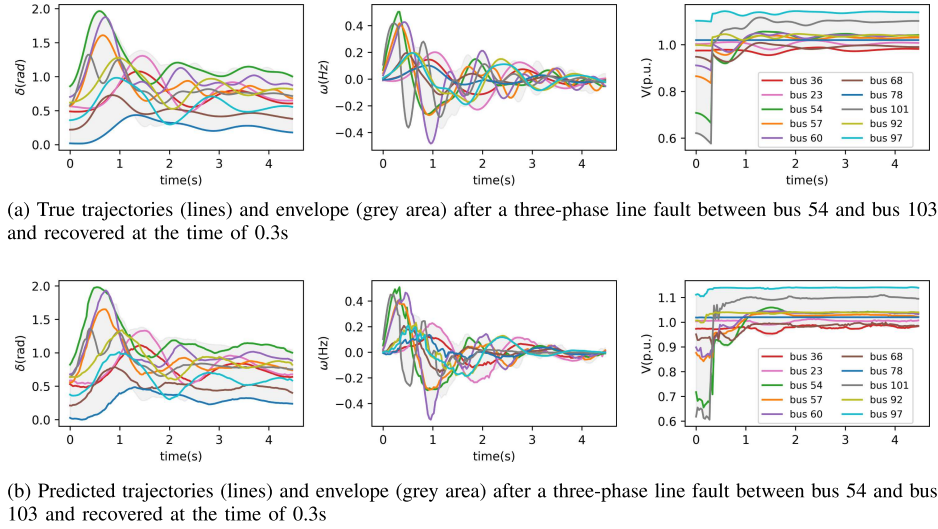


Fig. 9. Stable dynamics of angle  $\delta$  (left), frequency deviation  $\omega$  (middle) and voltage  $V$  (right) in NPCC corresponding to (a) the ground truth produced by a solver. (b) Prediction of FNO. The grey area shows the envelope of the trajectories for all generator buses. Lines with different colors show the trajectories in selected generator buses. The proposed method predicts both the magnitude and oscillations accurately.

and was involved in the 2003 blackout event [30]. The power system toolbox in MATLAB is used to generate dataset of power system transient dynamics with the full 6-order generator model, turbine-governing system and excitors [25]. The power system toolbox utilizes kron-reduced admittance matrix and simulates dynamics with equivalent ordinary differential equations [25]. The trajectories are generated considering the actions of protective relays in 4–20 cycles [33]. We create cases with stressed conditions (stable and unstable) by increasing the level of loads until the system is unstable. The cases with stressed conditions account for 15% in the training set.

The input trajectories evolve  $\tau_{in} = 20$  time steps, with time interval  $\Delta t$  between neighbouring time steps to be 0.03 (i.e., approximate two cycles that can be attained by most phasor measurement unit (PMU)). We predict the subsequent trajectories of the length  $\tau_{out} = 150$  for a total duration of 4.5 s. The training time of 4000 episodes is 4424.33 s. We quantify the prediction accuracy through the relative mean squared error (RMSE) defined as  $\|s_{out} - \hat{s}_{out}\|_2^2 / \|s_{out}\|_2^2$ .

Fig. 9 shows true (i.e., simulated) and predicted trajectories of the system after a three-phase line fault between bus 54 and bus 103 cleared at the time of 0.3 s. Let the time of fault happens as the time  $t = 0$ . The prediction starts at  $t_{on} = 0.06$  s, which means one time step in the input trajectories corresponds to the fault-on system. The grey area is the envelope of trajectories in all generator buses and the lines are the trajectories in ten generator buses. For all the three states variables (i.e.,  $\delta$ ,  $\omega$  and  $V$ ), the predicted trajectories in Fig. 9(b) has similar envelope as the accurate trajectories in Fig. 9(a). The RMSE for the prediction in Fig. 9 is 0.0041. The convergence of the envelope in frequency deviation  $\omega$  to zeros indicates that the system is stable after the fault and its clear action. Moreover, both the magnitude and the periodic oscillations in the ten generator buses are all captured by the prediction with FNO for the on-fault and post-fault period. As illustrated in Fig. 7, the type of fault and the fault-clear action at  $t = 0.3$  s is encoded in the input tensor

of FNO. Correspondingly, Fig. 9(b) predicts a step increase of voltage at  $t = 0.3$  s, which is the same as Fig. 9(a). Notably, the magnitude of the voltage at bus 54 and bus 101 is below 0.8 p.u. before  $t = 1$  s, exceeding the permissible ranges of 5% from nominal. This may cause low-voltage curtailment of the generators and warrant attention from system operators. Therefore, the proposed prediction can provide sufficient information for identifying how danger the system is.

To illustrate the performance of the proposed method in predicting an unstable system, Fig. 10 shows true (i.e., simulated) and predicted trajectories of the system after a line-to-line fault between bus 75 and bus 124 and recovered at the time of 0.3 s. Especially, it is a stressed unstable test case by gradually increasing the level of loads until the system is unstable. Although the magnitude of the trajectories is still bounded, the drifted angle and the collapse of voltage have reflected the unstable behaviors. The proposed method captures both the trend and oscillations of the unstable behaviors. The RMSE for the prediction in Fig. 10 is 0.1198. Moreover, the accuracy in terms of predicting unstable cases is to identify the unstable behaviors. In the next subsection, we verify in the test set that the proposed method can predict all the unstable systems accurately shortly after a fault happens.

The trajectory in the frequency domain (computed by Fast Fourier Transform) for the stable case in Fig. 9 and the unstable case in Fig. 10 is given in Fig. 11 and Fig. 12, respectively. The proposed method also achieves high accuracy in the frequency domain. Moreover, the high-frequency component in both Fig. 11 and Fig. 12 is almost zero. This provides the intuition why the low-pass filter can reduce computational complexity without affecting the prediction performances.

Numerical studies about the effect of  $k_{max}$  on the low-pass filter can be found in Appendix-B of the longer online version [32]. The visualization of the predictions with different influence factors including fault-on/clear actions, fault type, and fault location is shown in Appendix-C of the longer online version of this paper [32].

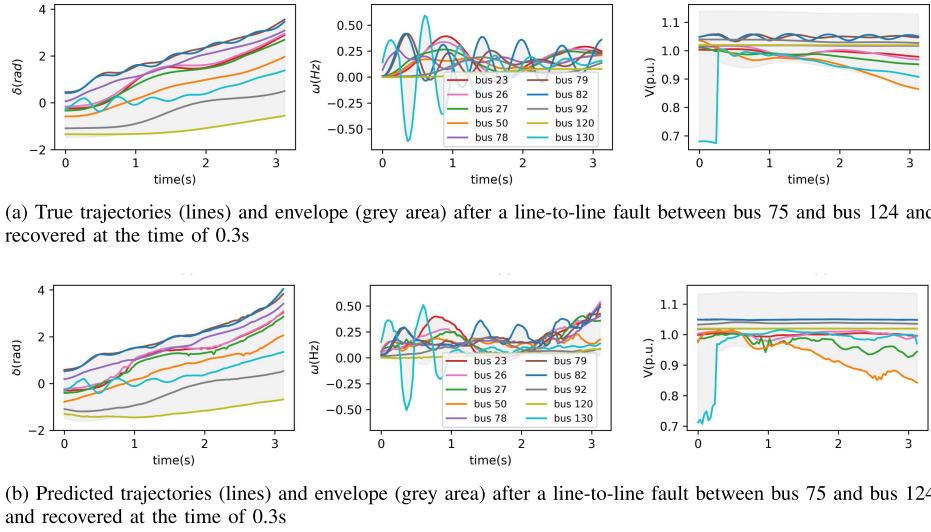


Fig. 10. Unstable dynamics of angle  $\delta$  (left), frequency deviation  $\omega$  (middle) and voltage  $V$  (right) in NPCC corresponding to (a) the ground truth produced by a solver. (b) Prediction of FNO. The grey area shows the envelope of the trajectories for all generator buses. Lines with different colors shows the trajectories in selected generator buses. The proposed method predicts both the magnitude and oscillations accurately.

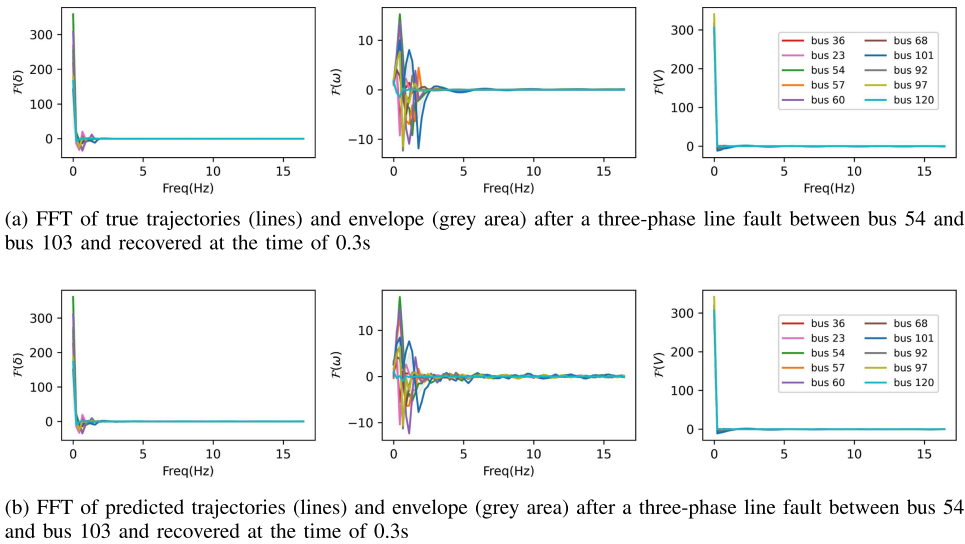


Fig. 11. Frequency-domain trajectories of angle  $\delta$  (left), angle speed deviation  $\omega$  (middle) and voltage  $V$  (right) for stable dynamics in NPCC corresponding to (a) the ground truth produced by a solver. (b) Prediction of FNO.

### C. Quantifying the Performance on NPCC

As shown in Fig. 9 and Fig. 10, the dynamics of the power system transient states differ greatly with different fault types and system parameters. To quantify the performance of the proposed method in stochastic scenarios, we calculate the mean prediction error in the test set with 100 cases where initial states, location of fault, type of fault and fault-clearing time are randomly generated. Three metrics are included:

- 1) Relative mean squared error (RMSE).
- 2) Type1-error: Unstable cases predicted to be stable. This is the more severe type of error (instability is declared when the average value of  $\omega$  from  $t = 4$  s to  $t = 4.5$  s exceed 0.5 Hz).
- 3) Type2-error: Stable cases predicted to be unstable.

Notably, we fix the length of the input trajectories  $s_{in}$  to be  $\tau_{in} = 20$ . The input trajectories contain data before and after the fault (similar to a rolling window). As illustrated in Fig. 4, more data point after the fault will be observed if the prediction starting point  $t_{on}$  is larger (if we wait longer after the fault to do a prediction). The more steps after the fault in  $s_{in}$ , the better the prediction performance. Table I summarizes the metrics for the prediction error corresponding to different number of on-fault cycles (one cycle is  $1/60 = 0.017$  s) involved in the input trajectories  $s_{in}$ .

From Table I, the RMSE of FNO is much lower than the case in DNN, respectively. Interestingly, DNN has the Type1-error to be approximately zero while extremely high Type2-error. The reason is that DNN will easily overfit since the majority (93%) of training samples is stable. By contrast, FNO brings zero

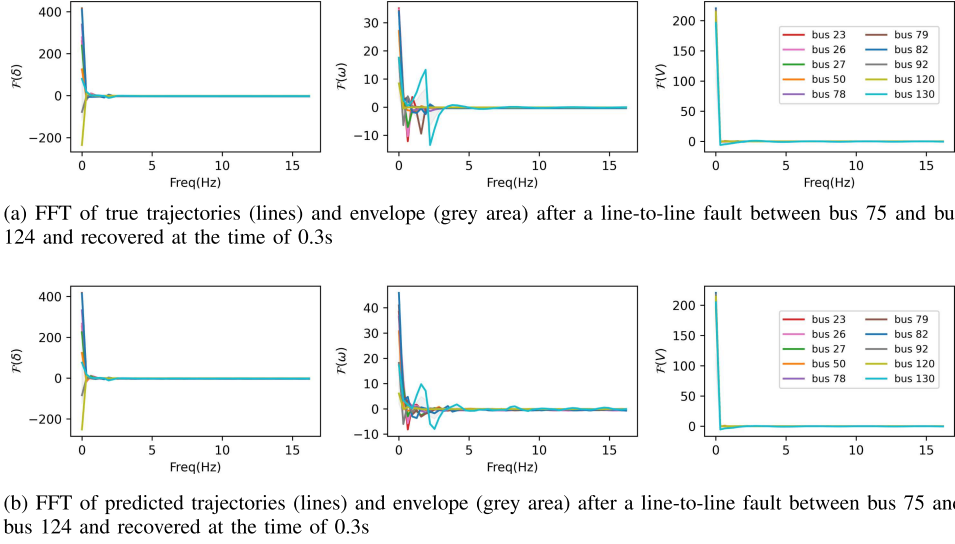


Fig. 12. Frequency-domain trajectories of angle  $\delta$  (left), angle speed deviation  $\omega$  (middle) and voltage  $V$  (right) for unstable dynamics in NPCC corresponding to (a) true trajectories the ground truth produced by a solver. (b) Prediction of FNO.

TABLE I  
PERFORMANCE METRICS IN THE TEST SET WITH 100 RANDOMLY GENERATED CASES

| Metric            | Relative mse  |               |               |               |               | Type1 Error |          |          |          |          | Type2 Error |   |   |       |    |
|-------------------|---------------|---------------|---------------|---------------|---------------|-------------|----------|----------|----------|----------|-------------|---|---|-------|----|
| Cycle after fault | 0             | 2             | 4             | 10            | 20            | 0           | 2        | 4        | 10       | 20       | 0           | 2 | 4 | 10    | 20 |
| FNO               | <b>0.0144</b> | <b>0.0063</b> | <b>0.0055</b> | <b>0.0053</b> | <b>0.0051</b> | <b>0</b>    | <b>0</b> | <b>0</b> | <b>0</b> | <b>0</b> | 0.011       | 0 | 0 | 0     | 0  |
| DNN               | 0.0778        | 0.0712        | 0.0655        | 0.0654        | 0.0656        | 1           | 1        | 1        | 0.667    | 0.167    | 0           | 0 | 0 | 0.011 | 0  |

TABLE II  
RMSE FOR THE DATASET WITHOUT STRESSED UNSTABLE CASES

| Cycle after fault | 0      | 2      | 4      | 10     | 20     |
|-------------------|--------|--------|--------|--------|--------|
| FNO               | 0.0077 | 0.0011 | 0.0009 | 0.0008 | 0.0007 |
| DNN               | 0.0154 | 0.0146 | 0.0141 | 0.0125 | 0.0104 |

TABLE III  
AVERAGE COMPUTATIONAL TIME

| Methods                   | FNO    | Matlab toolbox | Speed up |
|---------------------------|--------|----------------|----------|
| Prediction horizon = 3s   | 0.0036 | 1.69           | 469x     |
| Prediction horizon = 4.5s | 0.0037 | 2.24           | 605x     |
| Prediction horizon = 6s   | 0.0039 | 3.38           | 867x     |

Type1-error and Type2-error once there is an on-fault data point entered in the input trajectories. Therefore, the proposed prediction with FNO will capture all the dangerous unstable cases. The low RMSE indicates that the proposed method can also simulate the dynamics of trajectories accurately.

Considering that unstable behaviors may not be observed in real measurements, we also investigate the performance of the proposed method where unstable behaviors are not present in the training set. Table II shows the comparison of RMSE on the dataset without unstable cases. The RMSE decreases greatly after eliminating the unstable cases. Of course, this is an easier problem, and the RMSE decreases greatly after eliminating the unstable cases.

Lastly, we compare the average computational time in the test set for FNO and Power System Toolbox in MATLAB as shown in Table III. The execution time for 3D Fourier Transform and 3D inverse Fourier Transform in one layer is  $7.79 \times 10^{-5}$  s and  $1.21 \times 10^{-4}$  s, respectively. For the prediction time horizon ranges from 3 s, 4.5, and 6 s, the computational time of FNO is 0.0036 s, 0.0037 s, 0.0039 s, respectively. By contrast, the

computational time of MATLAB toolbox are 469, 605 and 867 times slower than FNO. Therefore, the proposed approach will significantly speed up the simulation for power system transient dynamics.

#### D. Case Study on the Great Britain Transmission Network

We conduct a case study on GB system to test the performance of the proposed method on large power networks. We use ANDES (an open source package for power system dynamic simulation) to generate dataset of power system transient dynamics with the full 6-order generator model, turbine-governing system and excitors [31]. Differential-algebraic equations are solved for dynamic simulation [31]. The input trajectories evolve  $\tau_{in} = 20$  time steps, with a sampling period of 1/30 s (i.e., two cycles). We predict the subsequent trajectories of the length  $\tau_{out} = 150$  steps, for a total duration of 5 s. We use 200 trajectories for training and the training time of 4000 episodes is 9257 s. Using more trajectories may improve the performance, but 200 is sufficient for our simulations.

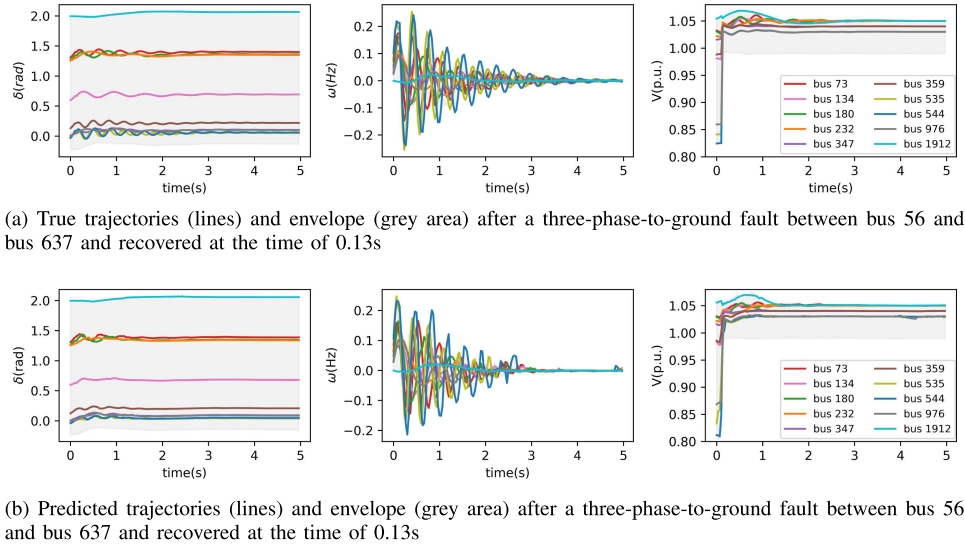


Fig. 13. Dynamics of angle  $\delta$  (left), frequency deviation  $\omega$  (middle) and voltage  $V$  (right) in GB corresponding to (a) the ground truth produced by a solver (b) prediction of FNO. The grey area shows the envelope of the trajectories for all generator buses. Lines with different colors shows the trajectories in selected generator buses. The proposed method predicts both the magnitude and oscillations accurately.

Let the fault happens as the time  $t = 0$ . The prediction starts at  $t_{\text{on}} = 2/30$  s, which means one time step in the input trajectories corresponds to the on-fault system. Fig. 13 shows the prediction on a line fault (a three-phase-to-ground fault between buses 56 and bus 637 and recovered 0.13 s later) that is not in the training set. The grey area is the envelope of trajectories in all generator buses and the lines are the trajectories in ten generator buses. For all the three states variables (i.e.,  $\delta$ ,  $\omega$  and  $V$ ), the predicted trajectories in Fig. 13(b) have similar envelopes as the simulated (i.e., ground-truth) trajectories in Fig. 13(a). Moreover, both magnitude and the periodic oscillations in the generator buses are captured by the prediction with FNO for the on-fault and post-fault period.

In 100 test cases where initial states, location of fault, and fault-clearing time are randomly generated, the mean value of RMSE for DNN is 0.0034. By contrast, the mean value of RMSE for FNO is 0.0001, which is 97% smaller than DNN. Hence, the proposed method achieves much higher accuracy compared with generic deep neural networks.

The above experiments show that the neural network is not simply memorizing but generalizing as well [34], [35]. Since the power system is a synchronized and connected network, transients from different faults could be related. We conjecture that there should be some sparse pattern behind the transient dynamics of the system, and this is the reason why we can learn well with a moderate amount of data. These relationships may be hard to visualize or analytically characterize, which makes machine learning useful. Theoretical analysis of the phenomenon is an important future direction for us.

## VII. CONCLUSION

This paper proposes a frequency domain approach for predicting power system transient dynamics. Inspired by the intuition that there are relatively few dominant modes in the frequency

domain, we construct neural networks with Fourier transform and filtering layers. We design the dataframe to encode the power system topology and fault-on/clear information in transient dynamics, allowing the extraction of spatial-temporal relationships through 3D Fourier transform. Simulation results show that the proposed approach speeds up prediction computations by orders of magnitude and is highly accurate for different fault types. Compared with state-of-the-art AI methods, the proposed method reduces MSE prediction error by more than 50% and vastly improves the detection of unstable dynamics.

The simulation results point to an interesting observation that there are sparse patterns behind the transient dynamics of the system, and it is this sparsity that allows the neural networks to learn and predict. Making the theory rigorous is an important future direction for us. The RAM space of GPU resources constrains the amount of data that can be processed to train the neural networks. Investigating the parallel training on multiple GPU resources and the better usage of RAM space are also important future directions for the proposed method to be utilized in larger systems.

## REFERENCES

- [1] T. T. Mai et al., "Electrification futures study: Scenarios of electric technology adoption and power consumption for the United States," *Nat. Renew. Energy Lab.*, Golden, CO, USA, Tech. Rep. NREL/TP-6A20-71500, 2018.
- [2] R. Golden and B. Paulos, "Curtailed of renewable energy in California and beyond," *Electricity J.*, vol. 28, no. 6, pp. 36–50, 2015.
- [3] P. S. Kundur, N. J. Balu, and M. G. Lauby, "Power system dynamics and stability," *Power Syst. Stability Control*, vol. 3, pp. 700–701, 2017.
- [4] H.-D. Chiang, *Direct Methods for Stability Analysis of Electric Power Systems: Theoretical Foundation, BCU Methodologies, and Applications*. Hoboken, NJ, USA: Wiley, 2011.
- [5] P. W. Sauer, M. A. Pai, and J. H. Chow, *Power System Dynamics and Stability: With Synchrophasor Measurement and Power System Toolbox*. Hoboken, NJ, USA: Wiley, 2017.
- [6] FRCC Event Analysis Team, "FRCC system disturbance and underfrequency load shedding event report," FRCC Operating Committee, Tech. Rep., 2008. [Online]. Available: [https://www.balch.com/files/upload/FRCC\\_System\\_Dist\\_Repor\\_2\\_26\\_2008.pdf](https://www.balch.com/files/upload/FRCC_System_Dist_Repor_2_26_2008.pdf)

- [7] P. Kundur, N. J. Balu, and M. G. Lauby, *Power System Stability and Control*, vol. 7, New York, NY, USA: McGraw-Hill, 1994.
- [8] R. Huang et al., "Faster than real-time dynamic simulation for large-size power system with detailed dynamic models using high-performance computing platform," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, 2017, pp. 1–5.
- [9] Y.-K. Wu, S.-M. Chang, and P. Mandal, "Grid-connected wind power plants: A survey on the integration requirements in modern grid codes," *IEEE Trans. Ind. Appl.*, vol. 55, no. 6, pp. 5584–5593, Nov./Dec. 2019.
- [10] V. H. Chalishazar, S. Poudel, S. Hanif, and P. T. Mana, "Power system resilience metrics augmentation for critical load prioritization," PNNL, Richland, WA, Tech. Rep. PNNL-30837, 2021.
- [11] C. Ren, Y. Xu, and R. Zhang, "An interpretable deep learning method for power system transient stability assessment via tree regularization," *IEEE Trans. Power Syst.*, vol. 37, no. 5, pp. 3359–3369, Sep. 2022.
- [12] L. Zhu, D. J. Hill, and C. Lu, "Intelligent short-term voltage stability assessment via spatial attention rectified RNN learning," *IEEE Trans. Ind. Informat.*, vol. 17, no. 10, pp. 7005–7016, Oct. 2021.
- [13] L. Zhu and D. J. Hill, "Networked time series shapelet learning for power system transient stability assessment," *IEEE Trans. Power Syst.*, vol. 37, no. 1, pp. 416–428, Jan. 2022.
- [14] R. Yan, G. Geng, Q. Jiang, and Y. Li, "Fast transient stability batch assessment using cascaded convolutional neural networks," *IEEE Trans. Power Syst.*, vol. 34, no. 4, pp. 2802–2813, Jul. 2019.
- [15] M. Cui, F. Li, H. Cui, S. Bu, and D. Shi, "Data-driven joint voltage stability assessment considering load uncertainty: A variational Bayes inference integrated with multi-CNNs," *IEEE Trans. Power Syst.*, vol. 37, no. 3, pp. 1904–1915, May 2022.
- [16] B. Xia, H. Wu, Y. Qiu, B. Lou, and Y. Song, "A Galerkin method-based polynomial approximation for parametric problems in power system transient analysis," *IEEE Trans. Power Syst.*, vol. 34, no. 2, pp. 1620–1629, Mar. 2019.
- [17] H.-Y. Su and H.-H. Hong, "An intelligent data-driven learning approach to enhance online probabilistic voltage stability margin prediction," *IEEE Trans. Power Syst.*, vol. 36, no. 4, pp. 3790–3793, Jul. 2021.
- [18] H. Yang, W. Zhang, F. Shi, J. Xie, and W. Ju, "PMU-based model-free method for transient instability prediction and emergency generator-shedding control," *Int. J. Elect. Power Energy Syst.*, vol. 105, pp. 381–393, 2019.
- [19] J. C. Cepeda, J. L. Rueda, D. G. Colomé, and D. E. Echeverría, "Real-time transient stability assessment based on centre-of-inertia estimation from phasor measurement unit records," *IET Gener., Transmiss. Distrib.*, vol. 8, no. 8, pp. 1363–1376, 2014.
- [20] M. Jalali, V. Kekatos, S. Bhela, H. Zhu, and V. A. Centeno, "Inferring power system dynamics from synchrophasor data using Gaussian processes," *IEEE Trans. Power Syst.*, vol. 37, no. 6, pp. 4409–4423, Nov. 2022.
- [21] J. Stiasny, S. Chevalier, and S. Chatzivasileiadis, "Learning without data: Physics-informed neural networks for fast time-domain simulation," in *Proc. IEEE Int. Conf. Commun., Control, Comput. Technol. Smart Grids*, 2021, pp. 438–443.
- [22] Z. Li et al., "Fourier neural operator for parametric partial differential equations," in *Proc. Int. Conf. Learn. Representations*, 2021. [Online]. Available: <https://openreview.net/forum?id=c8P9NQVtmnO>
- [23] K. Jia, Q. Liu, B. Yang, L. Zheng, Y. Fang, and T. Bi, "Transient fault current analysis of IRES considering controller saturation," *IEEE Trans. Smart Grid*, vol. 13, no. 1, pp. 496–504, Jan. 2022.
- [24] P. W. Sauer, M. A. Pai, and J. H. Chow, *Power System Dynamics and Stability: With Synchrophasor Measurement and Power System Toolbox*. Hoboken, NJ, USA: Wiley, 2021.
- [25] J. H. Chow and K. W. Cheung, "A toolbox for power system dynamics and control engineering education and research," *IEEE Trans. Power Syst.*, vol. 7, no. 4, pp. 1559–1564, Nov. 1992.
- [26] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *J. Comput. Phys.*, vol. 378, pp. 686–707, 2019.
- [27] L. Guo, C. Zhao, and S. H. Low, "Graph laplacian spectrum and primary frequency regulation," in *Proc. IEEE Conf. Decis. Control*, 2018, pp. 158–165.
- [28] S. L. Brunton and J. N. Kutz, *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge, U.K.: Cambridge Univ. Press, 2019.
- [29] C. Zhao, U. Topcu, N. Li, and S. Low, "Design and stability of load-side primary frequency control in power systems," *IEEE Trans. Autom. Control*, vol. 59, no. 5, pp. 1177–1189, May 2014.
- [30] W. Ju, J. Qi, and K. Sun, "Simulation and analysis of cascading failures on an NPCC power system test bed," in *IEEE Power Energy Soc. Gen. Meeting*, 2015, pp. 1–5.
- [31] H. Cui, F. Li, and K. Tomsovic, "Hybrid symbolic-numeric framework for power system modeling and analysis," *IEEE Trans. Power Syst.*, vol. 36, no. 2, pp. 1373–1384, Mar. 2021.
- [32] W. Cui, W. Yang, and B. Zhang, "A frequency domain approach to predict power system transients," 2022, *arXiv:2111.01103*.
- [33] P. M. Anderson, C. Henville, R. Rifaat, B. Johnson, and S. Meliopoulos, *Power System Protection*. Hoboken, NJ, USA: Wiley, 2022.
- [34] K. Kawaguchi, Y. Bengio, and L. Kaelbling, *Generalization in Deep Learning*. Cambridge, U.K.: Cambridge Univ. Press, 2022, pp. 112–148.
- [35] Q. Huang, R. Huang, W. Hao, J. Tan, R. Fan, and Z. Huang, "Adaptive power system emergency control using deep reinforcement learning," *IEEE Trans. Smart Grid*, vol. 11, no. 2, pp. 1171–1182, Mar. 2020.



**Wenqi Cui** (Graduate Student Member, IEEE) received the B.Eng. degree in electrical engineering from Southeast University, Nanjing, China, in 2016, and the M.S. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 2019. She is currently working toward the Ph.D. degree in electrical engineering with the University of Washington, Seattle, WA, USA. Her research interests include control, optimization, and machine learning, with applications in cyberphysical power systems.



**Weiwei Yang** received the Bachelor of Science degree in engineering and applied science and the Master of Science degree in computer science from the California Institute of Technology (Caltech), Pasadena, CA, USA. She is currently a Principal Software Development Engineering Manager with Microsoft Research (MSR). Her research interests include developing resource-efficient alternate stochastic gradient descent (alt-SGD) machine learning methods inspired by biological learning, democratizing artificial intelligence by addressing issues of sustainability, robustness, scalability, and efficiency in machine learning.



**Baosen Zhang** (Member, IEEE) received the Bachelor of Applied Science degree in engineering science from the University of Toronto, Toronto, ON, Canada, in 2008, and the Ph.D. degree in electrical engineering and computer sciences from the University of California, Berkeley, CA, USA, in 2013. He was a Postdoctoral Scholar with Stanford University, Stanford, CA, USA. He is currently an Associated Professor of electrical and computer engineering with the University of Washington, Seattle, WA, USA. His research interests include control, optimization and learning applied to power systems, and other cyberphysical systems. He was the recipient of the NSF CAREER Award and several Best Paper awards.