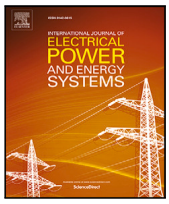
Electrical Power and Energy Systems 174 (2026) 111525

Contents lists available at ScienceDirect

International Journal of Electrical Power and Energy Systems

journal homepage: www.elsevier.com/locate/ijepes



Physics-informed neural network with adaptive activation for power flow

Zeynab Kaseb *, Stavros Orfanoudakis, Pedro P. Vergara, Peter Palensky

Department of Electrical Sustainable Energy, Delft University of Technology, The Netherlands

ARTICLE INFO

Keywords:
End-to-end algorithm
Deep learning
Multilayer perceptron
Load flow
Physics-aware neural network
Voltage calculation

ABSTRACT

We introduce a physics-informed neural network for power flow (PINN4PF) that effectively captures the nonlinear dynamics of large-scale modern power systems. The proposed neural network (NN) architecture consists of two important advancements in the training pipeline: (A) a double-head feed-forward NN that aligns with power flow (PF), including an activation function that adjusts to the net active and reactive power injections patterns, and (B) a physics-based loss function that partially incorporates power system topology information through a novel hidden function. The effectiveness of the proposed architecture is illustrated through 4-bus, 15-bus, 290-bus, and 2224-bus test systems and is evaluated against two baselines: a linear regression model (LR) and a black-box NN (MLP). The comparison is based on (i) generalization ability, (ii) robustness, (iii) impact of training dataset size on generalization ability, (iv) accuracy in approximating derived PF quantities (specifically line current, line active power, and line reactive power), and (v) scalability. Results demonstrate that PINN4PF outperforms both baselines across all test systems by up to two orders of magnitude, not only in terms of direct criteria, e.g., generalization ability, but also in terms of derived physical quantities.

1. Introduction

Power flow (PF) analysis is a foundational computational method for assessing and determining the steady-state operating conditions of electrical power systems by computing voltage magnitudes and phase angles at all buses. This analysis is crucial to ensure the reliability, stability, and optimal performance of power systems. It also allows operators to make informed decisions and mitigate potential issues, such as voltage violations, overloads, and system instability [1,2].

PF analysis can be performed by solving nonlinear and non-convex algebraic equations derived from the nodal balance of the net active and reactive power injections per bus in power systems [3]. Exact analytical solutions for these equations, which also involve impedance parameters, load characteristics, and generator conditions in power systems, are not possible. Therefore, iterative numerical methods, such as Gauss–Seidel and Newton–Raphson (NR) methods, are conventionally employed to converge to a solution that satisfies the PF equations within specified accuracy limits. Eventually, the solutions yield voltage phasors across the entire power system and provide a comprehensive understanding of the operational state [4].

Conventional iterative numerical methods, however, face computational challenges when applied to large-scale modern power systems [5]. These challenges include poor scalability, numerical instability under heavily loaded or ill-conditioned scenarios, and convergence failures [6]. Such methods also exhibit a considerable increase in computation time with system size and often require accurate initialization, which is not always feasible in practice. In addition, they struggle to account for system uncertainties, such as inaccurate line profiles due to weather conditions and aging, different types of loads, and missing data on renewable energy resources [7]. Ineffective PF analysis under these circumstances can lead to safety threats, including renewable energy generation curtailment and blackouts, as well as difficulties in accommodating distributed energy resources [8,9]. Addressing these challenges necessitates developing new approaches for PF analysis that are computationally efficient and numerically stable.

Deep learning approaches, and more specifically, neural networks (NNs), are currently the most powerful set of numerical tools for providing accurate approximations of nonlinear problems (e.g., [10–12]). Several studies have demonstrated the superiority of deep learning approaches in PF analysis in terms of computational time by orders of magnitude (e.g., [13–15]). At the same time, the accuracy of the solutions is competitive compared to the conventional iterative numerical methods (e.g., [16,17]). NNs, therefore, can address the challenges mentioned above by leveraging the availability of massive measurements and/or augmented data, learning complex input–output relationships that are often difficult or even impossible for conventional iterative numerical methods to comprehend, and achieving the accuracy required for real-world applications [18]. Nevertheless, NNs are subject to overfitting, the lack of generalization, and scalability issues. They are very unlikely to meet the physical constraints. Moreover, their

* Corresponding author.
E-mail address: Z.Kaseb@tudelft.nl (Z. Kaseb).

https://doi.org/10.1016/j.ijepes.2025.111525
Received 13 August 2025; Received in revised form 10 October 2025; Accepted 22 December 2025
Available online 11 January 2026

Table 1
List of literature on deep learning-based power flow (PF) and optimal power flow (OPF). Studies are grouped into: (i) pre-training; (ii) training; (iii) post-training.

Stage	Study	Application	Input*	Output*	Approach
Pre-training	Lei et al. (2021) [22]	OPF	p^d, q^d	p^g, q^g, $\|v_i\|$, δ_i	Integrating measurement data into the pre-trained NN
Training	Yang et al. (2020) [23]	PF	p_i, q_i	$\|v_i\|$, δ_i	Adding a penalty term of branch flows to loss function
	Jeddi & Shafieezadeh (2021) [25]	PF	Δp_i, Δq_i, $\|v_i\|$, δ_i, l_{ij}	$\|v_i\|$, δ_i	Using self-attention mechanism and adding a penalty term of nodal voltage to loss function
	Hu et al. (2021) [29]	PF	p^d, q^d, p^g, $\|v_g\|$	$\|v_i\|$, δ_i	Adding regularization terms based on structure of AC PF equations and topology of power system
	De Jongh et al. (2022) [26]	PF	p_i, q_i, $\|v_i\|$, δ_i, $\mathcal{G}$	p_i, q_i, $\|v_i\|$, δ_i	Adding a penalty term of nodal active and reactive power to loss function
	Kody et al. (2022) [28]	PF	$\|v_i\|$, δ_i	p_i, q_i, s_{ij}	Adding a physics-based term to NN formulation
	Yang et al. (2023) [27]	PF	p_i, q_i, $\mathcal{G}$, A	$\|v_i\|$, δ_i	Developing a topology-adaptive graph NN and adding penalty terms of equality constraints to loss function
	Lin et al. (2023) [30]	PF	p^d, q^d, p^g, $\|v_g\|$, l_{ij}, $\mathcal{G}$	p_i, q_i, $\|v_i\|$, δ_i	Combining message passing graph NNs and high-order graph convolutional NNs
	Liu et al. (2024) [31]	PF	p^d, q^d, l_{ij}	v_i	Using a physics-inspired structure integrating physical laws of PF
	Nellikkath & Chatzivasileiadis (2021) [32]	OPF	p^d	p^g	Adding a penalty term of active power generation and consumption to loss function
	Nellikkath & Chatzivasileiadis (2022) [33]	OPF	p^d, q^d	p^g, q^g	Adding a penalty term of active and reactive power generation and voltage magnitude to loss function
	Hu & Zhang (2023) [34]	OPF	p^d, q^d	$\|v_i\|$, δ_i	Developing an activation function and a modified loss function to satisfy physical constraints
	Wu et al. (2024) [35]	OPF	p_i, $\|v_i\|$	$\|v_i\|$, δ_i	Using a model-agnostic meta-learning algorithm and enforcing PF equations as constraints in loss function
Post-training	Donti et al. (2021) [24]	OPF	p^d, q^d, $\|v_r\|$	p^g, $\|v_{r,g}\|$	Implementing completion idea (hard constraints)
	Pan et al. (2022) [36]	OPF	p^d, q^d	p^g, q^g, $\|v_i\|$, δ_i	Implementing completion idea (hard constraints)
	Li et al. (2023) [37]	OPF	p^d	p^g, δ_i	Filtering generated values to ensure feasibility by comparing with actual values

*p_i=nodal active power, q_i=nodal reactive power, s_{ij}=line apparent power, $|v|$=voltage magnitude, δ=voltage angle, $\mathcal{G}$=graph topology, l_{ij}=line physical properties, A=Adjacency matrix, p^g=generator active power, q^g=generator reactive power, p^d=load active power, q^d=load reactive power.

performance relies heavily on the training dataset size and quality. In contrast, not enough data is always available due to privacy reasons and the presence of missing data, among others [19].

Several studies in the literature have investigated the impact of various modifications on the efficacy of deep learning approaches for PF analysis (e.g., [20,21]). These modifications are categorized into three main stages: (i) pre-training (e.g., [22]); (ii) training (e.g., [23]); and (iii) post-training (e.g., [24]), as outlined in Table 1. A majority of these studies have primarily focused on the training stage. Table 1 also highlights instances where topology information, such as line physical properties (e.g., [25]) and graph topology (e.g., [26]), has been integrated into the training stage. The literature review also indicates the utilization of different prior knowledge for deep learning approaches for PF analysis and optimal power flow (OPF), including equality and inequality constraints (e.g., [27]) and PF equations (e.g., [28]). Among the three presented stages, this work focuses on the training stage.

Measurement data has been used in [22] to enhance deep learning approaches for OPF. While this modification has demonstrated improvements in accuracy, its applicability is confined to the pre-training stage. Physical properties have been used on a few occasions in the past to enhance deep learning approaches for PF analysis (e.g., [25,38]), where the focus was mainly on the training stage. However, the high dependency on the physical properties of power systems affects the efficiency of this modification, certainly because physical properties are not always reliable due to different reasons, including aging and environmental conditions. In addition, integrating all the physical properties makes training processes computationally very expensive. The completion idea and filtering technique have also been employed in a few studies (e.g., [37]), where the focus is mainly on the post-training stage. Yet, similar to pre-training stage modifications, this approach has limitations as it is detached from the training stage.

Among various methods, modified loss functions have been widely used and shown significant promise in enhancing the performance and reliability of NNs for PF analysis (e.g., [29]) and OPF (e.g., [34]). Physics-based loss functions have shown significant promise in enhancing the performance and reliability of NNs for PF and OPF. However, most existing contributions (e.g., [35]) rely on the full integration of system topology data and physical parameters into the learning process. Although this strategy improves physical consistency, it significantly increases computational complexity, thereby limiting scalability. In addition, it often assumes complete and accurate system information, which can be unrealistic in practice due to measurement noise and topology uncertainties. Consequently, developing a physics-based loss function that retains the benefits of physical consistency while reducing dependency on full system topology data and computational overhead remains an open research challenge in the field.

Therefore, this study introduces a novel hidden function derived from the power balance equations, which partially encodes line characteristics into a modified loss function as prior physical knowledge. In contrast to existing approaches that require full system topology data

and complete line parameters, the proposed approach reduces dependency on such information while preserving physical interpretability. Building upon this concept, we propose an end-to-end deep learning architecture, hereafter called physics-informed neural network for power flow (PINN4PF). The focus is on the training stage and extends the existing body of knowledge on physics-informed NNs for PF analysis (e.g., [23,29]). PINN4PF learns the mapping from net active (p) and reactive (q) power injections to complex bus voltages. An adaptive activation function is also employed to enhance the representational flexibility of the NN. The effectiveness of PINN4PF is demonstrated on 4-bus [39], 15-bus [40], 290-bus [41], and 2224-bus [42] test systems. The main contributions are:

1. A double-head feed-forward NN architecture for PF analysis, where the two heads predict the real (μ) and imaginary (ω) parts of the complex bus voltages. Each head employs a modified ReLU activation function with a trainable slope, enabling adaptive nonlinear feature learning through backpropagation.
2. A physics-based loss function that incorporates a hidden function encoding partial topology data through the diagonal elements of the admittance matrix. This design enforces physical consistency while significantly reducing the need for full topology data and lowering computational cost relative to prior fully physics-informed learning approaches.

2. Power flow analysis

PF analysis aims to specify the state variables of power systems, i.e., $[\delta\, v]^T$, where δ and v denote the voltage phase angle and magnitude, respectively. It is a representation of Kirchhoff's laws and is formulated in rectangular coordinates [29] as:

$$p_i = \sum_{j=1}^{n} g_{ij}(\mu_i\mu_j + \omega_i\omega_j) + b_{ij}(\omega_i\mu_j - \mu_i\omega_j), \tag{1a}$$

$$q_i = \sum_{j=1}^{n} g_{ij}(\omega_i\mu_j - \mu_i\omega_j) - b_{ij}(\mu_i\mu_j + \omega_i\omega_j), \tag{1b}$$

where i and j are the indices of the buses, n the total number of buses in the power system, p_i and q_i the net active and reactive power injections at bus i, g_{ij} and b_{ij} the real and imaginary components of the admittance y_{ij} between buses i and j, $\mu_i = v_i cos\delta_i$ and $\omega_i = v_i sin\delta_i$ the real and imaginary components of the voltage phasor at bus i.

There are three types of buses in power systems: (i) reference bus, (ii) load bus (pq bus), and (iii) generation bus (pv bus). v_i and δ_i are known, while p_i and q_i are unknown for the reference bus. For load buses, p_i and q_i are known, while v_i and δ_i are unknown. v_i and p_i are known for generation buses, while q_i and δ_i are unknown.

This study considers cases with one reference bus and load buses for simplicity. It should also be noted that while a voltage magnitude of 1.0 p.u. and a phase angle of 0 degrees are assumed in this study, PINN4PF is flexible to accommodate different reference bus settings.

For a power system consisting of a reference bus and load buses, a set of PF equations with the same number of equations and unknowns is achieved [43]:

$$p_i^d - p_i = 0, \tag{2a}$$

$$q_i^d - q_i = 0, \tag{2b}$$

where p_i and q_i are defined by (1a) and (1b), respectively, and p_i^d and q_i^d are the net active and reactive power injections at bus i, respectively. (2) is conventionally solved iteratively to specify $[\delta\, v]^T$ until a convergence criterion is met, i.e., the mismatch between p_i and p_i^d and also q_i and q_i^d is small enough.

3. Deep learning approaches for power flow analysis

Deep learning approaches for PF analysis refers to developing NNs to approximate the state variables of power systems based on given historical system operation data, hereafter called dataset. The dataset includes input features $\vec{x}$ and output labels $\vec{y}$. For a power system with a reference bus and load buses, the input features are known variables, i.e., the net active and reactive power injections at load buses $\vec{x} = \{(\vec{p}_i^d, \vec{q}_i^d) : i = 1, 2, \dots, n\}$. The output labels are unknown variables, i.e., the real and imaginary components of complex voltages at load buses $\vec{y} = \{(\vec{\mu}_i, \vec{\omega}_i) : i = 1, 2, \dots, n\}$. The dimension of $\vec{x}$ and $\vec{y}$ is therefore $n \times 2$, where n is the number of load buses. Note that the voltage magnitude and phase angle are known for the reference bus $i = 0$, and the net active and reactive power injections are unknown. Having voltages at all load buses approximated, the net active and reactive power injections at the reference bus can be calculated.

The training of NNs is an iterative process and involves four steps. In the first iteration, the set of trainable parameters of the NN, i.e., the weight matrices and bias vectors $\theta = \{(W_k, b_k) : k = 1, 2, \dots, m\}$, are initialized in the *update* step, where m is the number of hidden layers. NN is developed in the *forward* step. Detailed information about this step is provided in Section 4.1. The deviations of the approximated output obtained by the NN $\hat{\vec{y}}$ from the output labels $\vec{y}$ are computed in the *loss* step. Finally, the gradient of the deviations is calculated with respect to θ in the *backward* step. For the next iteration, θ is fine-tuned in the *update* step to reduce the deviations. The process continues until the maximum number of epochs is reached. In practice, the four steps can be individually and jointly modified to improve the overall performance of NNs. For example, this can be achieved by enhancing the NN architecture in the *forward* step or by adding a physical penalty term to the loss function in the *loss* step.

4. Proposed architecture: PINN4PF

The proposed PINN4PF includes two key modeling innovations in the *forward* and *loss* steps, respectively, denoted as A and B in Fig. 1. These components together form a double-head architecture enhanced with an adaptive activation function and a physics-based loss function. Note that classical PF solvers, e.g., the NR method, and PINN4PF can be interchangeably used to specify the state of the power system. The following subsections provide a detailed description of the scientific modeling and mathematical formulation of PINN4PF.

4.1. Forward pass

A double-head feed-forward NN $f(\cdot) \in \{(f_0, f_1, f_2)\}$ is developed to approximate $\hat{\vec{y}} = \{(\hat{\vec{\mu}}_i, \hat{\vec{\omega}}_i) : i = 1, 2, \dots, n\}$ at load buses using the dataset $\{\vec{x}, \vec{y}\}$. It has three types of layers, i.e., input, hidden, and output layers, as highlighted in Fig. 1-A. The input and output layers correspond to the input features $\vec{x}$ and output labels $\vec{y}$, respectively. The neurons of the input layer contain active power $\vec{p}^d = [p_1, p_2, \dots, p_n]$ followed by reactive power $\vec{q}^d = [q_1, q_2, \dots, q_n]$ at all load buses, and hence, the input layer has $n \times 2$ neurons. Following the input layer, there is a set of shared hidden layers $f_0(\cdot)$ acting as a feature extractor that projects $\vec{x}$ to a higher-dimensional space, where the two heads, $f_1(\cdot)$ and $f_2(\cdot)$, separately involve a few hidden layers to respectively approximate $\vec{\mu} = [\mu_1, \mu_2, \dots, \mu_n]$ and $\vec{\omega} = [\omega_1, \omega_2, \dots, \omega_n]$. Thus, the output layer of each head has n neurons. The shared set of hidden layers and the two heads each are a chain of functions and can be represented as:

$$f_0(\vec{x}) = l_m^0 \circ \dots \circ l_1^0(\vec{x}), \tag{3a}$$

$$f_1(\vec{x}) = l_m^1 \circ \dots \circ l_1^1(\vec{x}), \tag{3b}$$

$$f_2(\vec{x}) = l_m^2 \circ \dots \circ l_1^2(\vec{x}), \tag{3c}$$

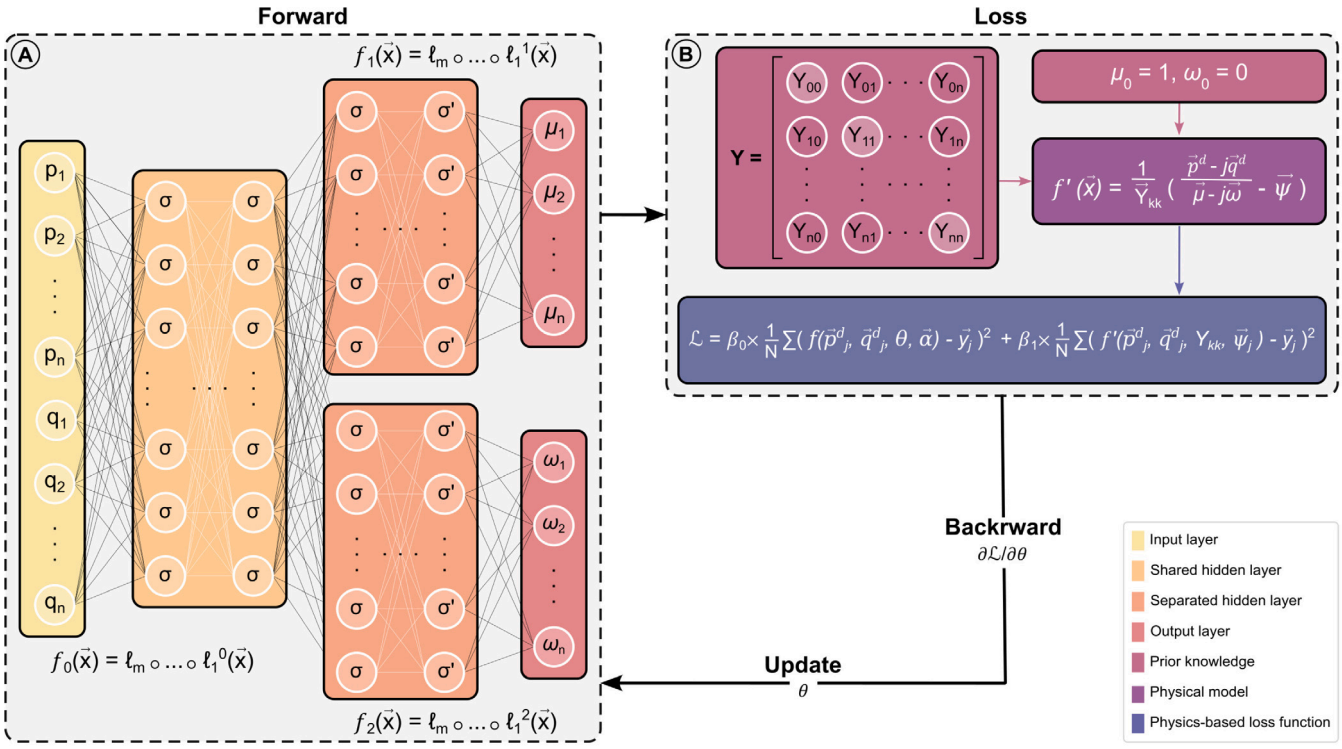


Fig. 1. Schematic diagram of PINN4PF. The model incorporates two key advancements: (A) a double-head feed-forward NN, $f(\cdot) \in \{f_0, f_1, f_2\}$, utilizing an adaptive activation function $\sigma'(z) = \max(0, \alpha z)$ with a trainable slope α; and (B) a physics-based loss function that combines supervised learning with physical constraints. The input layer receives a concatenated vector of active ($\vec{p}^d$) and reactive ($\vec{q}^d$) power at all n load buses, resulting in $2n$ input neurons. Shared hidden layers $f_0(\cdot)$ act as a feature extractor, projecting the input to a higher-dimensional latent space. Two separate network heads, $f_1(\cdot)$ and $f_2(\cdot)$, then predict the real ($\vec{\mu}$) and imaginary ($\vec{\omega}$) components of the complex bus voltages, each with n output neurons. The model is trained by minimizing a composite loss that enforces both data fidelity (using labels) and physical consistency.

which sequentially process $\vec{x}$ through m hidden layers to obtain $\vec{f}_0(\cdot)$, $\vec{f}_1(\cdot)$, and $\vec{f}_2(\cdot)$, respectively. For the shared set of hidden layers and the two heads, the kth hidden layer ($k = 1, 2, \ldots, m$) is given by:

$$l_k(\vec{x}) = \sigma\left(W_k^T \cdot \vec{x} + b_k\right). \tag{4}$$

Here, W_k^T and b_k are the weight matrix and bias vector for the corresponding hidden layer. Each hidden layer applies a linear transformation $W_k^T \cdot \vec{x} + b_k$, followed by a nonlinear transformation $\sigma(\cdot)$ to capture complex relationships between $\vec{x}$ and $\vec{y}$. Note that W_k and b_k are trainable parameters and are optimized during the training process.

The Rectified Linear Unit (ReLU) activation function $\sigma(\cdot)$ is applied to the shared set of hidden layers (3a). While an adaptive version of ReLU $\sigma'(\cdot)$ is developed to apply model-based nonlinear transformations to the two heads (3b) and (3c). That is, a trainable parameter α is introduced to scale the result of the linear transformation while applying nonlinearity. Hence, α serves as a guide to prevent overfitting and improve the generalization ability of the NN against unseen data. $\sigma(\cdot)$ and $\sigma'(\cdot)$ are represented as:

$$\sigma = \max(0, z), \tag{5a}$$

$$\sigma' = \max(0, \alpha z), \tag{5b}$$

where z is equivalent to $W_k^T \cdot \vec{x} + b_k$. During each iteration of the training process, the gradients of the loss function concerning the trainable parameters $\nabla_{W_k, b_k, \alpha_k} L$ are computed for each hidden layer k using the chain rule of differentiation. The resulting gradients are then backpropagated through NN to update and optimize α_k along with W_k and b_k, which collectively improve NN's performance in approximating $\hat{\vec{y}}$. More detailed information about adaptive activation functions can be found in [44,45].

4.2. Loss function

Prior physical knowledge is integrated with the NN architecture to make it informed; see Fig. 1-B. A part of the topology information of the power system, i.e., the diagonal elements of the admittance matrix Y_{kk}, is used to develop the physical model $f'(\cdot)$. In addition, the real and imaginary components of the complex voltage at the reference bus, i.e., $\mu_0 = 1, \omega_0 = 0$, are imported.

The derivation of $f'(\cdot)$ begins with Ohm's law, which relates voltage V, current I, and resistance R, that is, $V = I \times R$. By extending this equation to the complex domain, $\overline{V} = \overline{I} \times \overline{Z}$ relates the voltage phasor, current phasor, and impedance phasor. From this equation, $\overline{I}$ can be expressed in terms of $\overline{V}$ and $\overline{Y}$ (inverse of the impedance phasor), that is $\overline{I} = \overline{Y} \times \overline{V}$. The system of equations for all buses can then be presented in matrix form as:

$$[I]_{n\times 1} = [Y]_{n\times n} \times [V]_{n\times 1}. \tag{6}$$

By rearranging (6), the current flowing into bus k can be presented as a linear combination of the voltages at all other buses with weights given by the corresponding admittance matrix:

$$I_k = \sum_{i=1}^{n} Y_{ki} V_i. \tag{7}$$

Here, the power equation $S = V \times I^*$ provides further guides for subsequent derivations, that is $I = \frac{S^*}{V^*}$, where S is complex apparent power. The current and voltage phasors are related as:

$$\frac{S_k^*}{V_k^*} = Y_{k1}V_1 + Y_{k2}V_2 + \cdots + Y_{kk}V_k + \cdots + Y_{kn}V_n, \tag{8}$$

wherein the right-hand side is an extended form of the right-hand expression in (7). Finally, the expression for calculating the voltage

phasor at bus k can be written as:

$$V_k = \frac{1}{Y_{kk}} \times \left(\frac{S_k^*}{V_k^*} - \sum_{i=1, i\neq k}^{n} Y_{ki} V_i \right). \tag{9}$$

In (9), $Y_{kk} = G_{kk} + jB_{kk}$ is known from the power system topology. Considering the input features and output labels needed to develop PINN4PF, $V_k = \mu_k + j\omega_k$ and $V_k^* = \mu_k - j\omega_k$ are known from the output labels, and $S_k^* = p_k^d + jq_k^d$ is known from the input features. The only unknown expression remaining in (9) is:

$$\psi_k = \sum_{i=1, i\neq k}^{n} Y_{ki} V_i, \tag{10}$$

which is called *hidden function* in this study. Note that ψ is unique for each data point. Theoretically, ψ is also unique for each bus. During the training process, PINN4PF first approximates ψ and then uses it to approximate $V = \mu + j\omega$ using:

$$f'(\vec{x}) = \frac{1}{Y_{kk}} \times \left(\frac{p^d - jq^d}{\mu - j\omega} - \psi \right). \tag{11}$$

This unique relation improves the learning process by following the underlying physical laws of the power system. Instead of integrating the full admittance matrix $[Y]_{n\times n}$ into the physics-based loss function, only the diagonal elements Y_{kk} are needed. This simplification reduces the wall-clock training time by up to three orders of magnitude compared to the full admittance matrix integration used in [46]. Note that this comparison refers to NN training with physics-based loss functions. While numerical solvers already exploit the sparsity of the admittance matrix, integrating the full matrix in NN training remains computationally expensive. By using only the diagonal elements, PINN4PF preserves key physical consistency while drastically reducing computational requirements. It is also important to note that the diagonal entries represent each bus's self-admittance, which dominates local voltage-current coupling and implicitly accounts for the aggregate effect of connected branches. This allows PINN4PF to retain essential physical constraints without requiring full system topology information.

Typically, the goal of the training process is to minimize the difference between $\hat{\vec{y}}$, approximated by NN, and ground-truth output labels $\vec{y}$, obtained from NR, based on a loss function of choice. In this study, a modified loss function is developed that combines supervised and physics-based penalty terms. The supervised term enforces agreement with ground-truth data, while the physics-based term ensures local compliance with Ohm's and power balance laws. This hybrid formulation enhances learning stability and generalization, particularly under data-limited or uncertain operating conditions (e.g., [47]). The former term is the mean square of the difference between the output approximated by the NN $f(\cdot)$ and $\vec{y}$. Whereas the latter is the mean square of the difference between the output obtained from the physical model $f'(\cdot)$ and $\vec{y}$, as:

$$\begin{aligned} \mathscr{L} = {} & \beta_0 \times \underbrace{\frac{1}{N} \sum_{j=1}^{N} \left(f(\vec{p}_j^d, \vec{q}_j^d, \theta, \vec{\alpha}) - \vec{y}_j \right)^2}_{\text{Supervised penalty term}} \\ & + \beta_1 \times \underbrace{\frac{1}{N} \sum_{j=1}^{N} \left(f'(\vec{p}_j^d, \vec{q}_j^d, Y_{kk}, \vec{\psi}_j) - \vec{y}_j \right)^2}_{\text{Physical penalty term}}, \end{aligned} \tag{12}$$

where N is the total number of data points j. $f(\cdot) \in \{(f_0, f_1, f_2)\}$ and $f'(\cdot)$ are the NN architecture (3a)–(3c) and the physical model (11), respectively. β_0 and β_1 represent the coefficients for the supervised and physical penalty terms, respectively, with the constraint that $\beta_0 + \beta_1 = 1$. Initially, $\beta_0 = 1$ and $\beta_1 = 0$. After 100 epochs, the value of β_1 increases at a specified rate, with its maximum value determined from the sensitivity analysis.

5. Results

The performance of PINN4PF is evaluated against LR and MLP. Experiments are performed on 4-bus [39], 15-bus [40], 290-bus [41], and 2224-bus [42] test systems containing one reference bus for which the voltage in complex number form $\mu_0 + j\omega_0$ is known while p_0^d and q_0^d are unknown, and load buses i for which p_i^d and q_i^d are known, while complex voltages $\mu_i + j\omega_i$ are unknown.

Input features and output labels of the datasets are represented as $\vec{x}$ and $\vec{y}$, respectively, where $\vec{x}$ includes $(\vec{p}_i^d, \vec{q}_i^d)$ and $\vec{y}$ includes $(\vec{\mu}_i, \vec{\omega}_i)$ obtained from the NR method for all load buses i. The PandaPower Python package [48] is used to perform NR and generate the datasets. LR, MLP, and PINN4PF are compared in terms of their ability to approximate NR results efficiently and accurately, which is a common practice in the literature (e.g., [30]). Note that PandaPower specifies the state variables of power systems, i.e., $[\delta\ v]^T$, that is, there is a need for converting the state variables to $\mu_i = v_i cos\delta_i$ and $\omega_i = v_i sin\delta_i$ to yield the output labels $\vec{y} = \{(\vec{\mu}_i, \vec{\omega}_i) : i = 1, 2, \ldots, n\}$.

5.1. Model setup

A systematic approach is employed to generate the datasets. Considering $\vec{p}^d$ and $\vec{q}^d$ known for a specific scenario of the test system, $s_i^d = \sqrt{(p_i^d)^2 + (q_i^d)^2}$, and $pf_i = p_i^d / s_i^d$ are computed for each bus i. Here, s_i^d is the mean, and a deviation of 30% from s_i^d is the standard deviation to develop a normal distribution of size 5000 for each bus i, i.e., $S_i \sim \mathcal{N}(s_i^d, 0.3)$. For $S_i \in \{s_{ij}^d : j = 1, 2, \ldots, 5000\}$, $p_{ij}^d = s_{ij}^d \times pf_i$ and $q_{ij}^d = \sqrt{(s_{ij}^d)^2 - (p_{ij}^d)^2}$ are then computed for all buses i and all samples j. This approach yields a pool of 5000 scenarios from which the data points are randomly selected to form the dataset. The datasets contain different numbers of data points: 256, 512, 1024, and 2048 for the 4-bus, 15-bus, 290-bus, and 2224-bus test systems, respectively. The selected data points are sent to PandaPower to perform PF analysis. Each dataset is then split into three subsets: 40% for training, 20% for validation, and the remaining 40% for testing.

Note that the number of data points is deliberately limited to highlight the ability of PINN4PF to learn effectively even with a relatively small training dataset by embedding prior physical knowledge directly into the training process and leveraging adaptive activation functions. This approach is consistent with practices reported in the literature, where physics-informed learning has been employed as a strategy to reduce the amount of training data required (e.g., [49]).

The proposed end-to-end architecture is developed in Python. All components, including model definition, training loops, and evaluation routines, are implemented within the PyTorch environment, leveraging its built-in automatic differentiation and GPU acceleration capabilities. The training process ends after 5000 epochs for PINN4PF, MLP, and LR. The loss function used is (12), with $\beta_0 = 1$ and $\beta_1 = 0$ for MLP and LR, indicating a supervised penalty term. However, β_0 and β_1 are non-zero for PINN4PF, where $\beta_0 + \beta_1 = 1$, indicating a combination of supervised and physical penalty terms. The activation function is ReLU (5a) for MLP and the shared hidden layers of PINN4PF. The adaptive ReLU (5b) is used for the separated hidden layers of PINN4PF. The Adam optimization algorithm updates the trainable parameters during the training process for PINN4PF, MLP, and LR.

5.2. Model performance

The performance of PINN4PF, MLP, and LR is systematically evaluated based on (i) generalization ability, (ii) robustness, (iii) impact of training dataset size on generalization ability, (iv) accuracy in approximating derived PF quantities, and (v) scalability. Experiments are done using the 15-bus test system. Additional experiments are also performed using 4-bus, 290-bus, and 2224-bus test systems for scalability.

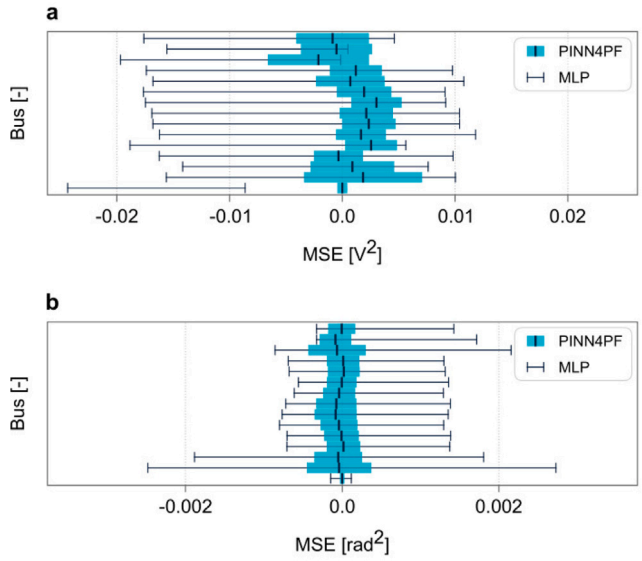


Fig. 2. Mean and standard deviation of MSE for direct physical quantities: (a) voltage magnitude $[V^2]$ and (b) voltage phase angle $[rad^2]$. The results are obtained by PINN4PF and MLP based on the testing dataset for the 15-bus test system.

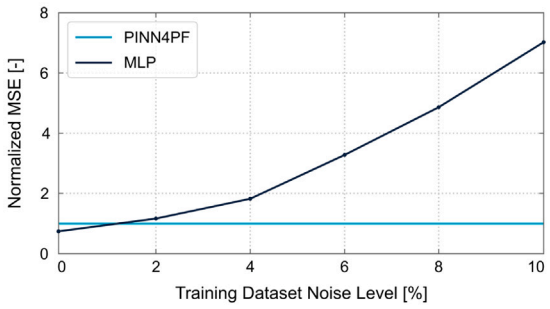


Fig. 3. Illustration of the performance of PINN4PF and MLP based on the MSE obtained for the voltage magnitude $[V^2]$ under varying noise levels in the training dataset for the 15-bus test system. The MSE values are normalized relative to that of PINN4PF at the 10% noise level.

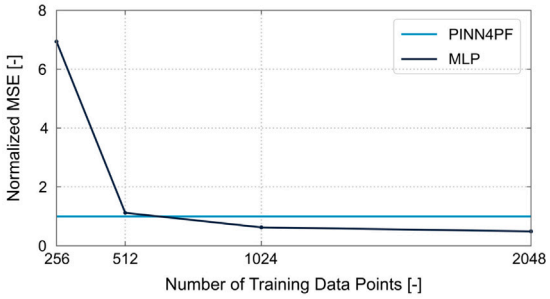


Fig. 4. Illustration of the performance of PINN4PF and MLP based on the MSE obtained for the voltage magnitude $[V^2]$ for the 15-bus test system under different training dataset sizes. The MSE values are normalized relative to that of PINN4PF with 256 training data points.

5.2.1. Generalization ability

The mean and standard deviation of the mean squared error (MSE) for the test dataset obtained by PINN4PF and MLP are compared in Fig. 2. PINN4PF is observed to achieve up to 85% and 65% lower maximum testing MSE for the voltage magnitude $[V^2]$ and voltage phase angle $[rad^2]$, respectively, compared to MLP.

5.2.2. Robustness

The robustness of PINN4PF, MLP, and LR is evaluated using the 15-bus test system under varying noise levels. Controlled uniform random noise is added to both the input features and output labels of the training dataset. This approach aligns with the use of bounded or non-Gaussian noise models commonly adopted in power system studies (e.g., [50]). The corrupted vectors are defined as $\vec{x}' = \vec{x} \pm \vec{r}_x$ and $\vec{y}' = \vec{y} \pm \vec{r}_y$, where $\vec{r}_x$ and $\vec{r}_y$ are vectors sampled from uniform distributions within the ranges $[0, 1]$ and $[0, 0.1]$, respectively. The larger perturbation range for the input features reflects higher uncertainty typically associated with active and reactive power measurements, while a smaller range is applied to the output labels to maintain physically meaningful voltage responses.

Fig. 3 illustrates the changes in the MSE for the voltage magnitude $[V^2]$ based on the testing dataset with varying noise levels obtained by MLP. The comparison is made relative to a constant curve representing the MSE obtained by PINN4PF using the highest noise level, i.e., 10%. At the 0% noise level, MLP outperforms PINN4PF trained with 10% noisy data by approximately 18%. However, as the noise level increases, the performance of MLP deteriorates significantly, with MSE increasing up to six times. LR is excluded from the graph as its MSE at different noise levels is two orders of magnitude higher than that of PINN4PF. In addition, LR shows limited improvement with increasing dataset sizes and consistently underperforms compared to both PINN4PF and MLP.

5.2.3. Training dataset size

The impact of the size of the training dataset on the performance of MLP is investigated using the 15-bus test system. Fig. 4 illustrates the changes in the MSE for the voltage magnitude $[V^2]$ based on the testing dataset with varying training dataset sizes. The comparison is made relative to a constant curve representing the MSE obtained for PINN4PF using 256 training data points. It is observed that MLP requires a training dataset twice as large as that of PINN4PF to achieve a still inferior performance. However, the performance of MLP improves with more training data, reaching a comparable level with a dataset four times larger than that of PINN4PF. This indicates that PINN4PF is more data-efficient. LR is not included in the graph as its MSE with different training dataset sizes is two orders of magnitude larger than that of PINN4PF. In addition, LR shows limited improvement with increasing training dataset sizes and consistently performs worse than PINN4PF and MLP. This indicates that the capacity of LR to capture complex relationships in PF analysis is significantly lower, and increasing the amount of training data is insufficient to bridge the performance gap.

5.2.4. Accuracy of derived power flow quantities

For derived physical quantities, i.e., line current, line active power, and line reactive power, the mean and standard deviation of the testing MSE obtained by PINN4PF and MLP are computed and compared in Fig. 5. It is observed that PINN4PF achieves up to 81%, 63%, and 66% lower maximum testing MSE for the line current $[A^2]$, line active power $[W^2]$, and line reactive power $[VAR^2]$, respectively, compared to MLP. Note that, while the testing MSE for direct quantities (voltage magnitude and phase) shows lower differences between PINN4PF and MLP, PINN4PF achieves substantially better accuracy for derived physical quantities. This improvement arises because the physics-based loss function enforces the underlying algebraic and system constraints that govern how these derived variables are computed from the predicted voltages. Consequently, error propagation to derived quantities is reduced. Such behavior, i.e., enhanced internal consistency and improved generalization for derived variables under physics-informed learning, has been observed in recent studies in power systems and related domains (e.g., [51]).

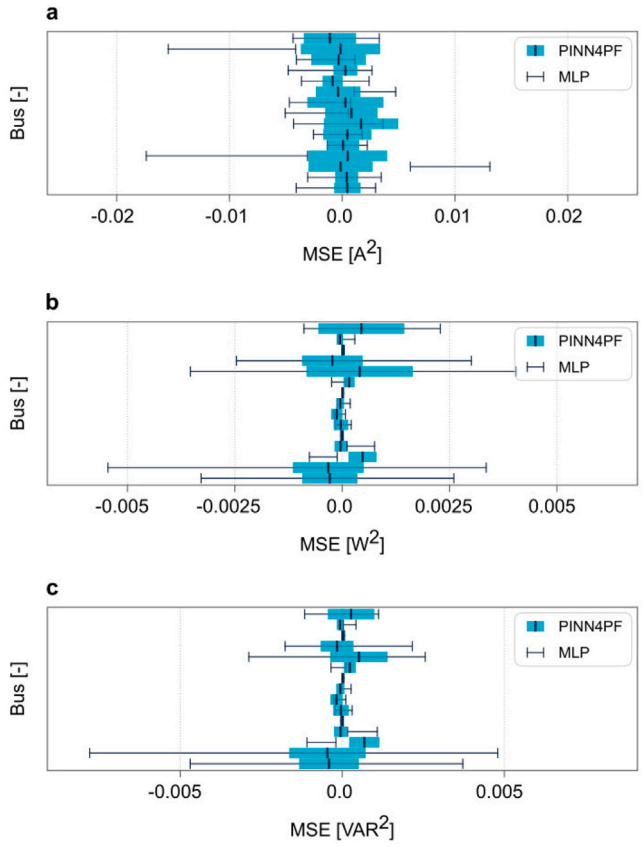


Fig. 5. Mean and standard deviation of MSE for derived physical quantities: (a) line current $[A^2]$, (b) line active power $[W^2]$, and (c) line reactive power $[VAR^2]$. The results are obtained by PINN4PF and MLP based on the testing dataset for the 15-bus test system.

5.2.5. Scalability

The experiments are extended to compare the performance of PINN4PF, MLP, and LR across different test system sizes: 4-bus, 15-bus, 290-bus, and 2224-bus test systems. The maximum testing MSE for the direct physical quantities obtained for all test system sizes is presented in Table 2. Accordingly, PINN4PF significantly outperforms MLP and LR. To further highlight the performance of PINN4PF under extreme conditions with limited training data, we consider a large-scale test system operating near its loadability limits, with a few bus voltages reaching approximately 0.85 *p.u.* and exceeding 1.05 *p.u.*. The direct physical quantities for the 2224-bus test system are depicted in Fig. 6. PINN4PF achieves a maximum MSE $[V^2]$ 50% lower than MLP for this test system. This improvement is crucial for power system operations, particularly during unexpected events that cause deviations in voltages from the nominal value.

The analysis also considers the performance of PINN4PF, MLP, and LR in approximating derived physical quantities. The maximum testing MSE for all test system sizes is presented in Table 3. The table demonstrates the superiority of PINN4PF over MLP and LR across all test system sizes. For the 2224-bus test system, for example, the maximum MSE $[A^2]$ obtained by PINN4PF is respectively 92% and 98% lower than MLP and LR.

In addition, as the size of the test system increases, the performance gap between PINN4PF, MLP, and LR becomes more pronounced. Fig. 7 compares the performance of PINN4PF and MLP under extreme conditions in the 2224-bus test system for the derived physical quantity. It should be noted that while the testing MSE for the direct quantities, i.e., voltage magnitude and phase angle, shows marginal differences between PINN4PF and MLP, PINN4PF significantly outperforms MLP in terms of the derived physical quantities, such as line current.

Table 2
Performance comparison of power flow solvers based on MSE obtained for direct physical quantities.

Case	Model	MSE $[V^2]$	MSE $[rad^2]$
4-bus	PINN4PF	4.85×10^{-4}	3.81×10^{-8}
	MLP	5.70×10^{-3}	3.72×10^{-7}
	LR	2.52×10^{-2}	4.70×10^{-6}
15-bus	PINN4PF	5.73×10^{-6}	1.06×10^{-7}
	MLP	3.96×10^{-5}	3.09×10^{-7}
	LR	6.32×10^{-4}	1.57×10^{-5}
290-bus	PINN4PF	9.54×10^{-10}	1.65×10^{-8}
	MLP	3.03×10^{-9}	1.78×10^{-8}
	LR	1.03×10^{-7}	2.49×10^{-8}
2224-bus	PINN4PF	1.03×10^{-4}	8.15×10^{-5}
	MLP	2.07×10^{-4}	7.66×10^{-5}
	LR	3.62×10^{0}	2.93×10^{-4}

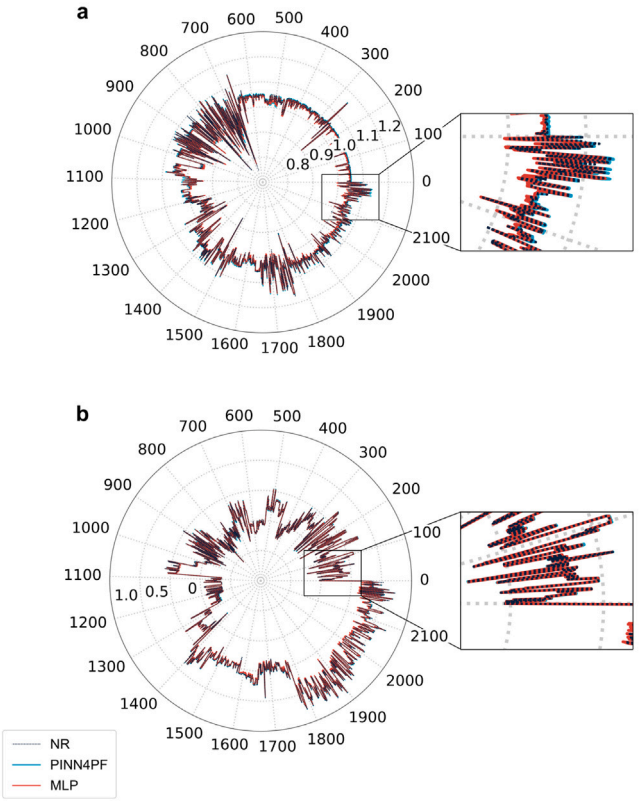


Fig. 6. Comparison of (a) voltage magnitude $\vec{v}$ and (b) voltage phase angle $\vec{\delta}$ obtained from NR, PINN4PF, and MLP for the 2224-bus test system. The bus indices are positioned along the outer circle, while the corresponding quantities are plotted along the radial direction.

5.3. Model sensitivity analysis

The architecture of PINN4PF and MLP, including the number of hidden layers and neurons per hidden layer, as well as hyperparameters, including the learning rate, the weight decay rate, and the percentage of dropout, are determined by sensitivity analysis to ensure a fair comparison[1]. The maximum weight for the share of the supervised penalty term (β_0) in (12) is also fine-tuned. It should be noted that for MLP, $\beta_0 = 1$ and $\beta_1 = 0$. Sensitivity analysis is not performed for LR since it involves linear combinations. The optimal configurations are identified based on minimizing both training and testing MSE, while

[1] An exhaustive search across 2000 unique hyperparameter combinations is conducted using the Optuna Python package [52].

Table 3
Performance comparison of power flow solvers based on MSE obtained for derived physical quantities.

Case	Model	MSE $[A^2]$	MSE $[W^2]$	MSE $[VAR^2]$
4-bus	PINN4PF	2.43×10^{-6}	4.25×10^{-4}	1.06×10^{-4}
	MLP	2.80×10^{-5}	8.27×10^{-3}	2.06×10^{-3}
	LR	4.04×10^{-5}	1.34×10^{-2}	3.36×10^{-2}
15-bus	PINN4PF	4.35×10^{-6}	2.09×10^{-7}	2.17×10^{-7}
	MLP	2.39×10^{-5}	5.77×10^{-7}	6.42×10^{-7}
	LR	1.10×10^{-2}	8.70×10^{-4}	1.74×10^{-3}
290-bus	PINN4PF	3.84×10^{-7}	2.01×10^{-13}	4.77×10^{-14}
	MLP	2.85×10^{-6}	1.10×10^{-12}	2.44×10^{-13}
	LR	1.36×10^{-5}	8.20×10^{-12}	1.84×10^{-12}
2224-bus	PINN4PF	1.15×10^{-6}	1.13×10^{-8}	9.45×10^{-7}
	MLP	3.69×10^{-5}	7.22×10^{-8}	5.12×10^{-5}
	LR	7.85×10^{-1}	3.55×10^{-1}	2.44×10^{2}

Table 4
Performance comparison of different configurations of PINN4PF based on MSE obtained for direct physical quantities and the 15-bus test system.

Model	MSE $[V^2]$	MSE $[rad^2]$
ReLU&Supervised	4.05×10^{-5}	3.24×10^{-7}
AdaptiveReLU&Supervised	1.42×10^{-5}	3.73×10^{-7}
ReLU&Physical	6.31×10^{-6}	4.70×10^{-7}
AdaptiveReLU&Physical	5.73×10^{-6}	1.06×10^{-7}

also ensuring that they remain close enough to indicate generalization ability. All experiments use the 15-bus test system and a noisy dataset comprising 512 data points.

5.3.1. Sensitivity analysis for MLP

We explore eight configurations, intentionally avoiding deeper NNs with more than eight hidden layers due to the limited size of both the dataset and the test system. Each configuration comprises input and output layers defined by $n \times 2$ neurons in the first and last hidden layers, respectively, while intermediate layers are structured with $n \times 4/3$ neurons. We observe that when the number of hidden layers is fewer than seven, a reduction in layer count correlates with a decrease in generalization ability. We examine learning rates ranging from 1×10^{-1} to 1×10^{-10} and weight decay rates within the same span. Dropout rates are tested between 0% and 2%. Sensitivity analysis results indicate that the selected configuration for MLP involves seven hidden layers and is trained with a learning rate of 2.3×10^{-4}, a weight decay of 1.8×10^{-5}, a dropout percentage of 0.2%, and a batch size of 16. For this configuration, the training and testing MSE achieved are $3.74 \times 10^{-5}[V^2]$ and $3.96 \times 10^{-5}[V^2]$, respectively.

5.3.2. Sensitivity analysis for PINN4PF

We systematically assess the influence of the number of hidden layers, ranging from one to four, within the shared set of hidden layers and the individual heads. We evaluate distinct configurations where the shared hidden layers contain $n \times 2$ neurons, and each head comprises n neurons. Learning and weight decay rates are varied from 1×10^{-1} to 1×10^{-10}, and dropout rates are varied from 0% to 2%. Sensitivity analysis reveals that the selected configuration for PINN4PF consists of two shared hidden layers alongside four hidden layers per head. In addition, PINN4PF is trained with a learning rate of 1.3×10^{-4}, a weight decay of 1.1×10^{-5}, a dropout rate of 0.1%, and a batch size of 16. The maximum weight for the share of the supervised penalty term is $\beta_1 = 0.71$. For this configuration, the achieved training and testing MSE are $5.64 \times 10^{-6}[V^2]$ and $5.73 \times 10^{-6}[V^2]$, respectively.

5.3.3. Impact of modifications on the performance of PINN4PF

We evaluate the following configurations: (i) a double-head feed-forward NN with ReLU (5a) and supervised penalty term (ReLU&Supervised), (ii) a double-head feed-forward NN with adaptive ReLU (5b) and supervised penalty term (AdaptiveReLU&Supervised), (iii) a double-head feed-forward NN with ReLU (5a) and physical penalty term (ReLU&Physical), and (iv) a double-head feed-forward NN with adaptive ReLU (5b) and physical penalty term (AdaptiveReLU&Physical). Table 4 presents a comparative analysis of their performance in terms of direct physical quantities. The results show that AdaptiveReLU&Physical yields superior performance for PINN4PF compared to the other combinations.

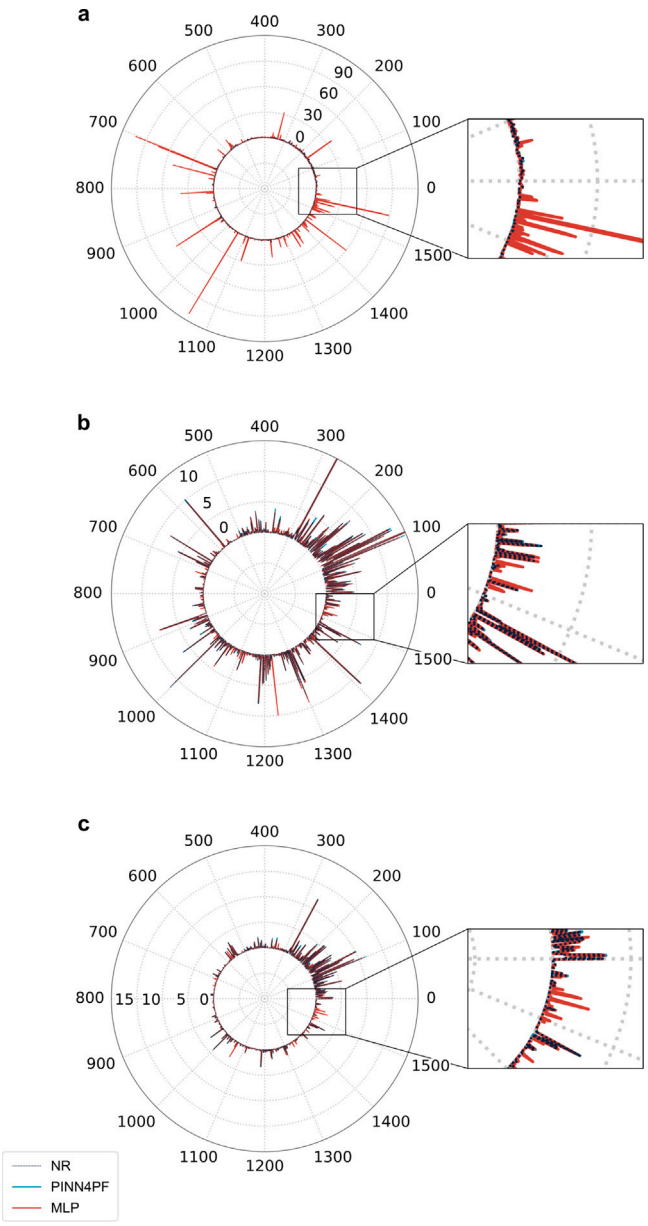


Fig. 7. Comparison of (a) line current $\vec{i}$, (b) line active power $\vec{p}_l$, and (c) line reactive power $\vec{q}_l$ obtained from NR, PINN4PF, and MLP for the 2224-bus test system. The bus indices are positioned along the outer circle, while the corresponding quantities are plotted along the radial direction.

6. Discussion

Several key points and observations arise from the findings:

- The optimization algorithm suggests that for larger test systems, the share of the supervised penalty term (β_0) in the loss function should be significantly smaller than the physical penalty term (β_1), which indicates the increasing importance of physical constraints in larger test systems.
- Although only the diagonal elements of the admittance matrix are used to develop the loss function, including a physical penalty term imposes additional calculations during training, which makes PINN4PF computationally more expensive than black-box NNs and highlights a trade-off between accuracy and computational cost.
- The datasets are simulated rather than derived from field measurements, which aligns with standard practice in power system studies (e.g., [30,32]), given the limited availability of large-scale, labeled operational data. Future work can focus on validating PINN4PF using real-world data.
- PINN4PF is trained on different operating conditions, and potentially can generalize to provide approximate solutions in ill-conditioned scenarios, where traditional methods fail to converge. Future work can explore the use of these solutions as warm starts for traditional methods and assess their potential to enhance convergence in challenging cases.
- In modern power systems, missing or noisy measurements and incomplete topology data are common, often causing traditional PF solvers to fail. PINN4PF addresses these challenges by integrating partial topology information, specifically, the diagonal elements of the admittance matrix. This design enables PINN4PF to learn complex system dynamics from limited data and enhances robustness against noise or incompleteness in both training samples and topology information.
- While PINN4PF demonstrates strong performance under fixed network topology, it requires retraining if the system topology changes. Future research can explore the potential of transfer learning to adapt trained models to new topologies with minimum additional training.
- The physics-based loss function acts as a training penalty to enforce power balance consistency and enhance the accuracy of bus voltage predictions, whereas actual power system losses are computed exactly from the predicted/calculated voltages using standard electrical formulations.

7. Conclusion

An end-to-end architecture called PINN4PF is introduced. It includes two important advancements in the training pipeline: (A) a double-head feed-forward NN that aligns with PF, including an activation function that adjusts to the net active and reactive power injection patterns, and (B) a physics-based loss function that partially incorporates power system topology information through a novel hidden function. PINN4PF, therefore, offers a straightforward yet effective approach to capturing the complexities inherent in large-scale modern power systems. The application of PINN4PF to four test power systems, including 4-bus, 15-bus, 290-bus, and 2224-bus systems, evaluates its performance against LR and MLP.

The results highlight PINN4PF's generalization ability, robustness against noise, data efficiency, and scalability to large-scale power systems. Specifically, PINN4PF consistently achieves lower MSE for direct and derived physical quantities, including line current, line active power, and line reactive power. In addition, as the test system size increases, the performance gap between PINN4PF, MLP, and LR becomes more pronounced. This advancement is crucial for enhancing power system operations, especially under extreme conditions and unexpected deviations, making PINN4PF a promising solution for future PF analysis and optimization tasks.

CRediT authorship contribution statement

Zeynab Kaseb: Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Stavros Orfanoudakis:** Writing – review & editing, Validation, Software, Investigation. **Pedro P. Vergara:** Writing – review & editing, Validation, Supervision, Resources, Project administration, Conceptualization. **Peter Palensky:** Validation, Supervision, Resources.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is part of the DATALESs project (project no. 482.20.602), which is jointly financed by Netherlands Organization for Scientific Research (NWO), and National Natural Science Foundation of China (NSFC). The work used the Dutch national e-infrastructure with the support of the SURF Cooperative (grant no. EINF-6569).

Data availability

Data will be made available on request.

References

[1] Li B, Wu Q, Cao Y, Li C. Search direction optimization of power flow analysis based on physics-informed deep learning. Int J Electr Power Energy Syst 2025;167:110602.
[2] Buason P, Misra S, Watson J-P, Molzahn DK. Adaptive power flow approximations with second-order sensitivity insights. IEEE Trans Power Syst 2024;1–12.
[3] Bugosen SI, Parker RB, Coffrin C. Applications of lifted nonlinear cuts to convex relaxations of the ac power flow equations. IEEE Trans Power Syst 2024;1–4.
[4] Wang Y, Qi D, Zhang J, Mei J. Novel optimal load control for power system frequency and voltage regulation. J Mod Power Syst Clean Energy 2023;11(4):1746–55.
[5] Swirydowicz K, Koukpaizan N, Ribizel T, Göbel F, Abhyankar S, Anzt H, Peleš S. GPU-resident sparse direct linear solvers for alternating current optimal power flow analysis. Int J Electr Power Energy Syst 2024;155:109517.
[6] Dorto M, Sjoblom S, Chien LS, Axner L, Gong J. Comparing different approaches for solving large scale power-flow problems with the Newton-Raphson method. IEEE Access 2021;9:56604–15.
[7] Kaseb Z, Möller M, Vergara PP, Palensky P. Power flow analysis using quantum and digital annealers: a discrete combinatorial optimization approach. Sci Rep 2024;14(1):23216.
[8] Sharma N, Acharya A, Jacob I, Yamujala S, Gupta V, Bhakar R. Major blackouts of the decade: Underlying causes, recommendations and arising challenges. In: 2021 9th IEEE international conference on power systems. IEEE; 2021, p. 1–6.
[9] Huo Y, Chen Z, Li Q, Li Q, Yin M. Graph neural network based column generation for energy management in networked microgrid. J Mod Power Syst Clean Energy 2024;12(5):1506–19.
[10] Dziugaite GK, Roy DM. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. 2017.
[11] Wolgast T, Nieße A. Learning the optimal power flow: Environment design matters. Energy AI 2024;17:100410.
[12] Cibaku E, Gama F, Park S. Boosting efficiency in state estimation of power systems by leveraging attention mechanism. Energy AI 2024;16:100369.
[13] Beinert D, Holzhüter C, Thomas JM, Vogt S. Power flow forecasts at transmission grid nodes using graph neural networks. Energy AI 2023;14:100262.
[14] Wang Z, Menke J-H, Schäfer F, Braun M, Scheidler A. Approximating multi-purpose AC optimal power flow with reinforcement trained artificial neural network. Energy AI 2022;7:100133.
[15] Ahshan R, Abid MS, Al-Abri M. Geospatial mapping of large-scale electric power grids: A residual graph convolutional network-based approach with attention mechanism. Energy AI 2025;20:100486.
[16] Huang B, Wang J. Applications of physics-informed neural networks in power systems - a review. IEEE Trans Power Syst 2023;38(1):572–88.

[17] Kaseb Z, Möller M, Balducci GT, Palensky P, Vergara PP. Quantum neural networks for power flow analysis. Electr Power Syst Res 2024;235:110677.
[18] Oliveira RAd, Bollen MH. Deep learning for power quality. Electr Power Syst Res 2023;214:108887.
[19] von Rueden L, Mayer S, Beckh K, Georgiev B, Giesselbach S, Heese R, Kirsch B, Pfrommer J, Pick A, Ramamurthy R, Walczak M, Garcke J, Bauckhage C, Schuecker J. Informed machine learning – a taxonomy and survey of integrating knowledge into learning systems. 2019.
[20] Li H, Liu L, Yu S, He Q, Wu Q, Zhang J, Lu Q. Graph attention convolution network for power flow calculation considering grid uncertainty. Int J Electr Power Energy Syst 2025;165:110513.
[21] Liang D, Li G, Liu X, Zeng L, Chiang H-D, Wang S. Bayesian state estimation for partially observable distribution networks via power flow-informed neural networks. Int J Electr Power Energy Syst 2025;170:110886.
[22] Lei X, Yang Z, Yu J, Zhao J, Gao Q, Yu H. Data-driven optimal power flow: A physics-informed machine learning approach. IEEE Trans Power Syst 2021;36(1):346–54.
[23] Yang Y, Yang Z, Yu J, Zhang B, Zhang Y, Yu H. Fast calculation of probabilistic power flow: A model-based deep learning approach. IEEE Trans Smart Grid 2020;11(3):2235–44.
[24] Donti PL, Rolnick D, Kolter JZ. DC3: A learning method for optimization with hard constraints. 2021.
[25] Jeddi AB, Shafieezadeh A. A physics-informed graph attention-based approach for power flow analysis. In: 2021 20th IEEE international conference on machine learning and applications. IEEE; 2021, p. 1634–40.
[26] De Jongh S, Gielnik F, Mueller F, Schmit L, Suriyah M, Leibfried T. Physics-informed geometric deep learning for inference tasks in power systems. In: 22nd power systems computation conference. Porto, Portugal; 2022.
[27] Yang M, Qiu G, Wu Y, Liu J, Dai N, Shui Y, Liu K, Ding L. Physics-guided graph neural networks for real-time AC/DC power flow analysis. 2023.
[28] Kody A, Chevalier S, Chatzivasileiadis S, Molzahn D. Modeling the AC power flow equations with optimally compact neural networks: Application to unit commitment. In: 22nd power systems computation conference. 2022.
[29] Hu X, Hu H, Verma S, Zhang Z-L. Physics-guided deep neural networks for power flow analysis. IEEE Trans Power Syst 2021;36(3):2082–92.
[30] Lin N, Orfanoudakis S, Cardenas NO, Giraldo JS, Vergara PP. PowerFlowNet: Power flow approximation using message passing graph neural networks. Int J Electr Power Energy Syst 2024;160:110112.
[31] Liu L, Shi N, Wang D, Ma Z, Wang Z, Reno MJ, Azzolini JA. Voltage calculations in secondary distribution networks via physics-inspired neural network using smart meter data. IEEE Trans Smart Grid 2024. 1–1.
[32] Nellikkath R, Chatzivasileiadis S. Physics-informed neural networks for minimising worst-case violations in DC optimal power flow. 2021.
[33] Nellikkath R, Chatzivasileiadis S. Physics-informed neural networks for AC optimal power flow. Electr Power Syst Res 2022;212:108412.
[34] Hu Z, Zhang H. Optimal power flow based on physical-model-integrated neural network with worth-learning data generation. 2023.
[35] Wu Z, Zhang M, Gao S, Wu Z-G, Guan X. Physics-informed reinforcement learning for real-time optimal power flow with renewable energy resources. IEEE Trans Sustain Energy 2024;1–11.
[36] Pan X, Chen M, Zhao T, Low SH. DeepOPF: A feasibility-optimized deep neural network approach for ac optimal power flow problems. IEEE Syst J 2022;1–11.
[37] Li Y, Zhao C, Liu C. Model-informed generative adversarial network for learning optimal power flow. IISE Trans 2023;1–14.
[38] Liao W, Yang D, Liu Q, Jia Y, Wang C, Yang Z. Data-driven reactive power optimization of distribution networks via graph attention networks. J Mod Power Syst Clean Energy 2024;12(3):874–85.
[39] Grainger JJ, Stevenson Jr. WD. Power system analysis. McGraw-Hill, Inc.; 1994.
[40] Farhangi H, Joos G. Microgrid benchmarks. In: Microgrid planning and design. Wiley; 2019, p. 25–36.
[41] Kerber G. aufnahmefähigkeit von niederspannungsverteilnetzen für die einspeisung aus photovoltaikkleinanlagen [Ph.D. thesis], TECHNISCHE UNIVERSITÄT MÜNCHEN; 2011.
[42] Bukhsh WA, McKinnon K. Network data of real transmission networks. 2013.
[43] Andersson G. Modelling and Analysis of Electric Power Systems Power. EEH Power System Laboratory ETH Zürich; 2008.
[44] Kaseb Z, Xiang Yu, Palensky P, Vergara PP. Adaptive activation functions for deep learning-based power flow analysis. In: IEEE PES ISGT EUROPE. 2023.
[45] Jagtap AD, Kawaguchi K, Karniadakis GE. Adaptive activation functions accelerate convergence in deep and physics-informed neural networks. J Comput Phys 2020;404:109136.
[46] Yan S, Vazinram F, Kaseb Z, Spoor L, Stiasny J, Mamudi B, Ardakani AH, Orji U, Vergara PP, Xiang Y, Guo J. Data driven approach towards more efficient Newton-raphson power flow calculation for distribution grids. 2025.
[47] Ngo Q-H, Nguyen BL, Vu TV, Zhang J, Ngo T. Physics-informed graphical neural network for power system state estimation. Appl Energy 2024;358:122602.
[48] Thurner L, Scheidler A, Schafer F, Menke J-H, Dollichon J, Meier F, Meinecke S, Braun M. Pandapower—An open-source python tool for convenient modeling, analysis, and optimization of electric power systems. IEEE Trans Power Syst 2018;33(6):6510–21.
[49] Safonova A, Ghazaryan G, Stiller S, Main-Knorn M, Nendel C, Ryo M. Ten deep learning techniques to address small data problems with remote sensing. Int J Appl Earth Obs Geoinf 2023;125:103569.
[50] Cubonovic S, Cetenovic D, Rankovic A. Impact of the non-Gaussian measurement noise on the performance of state-of-the-art state estimators for distribution systems. Serbian J Electr Eng 2024;21:113–33.
[51] Falas S, Asprou M, Konstantinou C, Michael MK. Physics-informed neural networks for accelerating power system state estimation. In: 2023 IEEE PES innovative smart grid technologies Europe. IEEE; 2023, p. 1–5.
[52] Akiba T, Sano S, Yanase T, Ohta T, Koyama M. Optuna: A next-generation hyperparameter optimization framework. In: The 25th ACM SIGKDD international conference on knowledge discovery & data mining. 2019, p. 2623–31.