Self-Supervised Primal-Dual Learning for AC Power Flow: A Neural Network Approach

Swastic Keshari Naman Meena Aditi
Electrical Engineering
Indian Institute of Technology Roorkee

***Abstract*—This paper presents a self-supervised primal-dual learning (PDL) approach for solving AC power flow problems in electrical power systems. Unlike traditional supervised learning methods that require extensive pre-solved datasets, our approach mimics the Augmented Lagrangian Method (ALM) trajectory by jointly training primal and dual neural networks. The method learns instance-specific Lagrangian multipliers, enabling fine-grained balance between constraint satisfaction and optimality. Experimental results on six IEEE test cases (14-bus through 300-bus systems) demonstrate that PDL achieves acceptable constraint violations with mean active power violations ranging from 0.011 pu to 0.067 pu, while maintaining real-time inference capabilities (0.012–0.063 ms per sample). The approach eliminates the need for pre-solved training instances and optimization solvers during both training and inference, making it particularly suitable for real-time power system applications.**

***Index Terms*—Power flow, neural networks, primal-dual learning, constrained optimization, self-supervised learning.**

I. Introduction

AC power flow (ACPF) analysis is fundamental to power system operation, providing voltage magnitudes, phase angles, and power flows throughout the network. Traditional numerical methods such as Newton-Raphson require iterative solutions that may be computationally prohibitive for real-time applications, especially as power systems grow in complexity and require frequent re-evaluation under varying load conditions.

Recent advances in machine learning have shown promise in approximating solutions to optimization problems [1]. Supervised learning approaches have been proposed for optimal power flow (OPF) problems [2]–[4], but these methods face several limitations: (i) they require expensive generation of pre-solved training datasets, (ii) they use aggregated constraint multipliers across all instances rather than instance-specific penalties, and (iii) they often struggle to maintain feasibility, requiring post-processing correction steps.

Self-supervised learning methods, such as DC3 [5], address the data generation challenge but remain primal-only methods with limited ability to ensure constraint satisfaction. The Primal-Dual Learning (PDL) framework [6] offers a promising alternative by mimicking Augmented Lagrangian Method (ALM) trajectories and jointly learning both primal and dual networks, enabling instance-specific constraint handling.

This paper applies PDL to AC power flow problems and makes the following contributions:

1) **Self-Supervised Training**: We eliminate the need for pre-solved instances by training directly on the power flow equations and constraints.
2) **Instance-Specific Multipliers**: Through dual network learning, we obtain instance-specific Lagrangian multipliers that provide superior constraint satisfaction compared to aggregated penalty approaches.
3) **Computational Efficiency**: We demonstrate real-time inference (< 0.1 ms) with competitive accuracy on standard IEEE test cases.

II. Problem Formulation

A. AC Power Flow Problem

The AC power flow problem determines the steady-state operating point of a power system. For a network with N buses, the formulation seeks voltage magnitudes V_i and angles θ_i that satisfy power balance equations.

Given the admittance matrix $\mathbf{Y}_{\text{bus}} \in \mathbb{C}^{N\times N}$, the power balance equations at each bus i are:

$$P_i^{\text{inj}} = \sum_{j=1}^{N} |V_i||V_j||Y_{ij}| \cos(\theta_{ij} + \delta_j - \delta_i) \tag{1}$$

$$Q_i^{\text{inj}} = \sum_{j=1}^{N} |V_i||V_j||Y_{ij}| \sin(\theta_{ij} + \delta_j - \delta_i) \tag{2}$$

where P_i^{inj} and Q_i^{inj} represent net active and reactive power injections (generation minus demand), and θ_{ij} is the angle of Y_{ij}.

For power flow analysis, we classify buses into three types:

- **Slack bus**: Voltage magnitude and angle are fixed (typically $V = 1.0$ pu, $\theta = 0$)
- **PV buses**: Generators with fixed active power and voltage magnitude
- **PQ buses**: Load buses with fixed active and reactive power

The power flow problem can be formulated as finding primal variables $\mathbf{y} = [P_g, Q_g, V, \theta]$ such that power balance violations

are minimized:

$$\min_{\mathbf{y}} \quad \|\mathbf{g}_P(\mathbf{y},\mathbf{x})\|^2 + \|\mathbf{g}_Q(\mathbf{y},\mathbf{x})\|^2 \tag{3}$$

$$\text{s.t.} \quad \mathbf{g}_P(\mathbf{y},\mathbf{x}) = \mathbf{P}_{\text{calc}} - \mathbf{P}_{\text{inj}} = 0 \tag{4}$$

$$\mathbf{g}_Q(\mathbf{y},\mathbf{x}) = \mathbf{Q}_{\text{calc}} - \mathbf{Q}_{\text{inj}} = 0 \tag{5}$$

$$P_g^{\min} \le P_g \le P_g^{\max}, \quad Q_g^{\min} \le Q_g \le Q_g^{\max} \tag{6}$$

$$V^{\min} \le V \le V^{\max} \tag{7}$$

where $\mathbf{x}$ represents the instance parameters (load demands).

III. Primal-Dual Learning Methodology

A. Overview

PDL mimics the trajectory of an Augmented Lagrangian Method by jointly training two neural networks: a primal network P_θ that outputs solution estimates, and a dual network D_ϕ that outputs Lagrangian multiplier estimates. The training alternates between primal and dual learning phases.

B. Primal Network Architecture

The primal network maps load demands to power flow solutions:

$$P_\theta : \mathbb{R}^{2N} \to \mathbb{R}^{n_{\text{out}}} \tag{8}$$

where the input consists of concatenated active and reactive load demands at all buses, and the output includes:

- Generator active powers P_g for all generators
- Generator reactive powers Q_g for all generators
- Voltage magnitudes V for PQ buses only
- Voltage angles θ for all buses except slack

The network uses a fully-connected architecture with embedding layers followed by separate output heads for each variable type. Bound constraints are enforced using sigmoid activations scaled to the appropriate limits:

$$P_g = P_g^{\min} + \sigma(h_{Pg}) \cdot (P_g^{\max} - P_g^{\min}) \tag{9}$$

where $\sigma(\cdot)$ is the sigmoid function and h_{Pg} is the hidden representation.

C. Primal Loss Function

The primal network is trained using an augmented Lagrangian loss function [6]:

$$\mathcal{L}_p = \boldsymbol{\lambda}_P^T \mathbf{g}_P + \boldsymbol{\lambda}_Q^T \mathbf{g}_Q + \frac{\rho}{2}(\|\mathbf{g}_P\|^2 + \|\mathbf{g}_Q\|^2) \tag{10}$$

where $\boldsymbol{\lambda}_P$ and $\boldsymbol{\lambda}_Q$ are dual estimates from the frozen dual network D_ϕ, and ρ is the penalty coefficient. This formulation provides instance-specific penalties for constraint violations.

D. Dual Network and Learning

The dual network learns to approximate Lagrangian multipliers:

$$D_\phi : \mathbb{R}^{2N} \to \mathbb{R}^{2N} \tag{11}$$

The dual loss function mimics the ALM multiplier update rule:

$$\begin{aligned}\mathcal{L}_d = &\|\boldsymbol{\lambda}_P - (\boldsymbol{\lambda}_P^k + \rho \mathbf{g}_P)\|^2 \\ &+ \|\boldsymbol{\lambda}_Q - (\boldsymbol{\lambda}_Q^k + \rho \mathbf{g}_Q)\|^2\end{aligned} \tag{12}$$

where $\boldsymbol{\lambda}^k$ are the outputs of a frozen copy of the dual network from the previous outer iteration.

E. Penalty Coefficient Adaptation

Following [7], the penalty coefficient ρ is adapted based on violation trends:

$$v^k = \max(\|\mathbf{g}_P\|_\infty, \|\mathbf{g}_Q\|_\infty) \tag{13}$$

$$\rho \leftarrow \min(\alpha\rho, \rho_{\max}) \quad \text{if } v^k > \tau v^{k-1} \tag{14}$$

where $\alpha > 1$ is the update multiplier and $\tau \in (0,1)$ is a tolerance parameter.

F. Training Algorithm

The complete training procedure alternates between primal and dual learning phases. At each outer iteration k:

1) **Primal Learning**: Update P_θ for L inner iterations while freezing D_ϕ
2) **Dual Learning**: Copy D_ϕ to D_ϕ^k, then update D_ϕ for L inner iterations while freezing P_θ
3) **Penalty Update**: Adjust ρ based on violation metrics

Training continues until maximum violations fall below a threshold or maximum iterations are reached.

IV. Experimental Results

A. Experimental Setup

We evaluate our approach on six IEEE test cases of varying sizes:

- **Case 14**: 14 buses, 4 generators, 11 loads
- **Case 30**: 30 buses, 5 generators, 20 loads
- **Case 39**: 39 buses, 9 generators, 21 loads
- **Case 57**: 57 buses, 6 generators, 42 loads
- **Case 118**: 118 buses, 53 generators, 99 loads
- **Case 300**: 300 buses, 68 generators, 193 loads

The networks were implemented using pandapower [8] with base MVA of 100. Training and test sets were generated with load variations following $f \sim \mathcal{N}(1.0, 0.3)$, where f is a load factor applied to base demands, clipped to $[0.7, 1.3]$.

For each case, we generated 25,000 training samples and 100 test samples. The primal and dual networks used 2-layer MLPs with hidden dimension 256. Training used the Adam optimizer with learning rate 1×10^{-4}. Hyperparameters were set as: $\rho_{\text{init}} = 1.0$, $\rho_{\max} = 10000$, $\alpha = 1.2$, $\tau = 0.5$, with 250 inner iterations per phase and maximum 50 outer iterations.

B. Performance Metrics

We evaluate solutions using:

1) **Maximum violation**: $\max(\|\mathbf{g}_P\|_\infty, \|\mathbf{g}_Q\|_\infty)$
2) **Mean violation**: Average absolute violations across all constraints
3) **Inference time**: Time to predict a solution

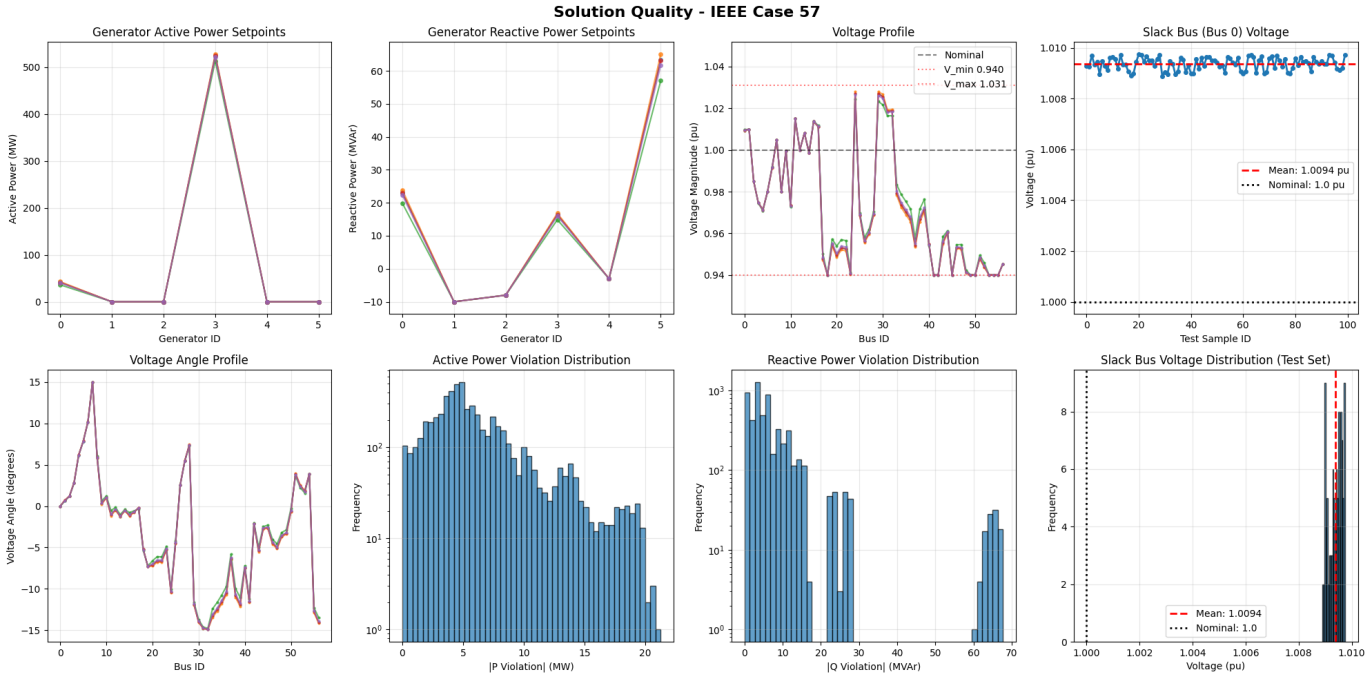

Fig. 1. PDL Training Results for the IEEE 57-bus System. The figure illustrates objective function evolution, primal and dual losses, constraint violations, slack bus voltage behavior, penalty coefficient progression, and distributional characteristics across training iterations.

C. Results

Table I presents comprehensive performance metrics across all six test cases. The results demonstrate that PDL scales effectively from small (14-bus) to large (300-bus) systems while maintaining real-time inference capabilities.

For smaller systems (Cases 14 and 30), PDL achieves excellent constraint satisfaction with maximum overall violations of 0.140 pu and 0.154 pu respectively, and very low mean active power violations (0.011 pu and 0.006 pu). The inference times remain exceptionally fast at 0.012 ms and 0.018 ms per sample.

Medium-sized systems (Cases 39, 57, and 118) show increasing but still acceptable violations. Case 118, with 118 buses and 53 generators, achieves maximum violations of 0.337 pu with mean active power violations of 0.033 pu (3.3 MW), while maintaining inference time of 0.032 ms. Case 57 exhibits slightly higher maximum violations (0.676 pu) but with reasonable mean violations of 0.060 pu for active power.

The largest system (Case 300) demonstrates the scalability of the approach, processing 300 buses in 0.063 ms per sample. While maximum violations increase to 1.755 pu (175 MVAr for reactive power), the mean violations remain practical at 0.067 pu for active power and 0.167 pu for reactive power.

Importantly, the slack bus voltage remains stable across all cases. The mean slack bus voltages range from 0.940 pu (Case 39) to 1.045 pu (Case 14), with very low standard deviations (0.0001–0.0013 pu), indicating stable convergence behavior. Case 30 shows the closest adherence to nominal voltage with a mean of 0.997 pu and deviation of only 0.0028 pu from 1.0 pu.

TABLE I
PERFORMANCE RESULTS ON IEEE TEST CASES

Metric	C14	C30	C39	C57	C118	C300
Max P (pu)	0.068	0.058	0.343	0.213	0.214	0.782
Max Q (pu)	0.139	0.154	0.845	0.676	0.337	1.755
Mean P (pu)	0.011	0.006	0.061	0.060	0.033	0.067
Mean Q (pu)	0.019	0.006	0.348	0.070	0.073	0.167
Slack V (pu)	1.045	0.997	0.940	1.009	1.030	0.983
Slack σ (pu)	0.000	0.000	0.000	0.000	0.000	0.001
Train (s)	195	255	337	407	1108	3550
Infer (ms)	0.012	0.018	0.018	0.020	0.032	0.063

All voltage magnitudes generally remain within or near system limits, though Case 300 shows some violations with minimum voltage of 0.929 pu (limit: 0.94 pu) and maximum of 1.074 pu (limit: 1.06 pu), suggesting the need for longer training or architectural refinements for very large systems.

D. Training Behavior

Training convergence varies across different system sizes. Smaller systems (Cases 14 and 30) show rapid initial reduction in violations, with maximum violations decreasing from over 1.0 pu to below 0.2 pu within the first 15–20 iterations. The penalty coefficient ρ increases adaptively, reaching values between 2500–3000 by iteration 50.

Medium-sized systems exhibit similar patterns but with more gradual convergence. Case 118 demonstrates steady violation reduction from 1.17 pu initially to 0.346 pu after 50 iterations, with ρ reaching 1020. Case 57 shows consistent improvement from 1.42 pu to 0.682 pu.

The largest system (Case 300) presents greater challenges, with violations decreasing from 11.26 pu to 1.80 pu after 50 iterations. The highly nonlinear nature and increased system complexity require longer training. Despite training for only 50 outer iterations, the penalty coefficient reaches 237, indicating active constraint enforcement.

The training process required no pre-solved instances. Training times scale approximately linearly with system size, ranging from 195 seconds for Case 14 to 3550 seconds (approximately 1 hour) for Case 300 on a Tesla RTX6000 GPU. In contrast, generating 25,000 pre-solved instances for supervised learning would require substantially more computational resources.

V. Conclusion

This paper presents a self-supervised primal-dual learning approach for AC power flow analysis. The method eliminates the need for pre-solved training datasets while achieving competitive accuracy through instance-specific Lagrangian multipliers learned by a dual network.

Experimental results on six IEEE test cases ranging from 14 to 300 buses demonstrate real-time inference capabilities (0.012–0.063 ms per sample) with acceptable constraint violations for practical applications. Smaller systems (Cases 14 and 30) achieve excellent accuracy with mean active power violations below 0.011 pu, while larger systems maintain practical performance with mean violations of 0.033–0.067 pu. The slack bus voltage remains stable across all cases, with standard deviations below 0.0013 pu.

The scalability analysis reveals that training time scales approximately linearly with system size (195 s for 14 buses to 3550 s for 300 buses), while inference remains real-time even for the largest systems. This makes the approach particularly suitable for applications requiring frequent re-evaluation under varying load conditions.

Future work will explore: (i) extended training with adaptive early stopping criteria for tighter convergence on large systems, (ii) architectural improvements such as graph neural networks to better capture power system topology, (iii) extension to optimal power flow with economic dispatch objectives, (iv) incorporation of contingency constraints for security-constrained analysis, and (v) development of hybrid approaches combining PDL with traditional solvers for critical applications requiring guaranteed convergence.

The self-supervised nature of PDL makes it particularly attractive for scenarios where gathering training data is expensive or where the system topology and parameters change frequently, requiring model retraining without the computational burden of generating extensive pre-solved datasets.

References

[1] Y. Bengio, A. Lodi, and A. Prouvost, "Machine learning for combinatorial optimization: a methodological tour d'horizon," *European Journal of Operational Research*, vol. 290, no. 2, pp. 405–421, 2021.

[2] A. S. Zamzam and K. Baker, "Learning optimal solutions for extremely fast AC optimal power flow," in *Proc. IEEE SmartGridComm*, 2020, pp. 1–6.

[3] F. Fioretto, T. W. Mak, and P. Van Hentenryck, "Predicting AC optimal power flows: Combining deep learning and Lagrangian dual methods," in *Proc. AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 630–637.

[4] M. Chatzos, F. Fioretto, T. W. Mak, and P. Van Hentenryck, "High-fidelity machine learning approximations of large-scale optimal power flow," *arXiv preprint arXiv:2006.16356*, 2020.

[5] P. L. Donti, D. Rolnick, and J. Z. Kolter, "DC3: A learning method for optimization with hard constraints," *arXiv preprint arXiv:2104.12225*, 2021.

[6] S. Park and P. Van Hentenryck, "Self-supervised primal-dual learning for constrained optimization," in *Proc. AAAI Conference on Artificial Intelligence*, 2023.

[7] R. Andreani, E. G. Birgin, J. M. Martínez, and M. L. Schuverdt, "On augmented Lagrangian methods with general lower-level constraints," *SIAM Journal on Optimization*, vol. 18, no. 4, pp. 1286–1309, 2008.

[8] L. Thurner et al., "pandapower—An open-source python tool for convenient modeling, analysis, and optimization of electric power systems," *IEEE Transactions on Power Systems*, vol. 33, no. 6, pp. 6510–6521, 2018.

[9] Some framework ideas were adapted with assistance from the *Claude AI assistant*. Available: https://www.anthropic.com/claude.