This is the Pre-Published Version.

The following publication B. Jiang, C. Qin and Q. Wang, "An Unsupervised Physics-Informed Neural Network Method for AC Power Flow Calculations," in IEEE Transactions on Power Systems, vol. 40, no. 5, pp. 4407-4410, Sept. 2025 is available at https://doi.org/10.1109/TPWRS.2025.3585727.

An Unsupervised Physics-Informed Neural Network Method for AC Power Flow Calculations

Bozhen Jiang, Chenxi Qin, and Qin Wang, *Senior Member, IEEE*

***Abstract*—Power flow (PF) calculation is essential for power system analysis. In recent years, data-driven methods have emerged as a promising approach to accelerate PF calculations. However, these methods require high-quality labeled data and often suffer from poor generalization. To address these issues, an unsupervised physics-informed neural network (UPINN) method is proposed for AC PF calculations. The proposed method follows the general process of Newton-Raphson's method. By minimizing the physics-informed loss function, which is designed based on active and reactive power mismatches, the PF equations will be satisfied directly without the need to calculate the Jacobian matrix's inverse. Proofs of the proposed UPINN training method's convergence are provided. Case study results on the IEEE 24-bus and 118-bus systems demonstrate the feasibility of the proposed approach, showing that UPINN's power flow model can achieve high generalization performance without relying on labeled data.**

***Index Terms*—AC Power flow, unsupervised learning, physics-informed neural network, power systems.**

I. Introduction

POWER flow (PF) calculation is essential for the analysis of power systems. It offers crucial insights for both planning and operation. Traditionally, Newton-Raphson (NR) or similar methods have been reliable and effective in solving PF equations. Recently, artificial neural networks (ANNs) have emerged as a faster alternative. Recent research has concentrated on leveraging ANNs to enhance the speed and accuracy of PF calculations [1]. When evaluating ANN-based methods, generalization is a critical factor. This refers to the algorithm's ability to remain accurate and stable when applied to new and unseen datasets. To improve the generalization of ANNs, researchers have integrated physics-informed neural networks (PINNs) with ANNs by adding flow constraints to the model's loss function [2]. For instance, some studies use branch flow as an additional loss term to ensure compliance with branch flow equations, a method known as the Branch PINN (BPINN) [3], [4]. Other studies employ graph convolutional neural networks (GCNs) to incorporate the power system's structure, referred to as the GCN PINN (GPINN) [5], [6]. However, the BPINN approach presumes that the reactive power of generators is a known variable [4], which typically requires calculation through power flow equations. Meanwhile, the GPINN method does not fully exploit the physical information.

This work was jointly supported by the Science and Technology Project of SGCC under contract number SGHAYJ00NNJS2400004 and the Hong Kong Polytechnic University under grant number P0047690.

B. Jiang, C. Qin and Q. Wang are with the Depart. of Electrical and Electronic Engineering, The Hong Kong Polytechnic University, Hong Kong. (Corresponding author: Q. Wang)

On the other hand, advances in metering infrastructure have simplified the collection of labeled data for PF calculations. However, variability in load fluctuations means that historical and future PF data often do not follow the same distribution, which can significantly impair the performance of ANN-based methods in future applications. To address these challenges, developing $unsupervised$ PINN learning methods for PF calculations can reduce the dependence on extensive labeled data while enhancing generalization performance.

This paper presents an innovative unsupervised PINN (UPINN) method for fast AC PF calculations. We began by defining the inputs and outputs of a PINN-based PF solver. Drawing inspiration from the NR method, we incorporated the imbalance between power injection and power withdrawal at each bus as a physical constraint. This informed the design of a loss function based on these constraints. Additionally, we provide a proof of convergence for the proposed UPINN training method. Ultimately, we developed an unsupervised learning framework using PINNs to solve PF equations, achieving high-precision solutions without the need for labeled data.

The remaining sections are organized as follows: Section II outlines the structure and principles of the proposed UPINN method. Section III presents and discusses the experimental results that validate the effectiveness of the UPINN method. Lastly, Section IV provides the conclusion.

II. Physics-Informed AC Power Flow Solver

A. Newton-Raphson Method for Solving AC PF Models

Assuming a n-bus power system contains r PQ buses, 1 balance bus, and $(n-r-1)$ PV buses. P_i and Q_i represent the active and reactive power injected by bus i respectively, while U_i and δ_i denote the voltage amplitude and phase angle of bus i respectively. For PQ buses, the PF equations are expressed as follows:

$$\Delta P_k = (P_k^g - P_k^d) - \text{RE}(\sum_{i\neq k} S_{kj}), k = 1, .., n-r-1 \quad (1)$$

$$\Delta Q_k = (Q_k^g - Q_k^d) + \text{IM}(\sum_{i\neq k} S_{kj}), k = 1, .., n-r-1 \quad (2)$$

where P_k^g (P_k^d) and Q_k^g (Q_k^d) refer to the active and reactive power of generators (loads) at bus k, respectively. $S_{ij} = U_i \sum_{j=1}^{n} U_j(G_{ij}+jB_{ij})e^{-j(\delta_i-\delta_j)}$ refer to the branch powers from bus i to bus j, and G_{ij} and B_{ij} denote the real and imaginary parts of the element Y_{ij} in the admittance matrix Y. RE$(\cdot)$ and IM$(\cdot)$ refer to taking the real and imaginary parts of a complex number. For PV buses, since $\{Q_{n-r}, ..., Q_{n-1}\}$

are unknown, we only need to write the active PF equations for them:

$$\Delta P_k = (P_k^g - P_k^d) - \mathrm{RE}(\sum_{i \neq k} S_{kj}), k = n-r, .., n-1 \quad (3)$$

Let $\Delta \underline{P} = [\Delta P_1, ..., \Delta P_{n-1}]$ and $\Delta \underline{Q} = [\Delta Q_1, ..., \Delta Q_{n-1-r}]$. Ideally, both $\Delta \underline{P}$ and $\Delta \underline{Q}$ should be zero vectors. The NR solution procedure can then be written as

$$\begin{bmatrix} \Delta \underline{P} \\ \Delta \underline{Q} \end{bmatrix} = - \begin{bmatrix} \frac{\partial \Delta \underline{P}}{\partial \underline{\delta}^T} & \frac{\partial \Delta \underline{P}}{\partial \underline{U}^T} \\ \frac{\partial \Delta \underline{Q}}{\partial \underline{\delta}^T} & \frac{\partial \Delta \underline{Q}}{\partial \underline{U}^T} \end{bmatrix} \begin{bmatrix} \Delta \underline{\delta} \\ \Delta \underline{U} \end{bmatrix}, \quad (4)$$

where $\underline{U} = [U_1, ..., U_{n-1-r}]^T$ is the bus voltage vector and $\underline{\delta} = [\delta_1, ..., \delta_{n-1}]^T$ is the bus angle vector. $\Delta \underline{P_0}$ and $\Delta \underline{Q_0}$ can be calculated using (1)–(3) with initialized $\underline{U_0}$ and $\underline{\delta_0}$. Then, $\Delta \underline{U_1}$ and $\Delta \underline{\delta_1}$ will be calculated with (4). This process will continue until $\Delta \underline{P_{k+1}}$ and $\Delta \underline{Q_{k+1}}$ reach the pre-defined thresholds by updating $\underline{\delta^{t+1}} = \underline{\delta^t} + \Delta \underline{\delta^t}$ and $\underline{U^{t+1}} = \underline{U^t} + \Delta \underline{U^t}$.

B. Unsupervised PINN for AC PF calculation

1) The Input and Output of PINN: The inputs to the PINN include the active and reactive power of PQ buses, the active power and voltage magnitudes of PV buses, and the voltage magnitude and phase angle of the balance bus. The outputs of the PINN are the reactive power values of the PV buses, the voltage magnitudes of the PQ buses, and the phase angles of all buses except the balance bus. Additionally, the reactive power of the PV buses and the active power of the balance bus, which can be derived from the PF equations, are also included as the output.

Fig. 1 illustrates the structure and parameter settings of the PINN model, which incorporates two GCNs for feature extraction. The admittance matrix Y serves as the connection matrix for GCN, while other inputs are used as features to represent the information on the buses. Additional features are extracted through multiple fully connected (FC) layers. Based on the hyperparameters shown in Fig. 1, a parameter search has been conducted during the experiment to identify the best model parameter configuration. Here, $Beta$ scales the output of the activation function to fit the desired range.

2) Physics-Informed Loss Function Design: In contrast to the approaches in [3], [4], which use branch power flow as a loss term constrained by physical information, this study adopts a more direct and practical loss term. Inspired by the iteration process of the NR method described in [7], the ANN generates the solutions $\underline{U^k}$ and $\underline{\delta^k}$. These solutions are then used to compute the power mismatches ΔP_k and ΔQ_k through (1)-(3). Subsequently, we can directly use the imbalance of the power flow equations as the loss term, defined as $\mathcal{L} = \frac{1}{2n} \sum_{i=1}^{n} (w_p \Delta P_i^2 + w_q \Delta Q_i^2)$ in the optimization process. There are two reasons for this. Firstly, $\mathcal{L}$ directly reflects the actual deviations in AC PF calculations. The corresponding ANN weights θ can be updated by $\theta_{t+1} = \theta_t - \eta \frac{\partial \mathcal{L}}{\theta}$, where η is the learning rate. Our method specifically adopts the error concept used in the NR method. By continuously minimizing the imbalance in the PF equations, a proper solution with the desired accuracy level can be obtained. Using imbalances as the loss term aligns with the principles of the NR method. Secondly, this concept inspires the expansion of our method into an unsupervised learning framework. By using the imbalances in the PF equations as the loss term, the model can be trained without requiring additional supervised data. Furthermore, the corresponding optimization problem can be expressed in the following form

$$\begin{aligned} \text{Minimize} \quad & \mathcal{L}(\theta; x, u) = \frac{1}{2n} \sum_{i=1}^{n} (w_p \Delta P_i^2 + w_q \Delta Q_i^2) \\ \text{Subject to} \quad & \Delta P_k(\theta; x, u),\ \Delta Q_k(\theta; x, u),\ \forall k;\ ||\theta||_1 < \eta, \end{aligned} \quad (5)$$

where u, x and θ refer to the input, output and the parameters of PINN model, respectively. According to the chain rule $\frac{\partial \mathcal{L}}{\partial \theta} = \frac{\partial \mathcal{L}}{\partial x} \frac{\partial x}{\partial \theta}$, $\frac{\partial \mathcal{L}}{\theta}$ can be written as

$$\frac{\partial \mathcal{L}}{\partial \theta} = \begin{bmatrix} \frac{1}{n w_p} \Delta P \\ \frac{1}{n w_q} \Delta Q \end{bmatrix}^T \begin{bmatrix} \frac{\partial \Delta \underline{P}}{\partial \underline{U}^T} & \frac{\partial \Delta \underline{P}}{\partial \underline{\delta}^T} & \frac{\partial \Delta P}{\partial P_n} & \frac{\partial \Delta \underline{P}}{\partial \underline{Q}^T} \\ \frac{\partial \Delta \underline{Q}}{\partial \underline{U}^T} & \frac{\partial \Delta \underline{Q}}{\partial \underline{\delta}^T} & \frac{\partial \Delta Q}{\partial P_n} & \frac{\partial \Delta \underline{Q}}{\partial \underline{Q}^T} \end{bmatrix} \begin{bmatrix} \frac{\partial \underline{U}}{\partial \theta} \\ \frac{\partial \underline{\delta}}{\partial \theta} \\ \frac{\partial P_n}{\partial \theta} \\ \frac{\partial \underline{Q}}{\partial \theta} \end{bmatrix} \quad (6)$$

where $\Delta P = [\Delta P_1, ..., \Delta P_n]$, $\Delta Q = [\Delta Q_1, ..., \Delta Q_n]$ and $\underline{Q} = [Q_{n-r}, ..., Q_n]$. Additionally, we employed L1 regularization and proximal optimization methods during network training to encourage sparsity in the weights. The corresponding θ can be updated by $\theta_{t+1} = \mathrm{Prox}_{\eta ||\theta||_1}(\theta_t - \eta \frac{\partial \mathcal{L}}{\partial \theta})$. Furthermore, we introduce the second-order moments of gradients to adaptively adjust the learning rate, i.e. $\theta_{t+1} = \mathrm{Prox}_{\frac{\eta}{\sqrt{v_t}} ||\theta||_1}(\theta_t - \frac{\eta}{\sqrt{v_t}} \frac{\partial \mathcal{L}}{\partial \theta})$, where $v_t = \beta * v_{t-1} + (1-\beta)(\frac{\partial \mathcal{L}}{\partial x^T})^2$, β is the decay rate of the second-order moment and ranges between 0 and 1. The overall PINN training process is shown in Fig. 1. Regarding $\mathcal{L}$, we assume it is continuously differentiable in the N-dimensional real space, i.e. $\mathcal{L} \in C^1(R^N)$.

Theorem 1: Suppose that there exists $L > 0$ so that

$$||\nabla \mathcal{L}(\theta_1; x) - \nabla \mathcal{L}(\theta_2; x)||_2 \leq L ||\theta_1 - \theta_1||_2, \forall \theta_1, \theta_2 \in R^N. \quad (7)$$

Then, for all θ_1 and $\theta_2 \in R^N$, it holds that [8]:

$$\mathcal{L}(\theta_2; x) \leq \mathcal{L}(\theta_1; x) + \nabla \mathcal{L}(\theta_1)^T (\theta_2 - \theta_1) + \frac{L}{2} ||\theta_2 - \theta_1||_2^2. \quad (8)$$

Generally, $L = ||\frac{\partial \mathcal{L}}{\partial \theta}||_{2,max}$. Due to the use of L1 regularization, $\frac{\partial x}{\partial \theta}$ will be very sparse and v_t is bounded by $\frac{G^2}{1-\beta}$, where $G = ||\frac{\partial \mathcal{L}}{\partial x}||_{2,max}$.

Lemma 1: If $||\frac{\partial x}{\partial \theta}||_2 \ll ||\frac{\partial \mathcal{L}}{\partial x^T}||_2$, then $||\frac{\partial \mathcal{L}}{\partial \theta}||_{2,max} \leq ||\frac{\partial \mathcal{L}}{\partial x^T}||_2 \leq ||\sqrt{v_t}|| \leq ||\frac{\sqrt{v_t}}{\eta}||_2$.

According to **Lemma 1**, L can be set as $||\frac{\sqrt{v_t}}{\eta}||_2$.

Theorem 2: Consider (6) and let $\{\theta_k\}$ be generated by the proximal gradient $\theta_{t+1} = \mathrm{Prox}_{\frac{\eta}{\sqrt{v_t}} ||\theta||_1}(\theta_t - \frac{\eta}{\sqrt{v_t}} \frac{\partial \mathcal{L}}{\partial \theta})$. Then for all $k \geq 1$, there are

$$F(\theta_k; x) - F(\hat{\theta}) \leq \frac{G}{2k\eta\sqrt{1-\beta}} ||\theta_0 - \hat{\theta}||_2^2, \quad (9)$$

where $F(\theta; x) := \mathcal{L}(\theta; x) + ||\theta||_1$; $\hat{\theta}$ is any element in Arg min F. Proofs are provided in [7]. This theorem indicates that the sequence $\{\theta_k\}$ generated by the proposed method converges to a stationary point. The convergence rate is featured by the initial distance $||\theta_0 - \hat{\theta}||_2$ and the number of iterations k, η, β and G.

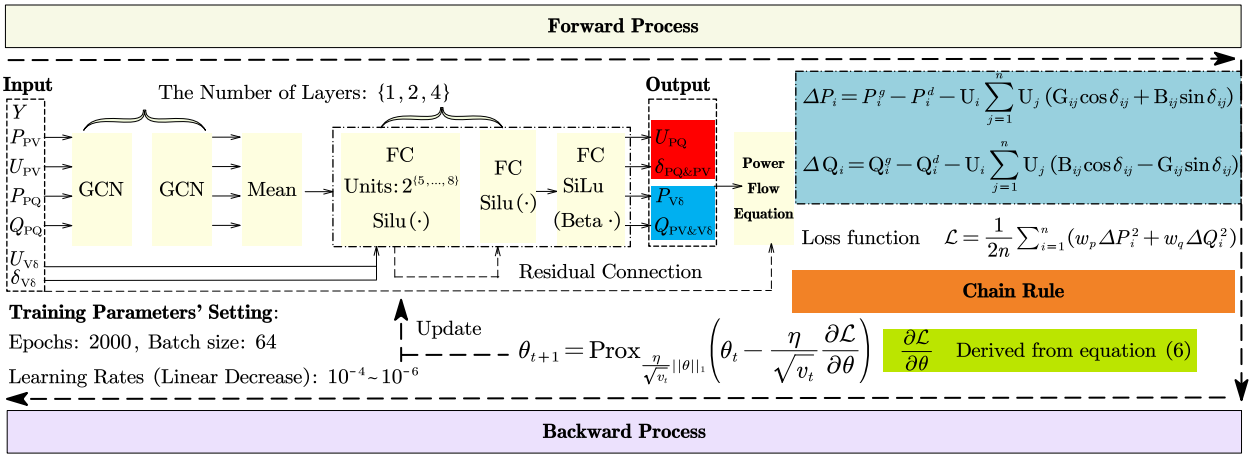

Fig. 1. Model Architecture and PINN for PF Training Process

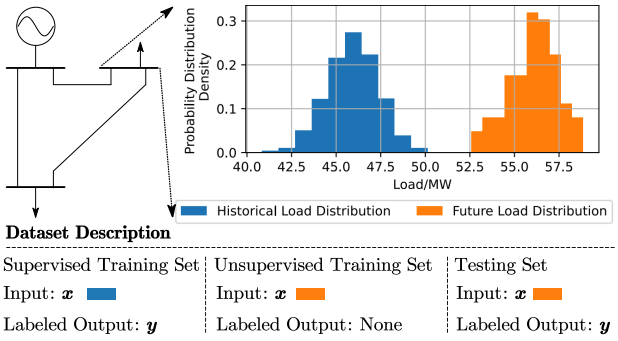

Fig. 2. Data Set Distribution Example

TABLE I
PERFORMANCE COMPARISONS ON THE IEEE 24-BUS AND IEEE 118-BUS SYSTEMS

Test Cases and Algorithm		Error (RMSE) ΔP	ΔQ	Br-P	Br-Q	U_{PQ}	Q_{PV}	$\delta_{PQ\&PV}$	Time/ sec.
IEEE case24	UPINN	0.0399	0.0683	0.0048	0.0348	0.0016	0.0123	0.0067	$\leq$ 0.1
	BPINN	0.9095	5.9128	0.4065	3.0583	0.1121	0.1962	0.0549	$\leq$ 0.1
	GPINN	0.7219	4.3715	0.3034	2.2949	0.0886	0.0937	0.0547	$\leq$ 0.1
	PM.jl	0.0041	0.0197	0.0000	0.0000	0.0000	0.0000	0.0000	37.6
IEEE case118	UPINN	0.0475	0.0358	0.0021	0.0102	0.0014	0.0204	0.1301	$\leq$ 0.1
	BPINN	0.6839	3.3929	0.2450	1.8415	0.0739	0.0961	0.0848	$\leq$ 0.1
	GPINN	0.8362	3.9006	0.2668	2.0220	0.0774	0.0562	0.0693	$\leq$ 0.1
	PM.jl	0.0053	0.0201	0.0000	0.0000	0.0000	0.0000	0.0000	150.1

III. CASE STUDY

A. Data Set

This paper utilizes PowerModel.jl (PM.jl) [9] to generate AC power flow data and implements the standard IEEE 24-bus and 118-bus systems to evaluate the proposed methods. We adjust the loads to a certain ratio based on the standard IEEE 24-bus and 118-bus systems, creating a total of 16,000 sample points for each test system. Given the availability and precision of probabilistic load forecasting results, we assume that the training set contains a subset of data that shares the same distribution as the test set. However, it is important to acknowledge that while ultra-short-term or short-term load forecasts are typically accurate [10], the associated training set data often lacks labels for AC power flow due to time constraints. Therefore, we have further segmented the dataset into three components: a supervised training set (the first 10,000 samples), an unsupervised training set (samples 10,001 to 15,000), and a test set (the remaining samples), as illustrated in Fig. 2. Since the data is already in per-unit format, no further normalization is applied in this study.

B. Result Analysis

The performance results are shown in the Table I, using PowerModel.jl as a benchmark. The root mean square error (RMSE) serves as a metric to evaluate the performances. Various factors are assessed, including the active and reactive power balance at all buses (ΔP and ΔQ), the branch active and reactive power (Br-P and Br-Q), the voltage magnitudes and reactive power of PQ buses (U_{PQ} and Q_{PV}), and the phase angles for PQ and PV buses ($\delta_{PQ\&PV}$). The term "$Time$" in Tables I indicates the total time required for the entire testing set. The comparative analysis shows that the proposed UPINN method offers superior generalization performance over existing approaches. It maintains high accuracy even when faced with distribution shifts, indicating its effectiveness in performing PF calculations under conditions of high load uncertainty. Compared to existing methods, the proposed approach improves precision by nearly two orders of magnitude. Additionally, the time required for the ANN-based PF solution is significantly less than that of the NR method.

Furthermore, the UPINN is part of a category of learning methods that do not require labeled data, akin to the autoregression technique used in large language models. The proposed approach could be used to improve the speed of solving large-scale AC optimal power flow (OPF) problems.

C. Discussion on UPINN Performance as the Power System Approaches the Maximum Loading Point

As demonstrated in Section II-A, it is essential to iteratively compute $\underline{\Delta U_1}$ and $\underline{\Delta \delta_1}$ based on equation (4). This process inevitably involves calculating the inverse of the Jacobian matrix. However, when the Jacobian matrix approaches singularity in certain regions (especially as the power system approaches the maximum loading point), it becomes non-invertible. In contrast, as discussed in Section II-B, during the

TABLE II
PERFORMANCE COMPARISONS UNDER NOISE CONDITION ON THE IEEE 24-BUS SYSTEM

Algorithm	Error (RMSE)			
	ΔP	ΔQ	Br-P	Br-Q
UPINN $\sigma = 0.001$	0.0393	0.0729	0.0055	0.0400
PowerModel.jl $\sigma = 0.001$	0.0059	0.0372	0.0028	0.0202
UPINN $\sigma = 0.005$	0.0507	0.1822	0.0149	0.1069
PowerModel.jl $\sigma = 0.005$	0.0260	0.1629	0.0142	0.1021
UPINN $\sigma = 0.01$	0.0707	0.3508	0.0284	0.2032
PowerModel.jl $\sigma = 0.01$	0.0540	0.3452	0.0287	0.2047

training process of the ANN, we only need to compute the Jacobian matrix itself, without the need to solve its inverse. This greatly enhances the stability of the proposed method.

D. Discussion on UPINN Performance under Measurement Unit Error Noise

Our proposed algorithm also presents high robustness, as demonstrated in [7]. To empirically validate these theoretical insights, we conducted experiments on the IEEE 24-bus system under varying noise levels, with $\epsilon \sim \mathcal{N}(0, \sigma^2)$ for $\sigma \in \{0.001, 0.005, 0.01\}$ [11]. In these tests, PowerModel.jl computes performance metrics using the noisy input $\hat{u}$ and the true solution x, whereas our method (UPINN) evaluates them using $\hat{u}$ and the network output $\hat{x}$. The results, as summarized in Table II, confirm that the proposed approach maintains strong performance across all noise conditions, demonstrating its practical reliability in noisy environments.

IV. CONCLUSION

This article introduces an UPINN method designed to eliminate the need for high-quality labeled data while enhancing generalization capabilities in AC PF calculations. The proposed method adheres to the general process of the NR method. By minimizing a physics-informed loss function, which is based on active and reactive power mismatches, the PF equations are directly satisfied without the need to compute the Jacobian matrix's inverse. Validation results on IEEE 24-bus and 118-bus systems demonstrate that the proposed UPINN-based model achieves high generalization performance without relying on labeled data. This method can be seamlessly integrated into existing ANNs for solving AC OPF by providing gradient information through automatic differentiation. Additionally, it can be integrated with the *MASK* method to solve the power flow equation in situations involving PQ-PV transformations. In future work, we plan to develop fully UPINN-based methods for solving AC OPF problems.

REFERENCES

[1] X. Hu and *et al.*, "Physics-guided deep neural networks for power flow analysis," *IEEE Trans. Power Syst.*, vol. 36, no. 3, pp. 2082–2092, 2021.

[2] B. Huang and J. Wang, "Applications of physics-informed neural networks in power systems-a review," *IEEE Trans. Power Syst.*, vol. 38, no. 1, pp. 572–588, 2022.

[3] Y. Yang, Z. Yang, J. Yu, B. Zhang, Y. Zhang, and H. Yu, "Fast calculation of probabilistic power flow: A model-based deep learning approach," *IEEE Transactions on Smart Grid*, vol. 11, no. 3, pp. 2235–2244, 2019.

[4] Y. Zhu, Y. Zhou, W. Wei, and N. Wang, "Cascading failure analysis based on a physics-informed graph neural network," *IEEE Trans. Power Syst.*, vol. 38, no. 4, pp. 3632–3641, 2022.

[5] D. Wang and *et al.*, "Probabilistic power flow solution with graph convolutional network," in *IEEE PES ISGT-Europe*. IEEE, 2020.

[6] M. Gao and *et al.*, "Physics embedded graph convolution neural network for power flow calculation considering uncertain injections and topology," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 11, 2024.

[7] B. Jiang, "The paper appendix," https://github.com/BozhenJIANG/The-UPINN-for-PF-Paper-Appendix, 2025, accessed: 2025-6-21.

[8] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[9] C. Coffrin and *et al.*, "Powermodels. jl: An open-source framework for exploring power flow formulations," in *PSCC*, Dublin, Ireland, 2018.

[10] B. Jiang, Y. Wang, Q. Wang, and H. Geng, "A novel interpretable short-term load forecasting method based on kolmogorov-arnold networks," *IEEE Trans. Power Syst.*, pp. 1–4, 2024.

[11] C. Pei, Y. Xiao, W. Liang, and *et al.*, "Canonical variate analysis for detecting false data injection attacks in alternating current state estimation," *IEEE Trans. Network Sci. Eng.*, vol. 11, no. 4, pp. 3332–3345, 2024.