Fast Graphical Learning Method for Parameter Estimation in Large-Scale Distribution Networks

Wenyu Wang, Nanpeng Yu
Department of Electrical and Computer Engineering
University of California, Riverside
Riverside, USA
wwang032@ucr.edu, nyu@ece.ucr.edu

Yue Zhao
Department of Electrical and Computer Engineering
Stony Brook University
Stony Brook, NY 11794, USA
yue.zhao.2@stonybrook.edu

***Abstract*—In distribution systems with growing distributed energy resources, accurate estimation of network parameters is crucial to feeder modeling, monitoring and management. Although existing state-of-the-art parameter estimation algorithms such as physics-informed graphical learning (GL) have accurate estimation, they can potentially suffer from scalability issues due to slow training in larger networks. In this paper, we propose an upgraded graphical learning method called fast graphical learning (FGL) to improve the computational efficiency and scalability while preserving the merits of GL. In FGL, we develop faster alternative algorithms to replace the fixed-point-iteration-based FORWARD and BACKWARD algorithms in GL. These alternative algorithms are based on fast power flow calculation of the current injection method and more efficient state initialization by the linearized power flow model. A comprehensive numerical study on IEEE test feeders and large-scale real-world distribution feeders shows that FGL improves the computational efficiency by as much as 60 times in larger distribution networks while attaining the accuracy of the state-of-art algorithms.**

***Index Terms*—Power distribution network, graph neural network, parameter estimation, smart meter.**

I. Introduction

Rapid growth of distributed energy resources (DERs) requires distribution systems to have accurate three-phase modeling. Monitoring and coordination of DERs rely on advanced applications, such as state estimation, network reconfiguration, three-phase power flow, and optimal power flow. These applications cannot work without accurate topology and parameters of three-phase distribution networks [1]. Although the geographic information system (GIS) has topology and parameter records of distribution networks, the records are usually highly inaccurate due to undocumented system modifications and upgrades [2].

Different from the extensive research in topology estimation [3], [4], more research is needed in parameter estimation of conductor impedance. Compared with transmission networks, parameter estimation in distribution networks is more difficult. The transposed line sections in transmission systems are usually balanced, and thus single-phase models are sufficient. [5]–[9]. In distribution networks, due to their unbalanced conductors and loads, line sections need to be modeled and estimated as 3×3 phase impedance matrices.

Most of existing parameter estimation literature relies on either a simplified single-phase line model or expensive phasor measurement units (PMUs). Typical parameter estimation approaches estimate states and parameter jointly; the techniques include augmenting state vectors with parameters and analyzing residual sensitivity [5] as well as adaptive data selection [6]. Instead of directly estimating parameters, parameter errors are detected by methods such as identification indices [7], enhanced normalized Lagrange multipliers [8], and projection statistics [9]. The supervisory control and data acquisition (SCADA) system data required in these methods are readily available. However, they only work with single-phase models and cannot estimate three-phase line parameters. For accurate parameter estimation in the three-phase lines, usage of PMU or micro-PMU data was proposed. Reference [10] uses state augmentation to estimate three-phase line parameters in transmission lines. In [11], three-phase admittance matrices of distribution lines are estimated by LASSO. Although the PMU-based methods work with three-phase line models, they are cost-prohibitive because a large number of PMUs or micro-PMUs need to be installed.

A few methods have been developed based on readily available smart meter data and they are applicable to three-phase parameter estimation. These methods include multiple linear regression [12], maximum likelihood estimation (MLE) [13], and physics-informed graphical learning model (GL) [14]. Although three-phase line models are considered in [12], it does not work with loads connected between phases. With a linearized power flow model, [13] shows high estimation accuracy on all kinds of phase connections. The accuracy is improved further by [14], which builds GL model based on nonlinear three-phase power flow. Although the GL shows one of the most accurate parameter estimates in three-phase distribution networks with readily available smart meter data, it faces obstacles in achieving scalability due to the rapid increase of training time in large networks.

In this paper, we propose an upgraded graphical learning method called fast graphical learning (FGL), which improves computational efficiency and scalability while preserving the merits of GL. Instead of using the fixed-point-iteration-based

This work was supported by NYSERDA with Agreement Number 159613. This work relates to Department of Navy award N00014-20-1-2858 issued by the Office of Naval Research.

978-1-6654-3254-2/22/$31.00

FORWARD and BACKWARD algorithms in GL, which cause the issue of slow training, we develop alternative algorithms that are equivalent to the original algorithms in terms of functionality but run much faster. These alternative algorithms are based on fast power flow calculation of the current injection method [15] and more efficient state initialization by the linearized power flow model [4]. Compared with the existing parameter estimation methods, the proposed algorithm not only attains the high parameter estimation accuracy of the state-of-the-art methods but also has much better scalability on larger distribution networks by consuming much less training time. A comprehensive numerical study on IEEE test feeders and large-scale real-world distribution feeders shows that FGL improves the computational efficiency by as much as 60 times in larger distribution networks while attaining the accuracy of state-of-the-art algorithms.

The rest of the paper is organized as follows. Section II describes the setup and assumptions of the problem and gives a brief review of the GL method. Section III explains the technical details of the FGL method. In section IV, by a comprehensive numerical study, the performance of the proposed algorithm is evaluated. Section V is the conclusion.

II. Problem Setup and Brief Description of Graphical Learning Framework

A. Problem Setup

Our goal is to estimate the 3×3 phase impedance matrix of primary lines of a distribution network. The network is assumed to contain $\mathfrak{L}$ lines and $N+1$ nodes; node 0 is the source node (e.g., a feeder head). The network contains M loads, which are connected to the non-source nodes. The loads can be single, two, or three-phase. We use $Z_l = R_l + jX_l$ to denote the impedance matrix of line l, where

$$R_l \triangleq \begin{bmatrix} r_l^{aa} & r_l^{ab} & r_l^{ac} \\ r_l^{ab} & r_l^{bb} & r_l^{bc} \\ r_l^{ac} & r_l^{bc} & r_l^{cc} \end{bmatrix}, \quad X_l \triangleq \begin{bmatrix} x_l^{aa} & x_l^{ab} & x_l^{ac} \\ x_l^{ab} & x_l^{bb} & x_l^{bc} \\ x_l^{ac} & x_l^{bc} & x_l^{cc} \end{bmatrix}. \quad (1)$$

Each line segment has six resistance and reactance parameters because Z_l is symmetric. Thus, the total number of parameters is $12\mathfrak{L}$.

B. Assumptions

The available measurement and the feeder information are based on the following assumptions. First, each load has a smart meter measuring its voltage magnitude and real and reactive power; if the load is on phase i, then the voltage and power on phase i are measured; if the load connects phase i and j, then the voltage and power across the two phase are measured; if load connects three phases, the voltage of one phase and the total three-phase power are measured. Second, the voltage at the source node is recorded by the SCADA system. Third, the utility knows all loads' phase connections. Fourth, the utility knows the primary feeder's topology. Fifth, the GIS is assumed to an inaccurate record of the network parameters. Assumptions one and two are normal for smart meters and SCADA systems. Assumptions three, four, and five are appropriate for most GIS records.

C. Brief Review of the GL Method

Fig. 1 shows the framework of the GL method [14] for distribution line parameter estimation. The core of the framework is a graphical learning model, in which nonlinear power flow is embedded. There are three inputs: smart meter-measured power consumption, distribution feeder topology, and distribution line impedances. The structure of the graphical learning model is a network of connected nodes, each representing a bus in the distribution circuit. Each node n has a corresponding state $\boldsymbol{x}_n$, which is the complex bus voltage in three phases. The states are derived by transition functions' iterations until convergence, which is called the FORWARD algorithm. The smart meter voltage magnitudes are calculated from the converged states. The estimated voltage magnitudes are compared with the actual smart meter data to calculate a loss function value. The gradient of the line parameters are computed through a BACKWARD algorithm, and then the line parameters are updated by stochastic gradient descent (SGD).

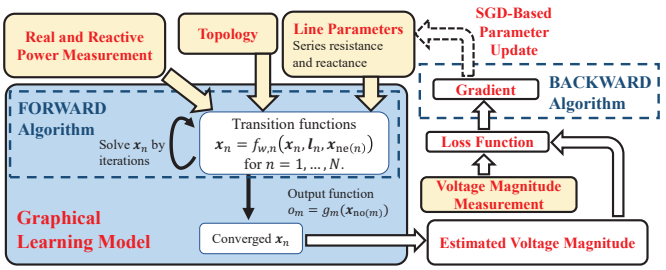

Fig. 1. Framework of proposed graphical learning method.

The technical details of this method are explained in [14]. Here, we only review the three most important computation components: the transition function, the output function, and the gradient computation.

1) Transition Function: The transition function is derived from the nonlinear three-phase power flow model [14]:

$$\boldsymbol{u}_n = Y_{nn}^{-1}\left((\boldsymbol{s}_n^* \oslash \boldsymbol{u}_n^*) + \sum_{k\in ne(n)} Y_{nk}\boldsymbol{u}_k\right) \quad (2)$$

Here, $\boldsymbol{s}_n \triangleq [s_n^a, s_n^b, s_n^c]^T$ is the complex power injection of node n in three phases. $\boldsymbol{u}_n \triangleq [u_n^a, u_n^b, u_n^c]^T$ is the complex voltage of node n in three phases. Define $Y_{nn} \triangleq \sum_{k\in \text{ne}(n)} Y_{nk}$. The 3×3 matrix Y_{nk} is the line admittance matrix between node n and k, which is derived from line parameters. Operator $\oslash$ is element-wise multiplication, $\text{ne}(n)$ is the set of n's neighboring nodes, and $(\cdot)^*$ represents complex conjugate.

For node n, define state vector $\boldsymbol{x}_n$ and feature vector as

$$\boldsymbol{x}_n \triangleq \begin{bmatrix} Re(\boldsymbol{u}_n) \\ Im(\boldsymbol{u}_n) \end{bmatrix}, \ \boldsymbol{l}_n \triangleq \begin{bmatrix} Re(\boldsymbol{s}_n) \\ Im(\boldsymbol{s}_n) \end{bmatrix} \quad (3)$$

Here, $Re(\cdot)$ and $Im(\cdot)$ denote the real and imaginary respectively. Then, from (2), the local transition function and the

corresponding global compact form can be derived [14]:

$$\begin{aligned} \boldsymbol{x}_n &= f_{\boldsymbol{w},n}(\boldsymbol{x}_n, \boldsymbol{l}_n, \boldsymbol{x}_{\mathrm{ne}(n)}) \text{ (local form of node } n) \\ [\boldsymbol{x}] &= F_{\boldsymbol{w}}([\boldsymbol{x}], [\boldsymbol{l}]) \text{ (global compact form),} \end{aligned} \tag{4}$$

where $\boldsymbol{w}$ is the set of line impedance, which is to be estimated. The function $f_{\boldsymbol{w},n}$ is determined by $\boldsymbol{w}$ and the topology of the distribution network. $[\boldsymbol{x}]$, $[\boldsymbol{l}]$, and $F_{\boldsymbol{w}}$ are stacks of $\boldsymbol{x}_n$, $\boldsymbol{l}_n$, and $f_{\boldsymbol{w},n}$ of different nodes. Given $\boldsymbol{w}$, the state $[\boldsymbol{x}]$ in a distribution system can be solved by an algorithm called FORWARD, in which (4) is iteratively applied until $[\boldsymbol{x}]$ converges.

2) Output Function: The solved $[\boldsymbol{x}]$ is then used to calculate the estimated voltage magnitude of smart meters by the output function [14]:

$$\begin{aligned} o_m &= g_m(\boldsymbol{x}_{\mathrm{no}(m)}) \text{ (local form of meter } m) \\ [o] &= G([\boldsymbol{x}]) \text{ (global compact form)} \end{aligned} \tag{5}$$

Here, o_m is smart meter m's voltage magnitude, no(m) is the node that connects smart meter m, $[o]$ is the stack of different smart meter's voltage magnitude o_m. From the Pythagorean theorem, g_m is defined as follows:

$$g_m(\boldsymbol{x}_k) = \begin{cases} \sqrt{(\alpha_k^i)^2 + (\beta_k^i)^2} & \text{if meter } m \text{ is single/three-phase} \\ & \text{measuring phase } i \\ \sqrt{(\alpha_k^i - \alpha_k^j)^2 + (\beta_k^i - \beta_k^j)^2} & \text{if meter } m \text{ is} \\ & \text{two-phase, measuring phase } ij \end{cases} \tag{6}$$

Here, α_k^i and β_k^i are the real and imaginary part of u_k in phase i. To remove trends in o_m, the first difference $\tilde{o}_m(t) \triangleq o_m(t) - o_m(t-1)$ is used instead of o_m. Let $\tilde{v}_m(t)$ be the actual first difference of meter m at time t and let there be M smart meters, then at time t, the loss function is [14]:

$$e_{\boldsymbol{w}}(t) = \frac{1}{M} \sum_{m=1}^{M} \left(\tilde{v}_m(t) - \tilde{o}_m(t) \right)^2 \tag{7}$$

When training the graphical learning model, the loss function is calculated over both the whole dataset and mini-batches, which are smaller subsets of the whole dataset. Thus, for a batch $\mathfrak{T}$ of time indices, the loss function is defined as [14]:

$$e_{\boldsymbol{w}}(\mathfrak{T}) \triangleq \frac{1}{|\mathfrak{T}|} \sum_{t \in \mathfrak{T}} e_{\boldsymbol{w}}(t) \tag{8}$$

3) Gradient Calculation: Directly calculating the loss function's gradient respecting the line parameters is very difficult due to the iterations of FORWARD functions. Thus, an algorithm called BACKWARD was designed by treating the FORWARD iterations as a recurrent neural network (RNN). More details of the BACKWARD function can be found in [14]. The core of the BACKWARD function is to initialize an all-zero vector $\boldsymbol{z}(t)$ and then iteratively update $\boldsymbol{z}(t)$ until convergence by (9) [14]:

$$\boldsymbol{z}(t) = \boldsymbol{z}(t) \cdot \hat{A}(t) + \hat{\boldsymbol{b}}(t) \tag{9}$$

Then, the gradient of $e_{\boldsymbol{w}}(t)$ with respect to $\boldsymbol{w}$ is calcualted as:

$$\frac{\partial e_{\boldsymbol{w}}(t)}{\partial \boldsymbol{w}} = \boldsymbol{z}(t)^{\tau} \cdot \frac{\partial \hat{F}_{\boldsymbol{w}}([\hat{\boldsymbol{x}}(t)], [\hat{\boldsymbol{l}}(t)])}{\partial \boldsymbol{w}} \tag{10}$$

Here $\hat{A}(t)$, $\hat{\boldsymbol{b}}(t)$, and $\hat{F}_{\boldsymbol{w}}([\hat{\boldsymbol{x}}(t)], [\hat{\boldsymbol{l}}(t)])$ are derived by function $F_{\boldsymbol{w}}$ and $G(\cdot)$. $\boldsymbol{z}(t)$ is an intermediate variable used in the Almeida-Pineda algorithm for the RNN backward propagation [16].

III. Technical Methodology

Although GL [14] shows high accuracy in parameter estimation, training the graphical learning model faces obstacles in large-scale problems. The training time of the model grows very quickly as the size of the distribution network increases. This is shown in Table II, in which we record the average runtime of the most time-consuming functions of GL over data timestamp batches of size 10 in test feeders of different sizes.

The computational time of GL's FORWARD algorithm, which is based on fixed-point iteration, grows in two aspects. First, as the network size increases, the number of local transition function $f_{\boldsymbol{w},n}$ in (4) increases proportionally. Second, the number of transition function iterations until convergence grows with the size of the network. The computational time of GL's BACKWARD algorithm grows in a similar way as the FORWARD function.

Here, we propose upgraded FORWARD and BACKWARD algorithms that improve the computation efficiency significantly over the GL's algorithms.

A. Fast-FORWARD Algorithm

In the theoretical derivation of the physics-informed graphical learning method [14], the role of the transition function model of (4) cannot be replaced because it is vital to the construction of the graphical learning model and the derivation of the BACKWARD algorithm. However, when we apply the GL parameter estimation algorithm after constructing the physics-informed graphical learning model, the FORWARD function is only used to compute the state $[\boldsymbol{x}]$, given the parameter set $\boldsymbol{w}$. Thus, we can re-design the FORWARD algorithm without the transition function, as long as it can solve the state $[\boldsymbol{x}]$ given $\boldsymbol{w}$.

In the upgraded FORWARD algorithm, we adopt two methods: by adopting a linearized power flow model [4], we first derive a nearly-accurate initial estimate of states; then, by adopting the current injection method [15], we obtain accurate states $[\boldsymbol{x}]$ from the given $\boldsymbol{w}$ in a timely manner.

The linearized power flow model [4] is shown in (11). The left hand side is the deviation of non-substation nodes from the substation, in which $\check{\boldsymbol{v}}$ denotes voltage magnitudes and $\check{\boldsymbol{\theta}}$ denotes voltage angles. $\check{\boldsymbol{p}}$ and $\check{\boldsymbol{q}}$ denote the real and reactive part of non-substation nodes' power. $\check{A}$ is a $6N \times 6N$ matrix derived from the network topology and line parameters $\boldsymbol{w}$.

$$\begin{bmatrix} \check{\boldsymbol{v}} \\ \check{\boldsymbol{\theta}} \end{bmatrix} = \check{A}^{-1} \begin{bmatrix} \check{\boldsymbol{p}} \\ \check{\boldsymbol{q}} \end{bmatrix} = \begin{bmatrix} \check{A}_{11} & \check{A}_{12} \\ \check{A}_{12} & -\check{A}_{11} \end{bmatrix}^{-1} \begin{bmatrix} \check{\boldsymbol{p}} \\ \check{\boldsymbol{q}} \end{bmatrix} \tag{11}$$

Here, $\check{A}$ can be calculated with $\check{A}_{11}$ and $\check{A}_{12}$, which are derived as follows. Let Y be the $3(N+1) \times 3(N+1)$ admittance matrix of the distribution network organized in three phases. Construct a block diagonal matrix $\Phi \triangleq \text{diag}(I_{(N+1)}, \alpha I_{(N+1)}, \alpha^2 I_{(N+1)})$, in which $\alpha \triangleq e^{-j\frac{2\pi}{3}}$ and $I_{(N+1)}$ is an identity matrix of size $N+1$. $A_{11} = Re(\Phi^{-1}Y\Phi)$ and $A_{12} = -Im(\Phi^{-1}Y\Phi)$. Then $\check{A}_{11}$ can be derived by removing three rows and columns from A_{11} that correspond to the substation. $\check{A}_{12}$ can be derived from A_{12} in a similar way. More details can be found in [4].

The current injection method [15] is then used to obtain accurate states. The basic idea is to iteratively apply (12) until convergence. Here, $[\Delta I]$ is a $6N \times 1$ vector of three-phase real and imaginary parts of nodal current mismatch. $[\Delta V]$ is a $6N \times 1$ vector of three-phase real and imaginary parts of nodal voltage update. J is a Jacobian matrix. In each iteration, $[\Delta I]$ and J are updated based on current state, and then $[\Delta V]$ is solved from (12). The current injection method has been proven to be a fast and accurate way to calculate power flow.

$$[\Delta I] = J[\Delta V] \tag{12}$$

The upgraded Fast-FORWARD Algorithm is shown in Algorithm 1 to perform three-phase power flow calculations and derive the corresponding state $[\boldsymbol{x}]$ given $\boldsymbol{w}$, which produces the same outputs as GL's FORWARD function and is much faster.

Algorithm 1 Fast-FORWARD($\boldsymbol{w}$, t)

Input: Parameter $\boldsymbol{w}$ and time index t.
Output: Distribution system state $[\boldsymbol{x}(t)]$ when line parameter is $\boldsymbol{w}$.

1: Use (11) to calculate $\check{\boldsymbol{v}}$ and $\check{\boldsymbol{\theta}}$. Combine $\check{\boldsymbol{v}}$, $\check{\boldsymbol{\theta}}$, and the source nodes' state $\boldsymbol{x}_0(t)$ to initialize the state $[\boldsymbol{x}(t)]$.
2: **repeat**
3: Update $[\Delta I]$ and J based on current state $[\boldsymbol{x}(t)]$. Solve $[\Delta I]$ from (12) and update $[\boldsymbol{x}(t)]$.
4: **until** The maximum absolute value in $[\Delta I]$ is less than ϵ_{CIM}
5: **return** $[\boldsymbol{x}(t)]$.

B. Fast-BACKWARD Algorithm

In the upgraded Fast-BACKWARD algorithm, we shorten the runtime by reducing the number of iterations of $z(t)$. This is done by an improved approach to properly initialize $\boldsymbol{z}(t)^0$. As shown in Algorithm 2, in Step 3, instead of using $\mathbb{0}_{1\times 12N}$, we first try to initialize $\boldsymbol{z}(t)^0$ by solving the function in (9). If the solution is not feasible (e.g., ill-conditioned matrices), then we still use $\boldsymbol{z}(t)^0 = \mathbb{0}_{1\times 12N}$. The converged $z(t)$ is then processed in Steps 8 to 10 and return the gradient.

C. The Upgraded Fast Graphical Learning Algorithm

Our proposed upgraded fast graphical learning algorithm (FGL) is based on the SGD approach. The algorithm's scheme is very similar to the GL algorithm in [14], which starts with inaccurate GIS records of parameters and updates them iteratively. The complete algorithm of FGL is omitted here due to space limitations. The only difference between FGL and GL is that the FORWARD and BACKWARD functions in GL are replaced by the Fast-FORWARD and Fast-BACKWARD function in FGL.

Algorithm 2 Fast-BACKWARD($\boldsymbol{w}$, $\mathfrak{T}$)

Input: Parameter $\boldsymbol{w}$ and batch of time indices $\mathfrak{T}$.
Output: Gradient $\frac{\partial e_{\boldsymbol{w}}(\mathfrak{T})}{\partial \boldsymbol{w}}$.

1: Calculate $\hat{A}(t)$ and $\hat{\boldsymbol{b}}(t)$ for $t \in \mathfrak{T}$ as in the BACKWARD function in [14].
2: **for** $t \in \mathfrak{T}$ **do**
3: Initialize $\boldsymbol{z}(t)^0 = \hat{\boldsymbol{b}}(t)(I - \hat{A}(t))^{-1}$. If it is not feasible, let $\boldsymbol{z}(t)^0 = \mathbb{0}_{1\times 12N}$. $\tau = 0$.
4: **repeat**
5: $\boldsymbol{z}(t)^{\tau+1} = \boldsymbol{z}(t)^{\tau} \cdot \hat{A}(t) + \hat{\boldsymbol{b}}(t)$
6: $\tau = \tau + 1$
7: **until** $\|\boldsymbol{z}(t)^{\tau} - \boldsymbol{z}(t)^{\tau-1}\|^2 < \epsilon_{\text{backward}} \cdot \|\boldsymbol{z}(t)^{\tau-1}\|^2$
8: Calculate $\frac{\partial \hat{F}_{\boldsymbol{w}}([\hat{\boldsymbol{x}}(t)],[\hat{\boldsymbol{l}}(t)])}{\partial \boldsymbol{w}}$ as in [14], and $\frac{\partial e_{\boldsymbol{w}}(t)}{\partial \boldsymbol{w}} = \boldsymbol{z}(t)^{\tau} \cdot \frac{\partial \hat{F}_{\boldsymbol{w}}([\hat{\boldsymbol{x}}(t)],[\hat{\boldsymbol{l}}(t)])}{\partial \boldsymbol{w}}$, for $t \in \mathfrak{T}$.
9: **end for**
10: $\frac{\partial e_{\boldsymbol{w}}(\mathfrak{T})}{\partial \boldsymbol{w}} = \frac{1}{|\mathfrak{T}|}\sum_{t\in\mathfrak{T}} \frac{\partial e_{\boldsymbol{w}}(t)}{\partial \boldsymbol{w}}$
11: **return** $\frac{\partial e_{\boldsymbol{w}}(\mathfrak{T})}{\partial \boldsymbol{w}}$

IV. Numerical Study

A. Numerical Study Setup

Our proposed FGL algorithm and some state-of-the-art algorithms are tested on a 178-bus feeder. This feeder is modified from a real-world 1922-bus distribution circuit in the service area of National Grid in the State of New York. The modifications are as follows. Based on our problem setup, we only keep the three-phase primary lines. The loads and solar photovoltaic (PV) systems on single-phase or two-phase lines are reconnected to their nearest three-phase buses. In the original feeder, if a series of line sections have no smart meters or branch points between them, we treat them as one equivalent line section in the modified feeder. This is because the smaller line sections in such line series have an infinite number of impedance solutions to any power flow condition. Thus, it is appropriate to treat them as one equivalent line section for feeder modeling. The test feeder is illustrated in Fig. 2.

The feeder is operated at 13.2 kV with a peak load at 4.16 MW, and it contains 177 line sections, 491 loads, 23 solar photovoltaic (PV) systems, and a three-phase capacitor of 900 kVAR. The loads and solar PVs have different phase connections of AN. BN, CN, and ABC. To further accelerate the speed of the algorithms, we partition the distribution feeder into 10 sub-networks following the partition procedure in [14]. The sizes of the sub-networks are selected to be approximately the same. For each sub-network, a node is chosen as the sub-network's source, whose power injection

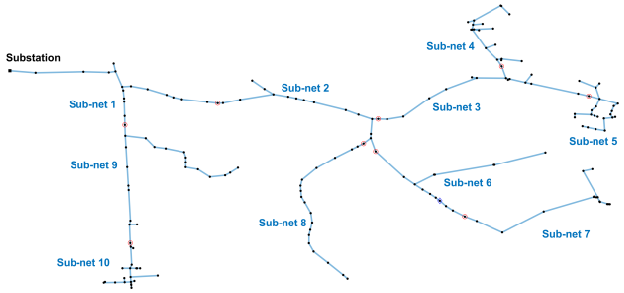

Fig. 2. Schematic of the 178-bus test feeder. The red-circled nodes partition the feeder into sub-networks. The location of the capacitor is marked by a blue diamond.

and voltage magnitude in three phases, as well as phase-to-phase voltage angle differences are measured. Thus, each sub-network can be seen as an independent network and its "source node" works as a substation; the power flow models of all the sub-network can be constructed independently and they work equivalently as the model for a full network. The parameter estimation algorithms are applied to all sub-networks in parallel, and the estimated parameters of each sub-networks are combined together to obtain the final parameter estimation. The partition is also illustrated in Fig. 2, and each sub-network has 14 to 22 line sections.

We use the 15-minute power measurement data from the actual smart meters in the feeder to simulate the real power of loads. Each time series in the dataset has 1440 readings, which correspond to a 15-day time window. The reactive power is simulated by random lagging power factors with a uniform distribution $\mathcal{U}(0.9, 1)$. The temperature and irradiance data in New York are used to simulate the solar PV generation. The capacitor, which is installed in a midstream location of the feeder, is on when the voltage of its connection point is below 1.0 p.u. and off when the voltage is above 1.05 p.u.. Each solar PV and each phase of the capacitor are treated the same as a load with smart meter measurement, which follows the assumptions in Section II-B. To simulate the inaccurate GIS record of line parameters $\boldsymbol{w}_{\text{initial}}$, we randomly select parameter values within $\pm 50\%$ of the correct values under uniform distributions.

Starting from the initial parameter values $\boldsymbol{w}_{\text{initial}}$, the FGL algorithm iteratively updates line parameters. We use the same set of hyperparameters as in GL [14]. The hyperparameter values' setup for the FGL algorithm is as follows. Batch size $n_{\text{batch}} = 10$, early stopping patience $n_{\text{patience}} = 10$, initial step size $s_{\text{initial}} = 1000$, $\alpha = 0.3$, $\beta = 0.5$, and $\epsilon_{stop} = 0.01$. The $\epsilon_{\text{forward}}$ in the Fast-FORWARD function and $\epsilon_{\text{backward}}$ in the Fast-BACKWARD function are set as $1e-20$. These values are determined empirically to make sure the algorithm reduces the loss function value adequately and stops when it saturates. We implement the numerical study by MATLAB, which runs on a workstation with 16 CPU cores (3.0 GHz) and 192 GB RAM.

B. Performance Evaluation

We use mean absolute deviation ratio (MADR) [14] to evaluate the error of line parameters. If $\boldsymbol{w}^\dagger$ and $\boldsymbol{w}$ are the correct and estimated parameters, then the MADR is defined as:

$$\text{MADR} \triangleq \frac{\sum_{i=1}^{12\mathfrak{L}} |w_i - w_i^\dagger|}{\sum_{i=1}^{12\mathfrak{L}} |w_i^\dagger|} \times 100\% \tag{13}$$

We use the percentage of MADR improvement to evaluate the performance of parameter estimation algorithms:

$$\text{MADR improvement} \triangleq \frac{\text{MADR}_{initial} - \text{MADR}_{final}}{\text{MADR}_{initial}} \times 100\% \tag{14}$$

Here, $\text{MADR}_{initial}$ and MADR_{final} are the MADR of the initial and the final parameter estimates. Higher MADR improvement means higher accuracy of parameter estimation.

C. Performance Comparison of the Proposed FGL and State-of-the-Art Algorithms

Our proposed FGL algorithm by design produces the same results as the state-of-the-art algorithm GL [14] but with much faster computation. Thus, we will not compare the parameter estimation accuracy between FGL and GL. Here, we compare the parameter estimation accuracy with another state-of-the-art algorithm: maximum likelihood estimation based on linearized power flow model (LMLE) [13]. Additionally, similar to GL [14], we further evaluate the relative importance of prior knowledge to FGL; we test the effect of parameter constraints (CON). The parameter constraints are defined as follows: the ith parameter w_i is bounded by $\frac{w_{\text{initial},i}}{1+50\%} = \frac{2}{3} w_{\text{initial},i}$ and $\frac{w_{\text{initial},i}}{1-50\%} = 2 w_{\text{initial},i}$. Because of the randomness introduced by SGD, each algorithm is tested with 20 different random seeds. Due to the limited computational power, the computation time of each run of algorithms is limited to 10 hours.

The MADR improvement of the tested algorithms is shown in Table I. Two categories (average/choose optimal value) of MADR improvement are reported here. The first category is the average MADR improvement of the 20 random tests of each algorithm. The second category is the MADR improvement if each sub-network's estimated parameters are from the random test with the lowest loss function value. The MADR improvement of the whole network and each sub-network is shown respectively. In the whole network, for FGL and FGL+CON, we can see that choosing the lowest loss function value of random tests leads to more accurate parameter estimations than the average performance. We can also see that FGL has significantly higher MADR improvement than LMLE with an additional 9.9 to 12 percent. In addition to the advantage of FGL, Table I also shows the benefit of CON, which improves the performance of FGL further with an additional 5.4 to 8.7 percent. These results show that by employing the prior distribution of line parameters, CON can increase the parameter estimation accuracy further.

In the results of sub-networks, we can draw similar conclusions as in the whole network. The MADR improvement varies in sub-networks due to different network sizes, smart

TABLE I
MADR IMPROVEMENT OF PARAMETER ESTIMATION METHODS IN THE TEST FEEDER (AVERAGE / CHOOSE OPTIMAL VALUE)

Network	LMLE	FGL	FGL+CON
Whole Network	10.8% / 13.5%	20.7% / 25.5%	29.4% / 30.9%
Sub-Net 1	10.3% / 9.1%	20.1% / 21.6%	23.1% / 27.1%
Sub-Net 2	7.3% / 9.2%	13.6% / 20.9%	26.7% / 29.3%
Sub-Net 3	9.9% / 12.2%	34.5% / 40.8%	41.6% / 43.7%
Sub-Net 4	4.5% / 4.7%	5.1% / 5.0%	12.4% / 13.2%
Sub-Net 5	11.6% / 14.8%	20.8% / 22.1%	21.0% / 22.3%
Sub-Net 6	12.0% / 24.3%	37.0% / 62.4%	61.8% / 63.5%
Sub-Net 7	9.3% / 9.3%	16.2% / 18.0%	31.6% / 32.9%
Sub-Net 8	13.9% / 16.5%	22.3% / 23.0%	24.9% / 25.5%
Sub-Net 9	17.3% / 21.3%	30.8% / 34.5%	37.9% / 38.2%
Sub-Net 10	7.6% / 9.6%	2.2% / 8.9%	19.0% / 20.4%

meter densities, and smart meter diversities. We can measure the smart meter density by the ratio of smart meter number to bus number, and measure smart meter diversities by the average unique phases measured by smart meters for a bus. Sub-net 6 is a smaller network (14 line sections) with higher smart meter density (9.67) and smart meter diversity (1.8); thus, it has higher MADR improvement. In comparison, sub-net 10 is a larger network (20 line sections) with lower smart meter density (0.86) and diversity (0.86), thus it has lower MADR improvement. Sub-networks such as sub-net 10 have higher level of complexity and less measurement, which makes parameter estimation more difficult. The average accuracy of FGL in sub-net 10 is low because many random tests converge to higher loss values. By choosing the result with the lowest loss value, we can find the test that is better optimized and thus acquire more accurate estimation. We can also see that CON is effective in guiding the parameter estimation algorithm to significantly better results.

D. Computation Time of FGL and State-of-the-Art Algorithms

To evaluate the time efficiency of the parameter estimation methods, we record the average runtime of the most time-consuming functions in each method as shown in Table II. In FGL, the time-consuming functions are the Fast-FORWARD and Fast-BACKWARD; in GL, the time-consuming functions are the FORWARD and BACKWARD; in LMLE, the time-consuming function is a gradient calculation function. These functions are tested on test feeders of 3 different sizes: 7-bus, 14-bus, and 22-bus, which are sub-networks of a modified IEEE 37-bus test feeder in [14]. Each function is run multiple times over random mini-batches of size 10. From Table II, we can see that all the functions' runtimes increase as the feeders get larger. However, the runtimes of the functions in FGL are much shorter than GL, and their growth is much slower. The Fast-FORWARD is over 20 times faster than FORWARD in the 7-bus feeder, and over 60 times faster in the 22-bus feeder. The Fast-BACKWARD is over 1.5 times faster than BACKWARD in the 7-bus-feeder, and about 6 times faster in the 22-bus feeder. Note that when solving the same problem, the numbers of calls of Fast-FORWARD and Fast-BACKWARD are equal to those of FORWARD and BACKWARD respectively as FGL and GL are by design equivalent. Thus, the total runtime of FGL will be much shorter than GL. We also note that the LMLE becomes increasingly slower in larger feeders. In fact, when generating the results in Table I, the average running time of LMLE in a sub-network is 245.4 minute, much longer than FGL (33.1 minute) and FGL+CON (27.5 minute). From these results, we can see that the proposed FGL and FGL+CON are the most efficient by a large margin among the tested parameter estimation methods.

TABLE II
AVERAGE RUNTIME (SECOND) OF MAIN FUNCTIONS OF PARAMETER ESTIMATION METHODS IN FEEDERS OF DIFFERENT SIZES

Method	Function	7-Bus	14-Bus	22-Bus
FGL	Fast-FORWARD	0.0142	0.0313	0.0519
	Fast-BACKWARD	0.068	0.1046	0.1761
GL	FORWARD	0.3028	1.1603	3.2839
	BACKWARD	0.1057	0.3325	1.0065
LMLE	Gradient Calculation	0.0079	0.1866	0.8531

V. CONCLUSION

In this paper, we develop a fast graphical learning algorithm to estimate line parameters of three-phase power distribution networks. The proposed algorithm has wide applicability because it only requires smart meter data, which is readily available, and it estimates three-phase series impedances. By using fast power flow calculation of the current injection method and more efficient state initialization from the linearized power flow model, we develop much faster alternative algorithms to replace GL's FORWARD and BACKWARD algorithms. A comprehensive numerical study on IEEE test feeders and large-scale real-world distribution feeders shows that the proposed method improves the computational efficiency by as much as 60 times in larger distribution networks while attaining the accuracy of state-of-the-art algorithms.

REFERENCES

[1] W. Wang, N. Yu, B. Foggo, J. Davis, and J. Li, "Phase identification in electric power distribution systems by clustering of smart meter data," in *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, Dec. 2016, pp. 259–265.

[2] B. Foggo and N. Yu, "Improving supervised phase identification through the theory of information losses," *IEEE Trans. Smart Grid*, vol. 11, no. 3, pp. 2337–2346, Nov. 2019.

[3] Y. Liao, Y. Weng, G. Liu, Z. Zhao, C.-W. Tan, and R. Rajagopal, "Unbalanced multi-phase distribution grid topology estimation and bus phase identification," *IET Smart Grid*, vol. 2, no. 4, pp. 557–570, Dec. 2019.

[4] W. Wang and N. Yu, "Maximum marginal likelihood estimation of phase connections in power distribution systems," *IEEE Trans. Power Syst.*, vol. 35, no. 5, pp. 3906–3917, Feb. 2020.

[5] P. Zarco and A. G. Exposito, "Power system parameter estimation: A survey," *IEEE Trans. Power Syst.*, vol. 15, no. 1, pp. 216–222, Feb. 2000.

[6] C. Li, Y. Zhang, H. Zhang, Q. Wu, and V. Terzija, "Measurement-based transmission line parameter estimation with adaptive data selection scheme," *IEEE Trans. Smart Grid*, vol. 9, no. 6, pp. 5764–5773, Apr. 2017.

[7] M. R. Castillo, J. B. London, N. G. Bretas, S. Lefebvre, J. Prévost, and B. Lambert, "Offline detection, identification, and correction of branch parameter errors based on several measurement snapshots," *IEEE Trans. Power Syst.*, vol. 26, no. 2, pp. 870–877, Aug. 2010.

[8] Y. Lin and A. Abur, "Enhancing network parameter error detection and correction via multiple measurement scans," *IEEE Trans. Power Syst.*, vol. 32, no. 3, pp. 2417–2425, Sep. 2016.

[9] J. Zhao, S. Fliscounakis, P. Panciatici, and L. Mili, "Robust parameter estimation of the French power system using field data," *IEEE Trans. Smart Grid*, vol. 10, no. 5, pp. 5334–5344, Nov. 2018.

[10] P. Ren, H. Lev-Ari, and A. Abur, "Tracking three-phase untransposed transmission line parameters using synchronized measurements," *IEEE Trans. Power Syst.*, vol. 33, no. 4, pp. 4155–4163, Dec. 2017.

[11] O. Ardakanian, V. W. S. Wong, R. Dobbe, S. H. Low, A. von Meier, C. J. Tomlin, and Y. Yuan, "On identification of distribution grids," *IEEE Trans. Control. Netw. Syst.*, vol. 6, no. 3, pp. 950–960, Jan. 2019.

[12] V. C. Cunha, W. Freitas, F. C. Trindade, and S. Santoso, "Automated determination of topology and line parameters in low voltage systems using smart meters measurements," *IEEE Trans. Smart Grid*, vol. 11, no. 6, pp. 5028–5038, Jun. 2020.

[13] W. Wang and N. Yu, "Parameter estimation in three-phase power distribution networks using smart meter data," in *2020 16th International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*. IEEE, Aug. 2020, pp. 1–6.

[14] W. Wang and N. Yu, "Estimate three-phase distribution line parameters with physics-informed graphical learning method," *IEEE Trans. Power Syst.*, vol. 37, no. 5, pp. 3577–3591, Dec. 2021.

[15] P. A. Garcia, J. L. R. Pereira, S. Carneiro, V. M. Da Costa, and N. Martins, "Three-phase power flow calculations using the current injection method," *IEEE Trans. Power Syst.*, vol. 15, no. 2, pp. 508–514, May 2000.

[16] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Dec. 2008.