# Homotopy-Guided Self-Supervised Learning of Parametric Solutions for AC Optimal Power Flow

Shimiao Li[1], Aaron Tuor[2], Draguna Vrabie[2], Larry Pileggi[3], Jan Drgona[4]

***Abstract*—Learning to optimize (L2O) parametric approximations of AC optimal power flow (AC-OPF) solutions offers the potential for fast, reusable decision-making in real-time power system operations. However, the inherent nonconvexity of AC-OPF results in challenging optimization landscapes, and standard learning approaches often fail to converge to feasible, high-quality solutions. This work introduces a *homotopy-guided self-supervised L2O method* for parametric AC-OPF problems. The key idea is to construct a continuous deformation of the objective and constraints during training, beginning from a relaxed problem with a broad basin of attraction and gradually transforming it toward the original problem. The resulting learning process improves convergence stability and promotes feasibility without requiring labeled optimal solutions or external solvers. We evaluate the proposed method on standard IEEE AC-OPF benchmarks and show that homotopy-guided L2O significantly increases feasibility rates compared to non-homotopy baselines, while achieving objective values comparable to full OPF solvers. These findings demonstrate the promise of homotopy-based heuristics for scalable, constraint-aware L2O in power system optimization.**

***Index Terms*—learning to optimize, homotopy, meta-optimization heuristics, optimal power flow**

## I. Introduction

Learning to Optimize (L2O) [1] offers an emerging alternative to repeatedly solving large-scale optimization problems in power system operations. Instead of running a numerical solver from scratch for each new operating condition, the idea is to learn a parametric solution map that directly predicts optimal (or near-optimal) solutions as a function of system inputs, enabling rapid inference suitable for real-time and embedded decision-making. This paradigm is particularly attractive for problems such as AC Optimal Power Flow (AC-OPF) [2], [3], where repeated decisions of optimal generation outputs and setting points are required across time-varying grid conditions.

However, realistic AC-OPF problems are highly constrained and strongly nonconvex. They include (i) nonlinear AC power balance equations and (ii) operational inequality limits on voltages, generations, and transmission lines. These constraints lead to nonconvex parametric nonlinear programs (pNLPs) that are, in general, NP-hard. Directly learning parametric solution maps for such problems is therefore difficult, due to the rugged optimization landscape, where simple penalty-based training often converges to infeasible or poor-quality solutions.

Earlier L2O approaches predominantly followed a supervised learning paradigm [4], where models are trained to mimic the solver's outputs in labeled data. Yet, these methods inherit the limitations of their offline training data and lack explicit enforcement of feasibility. As a result, their generalization to unseen operating conditions is often weak, producing solutions that violate power flow or operational constraints.

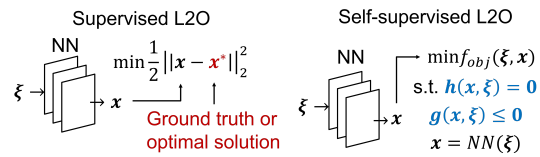


Fig. 1: Supervised vs self-supervised learning to optimize (L2O).

To address feasibility and convergence challenges, recent work has explored self-supervised L2O, where the learning objective is defined directly by the AC-OPF problem itself [5], [6]. This removes dependence on labeled datasets but introduces the difficulty of solving a nonconvex constrained optimization problem in the parameter space of the problem. As a result, numerous constraint-handling strategies have been adopted, including penalty-based formulations [7], [8], primal-dual Lagrangian training [9]–[12], sensitivity-informed training [13], projected or correction-based methods [14]–[16], and completion-layer architectures [14], [17]. While each method offers advantages, they can still suffer from unstable convergence in highly nonconvex settings, particularly when feasible solutions lie in narrow basins of attraction.

Broadly speaking, these new methods build on sensitivity analysis developed in the context of operations research [18], [19] and later adopted in control theory applications [20], [21]. In the literature, these methods can be found under different names, such as constrained deep learning, optimization learning, or deep declarative networks [1], [22], [23].

**Contributions.** This work contributes to these efforts by introducing a homotopy-guided training approach designed to improve convergence stability in self-supervised L2O for AC-OPF. Specifically, we make the following contributions:

1) **Homotopy Continuation Heuristics for Self-Supervised L2O.** We propose general homotopy-based heuristics that can be integrated into any self-supervised L2O framework for constrained nonlinear optimization. Two complementary strategies are developed: (a) *Relaxation-based homotopy*, which gradually tightens objective weights and constraint bounds to enlarge the basin of attraction; and (b) *AC-OPF-aware homotopy*, which reparameterizes the problem to expose feasible starting points and maintain feasibility during training.

Authors are with [1]University at Buffalo; [2]Pacific Northwestern National Lab; [3]Carnegie Mellon University; [4]John Hopkins University;

2) **Empirical Validation on AC-OPF Benchmarks.** We evaluate the proposed approach on standard IEEE test systems. Results show that homotopy-guided L2O substantially improves feasibility rates, while achieving objective values comparable to full AC-OPF solvers and outperforming non-homotopy L2O baselines.

## II. Related Work

Homotopy method is a type of meta-heuristics to handle hard problems, which can otherwise easily diverge or converge to a bad point. It decomposes the original (nonlinear) problem $F(x)$ into a series of sub-problems $H(x, \lambda_H)$, creating a path of optimizers driven by the change of homotopy parameter $\lambda_H$. As $\lambda_H$ shifts from 0 to 1, the sub-problem $H(x, \lambda_H)$ continuously transforms from a simple problem to the original one. Commonly, homotopy-based heuristics have been used for local optimization of nonlinear problems. The most popular form of designing $H(x, \lambda_H)$ is via a linear combination of a trivial problem $H_0(x)$ and the original one, such that $H(x, \lambda_H) = (1 - \lambda_H)H_0(x) + \lambda_H F(x)$. Existing works have developed perturbation techniques for general nonlinear [24], [25] and multi-objective problems [26].

Domain-specific homotopy heuristics have also been developed to design $H(x, \lambda_H)$ whose solutions are trivial when $\lambda_H = 0$. For example, in the domain of circuit simulation [27], Gmin-stepping initially shorts all nodes to ground and gradually removes the short-circuit effect; Tx-stepping initially shorts all transmission lines, and then gradually returns to the original branches. Works in [28] [29] [30] [29] further applied the circuit-theoretic homotopy ideas to power grid simulation and optimization tasks. Others [31] extended homotopy methods to global optimization tasks through an ensemble of solution points, with probabilistic convergence bounds. In this work, we build on these foundations and bring the idea of homotopy heuristics to self-supervised L2O.

## III. AC-OPF Problem Formulation

The problem of interest in this paper is the AC optimal power flow (ACOPF) [29], [32], which determines a cost-optimal generator dispatch that satisfies AC power balance and operational limits. An ACOPF formulation is:

$$\min_{x=[V_i, P_g, Q_g]} \sum_{g \in \mathcal{N}_g} \alpha_g P_g^2 + \beta_g P_g + \gamma_g \tag{1a}$$

$$\text{s.t.} (P_g - P_d) + j(Q_g - Q_d) = V \odot (Y_{\text{bus}} V)^* \tag{1b}$$

$$V^{\text{imag}}_{\text{slack bus}} = 0 \tag{1c}$$

$$|V|_i^- \leq |V|_i \leq |V|_i^+, \quad \forall \text{ bus } i \tag{1d}$$

$$P_g^- \leq P_g \leq P_g^+, \forall \text{ generator } g \tag{1e}$$

$$Q_g^- \leq Q_g \leq Q_g^+, \forall \text{ generator } g \tag{1f}$$

where $P_d, Q_d$ are demand at each load $d$, $(\alpha_g, \beta_g, \gamma_g)$ are generator cost coefficients, $P_g, Q_g$ are generator outputs, $V = V^{\text{real}} + jV^{\text{imag}}$ is the bus-voltage vector, $|V|$ its magnitude, and superscripts $-, +$ denote bounds.

## IV. Homotopy Heuristics for Self-Supervised Learning to Optimize

We propose a homotopy-based training framework for self-supervised learning to optimize (L2O) with structured transformations of objectives and constraints. A neural network policy $\pi_\Theta$ approximates the solution map of a constrained optimization problem, and homotopy is used to guide training from relaxed, easy-to-satisfy problems toward the original problem, improving feasibility and convergence.

### A. Self-Supervised Learning to Optimize

To avoid reliance on labeled optimal solutions and address scalability in parametric nonlinear programs, we adopt a self-supervised L2O formulation as the original problem $F(x)$:

$$\min_\Theta f_{\text{obj}}(x, \xi) \text{ s.t. } g(x, \xi) \leq 0, \; h(x, \xi) = 0, \tag{2a}$$

$$x = \pi_\Theta(\xi), \; \forall \xi \in \Xi, \tag{2b}$$

where $\pi_\Theta$ maps problem parameters $\xi$ to decision variables $x$.

In this work, we focus on applying a penalty-based formulation for training $\pi_\Theta$, where the training loss function is:

$$\min_\Theta \; f_{\text{obj}}(\cdot) + \sum_i w_i p_i(h_i(\cdot)) + \sum_i w_i p_i(g_i(\cdot)) \tag{3}$$

where $p_i(\cdot)$ penalize constraint violations (e.g., squared residuals for equalities and ReLU for inequalities), and gradients are computed via automatic differentiation through the neural policy $x = \pi_\Theta(\xi)$.

### B. Homotopy-Based Training

Direct optimization of (3) for the original highly nonconvex, constrained problem often suffers from infeasible solutions. We therefore introduce a homotopy parameter $\lambda_H \in [0, 1]$ and train $\pi_\Theta$ over a sequence of homotopy problems $H(x, \lambda_H)$:

$$\min_\Theta f_{\lambda_H}(\cdot) \text{ s.t. } g_{\lambda_H}(\cdot) \leq 0, \; h_{\lambda_H}(\cdot) = 0, \text{(2b)} \tag{4}$$

where $(f_{\lambda_H}, g_{\lambda_H}, h_{\lambda_H})$ interpolate between relaxed and original formulations. The same penalty method as in (3) is applied to the gradually tightened problem $H(x, \lambda_H)$, leading to

$$\min_\Theta f_{\lambda_H}(\cdot) + \sum_i w_i p_i(h_{\lambda_H, i}(\cdot)) + \sum_i w_i p_i(g_{\lambda_H, i}(\cdot)) \tag{5}$$

Training proceeds by: (i) initializing at an easy problem with $\lambda_H = 0$, (ii) optimizing $\Theta$ for a fixed number of steps, (iii) incrementing $\lambda_H \leftarrow \lambda_H + \Delta\lambda_H$, and (iv) continuing from the previous $\Theta$ as a warm start. This schedule gradually approaches feasible regions of the original hard problem instead of learning it from scratch, as in Fig. 2. We then design two complementary types of homotopy transformations.

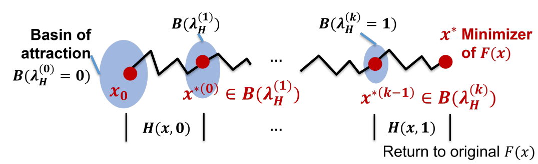


Fig. 2: A desirable homotopy path where the previous problem solution falls within the basin of attraction for the next.

### C. Type I: Relaxation-Based Homotopy

Type I heuristics start from a relaxed objective and constraints set with a large feasible basin, and then gradually enforce the true problem.

*Convexify Objective (CObj):* Replace a nonconvex $f_{\text{obj}}$ by a convex surrogate $f_{\text{cvx}}$ at $\lambda_H = 0$ and interpolate:

$$f_{\lambda_H} = (1 - \lambda_H) f_{\text{cvx}} + \lambda_H f_{\text{obj}}. \tag{6}$$

*Shrink Bounds (SBnds):* Start with loose initial bounds $x \in [\epsilon^-, \epsilon^+]$ and gradually tighten them back to $x \in [x^-, x^+]$:

$$x^-_{\lambda_H} = (1-\lambda_H)\epsilon^- + \lambda_H x^-; x^+_{\lambda_H} = (1-\lambda_H)\epsilon^+ + \lambda_H x^+ \tag{7}$$

*Grow Penalty (GPen):* Scale (inequality) constraint penalties with $\lambda_H$ so that violations are weakly penalized early and strictly penalized later:

$$\lambda_H g(\cdot) \leq 0 \tag{8}$$

*Split and Shrink (SaS) for Equalities:* Convert $h(\cdot) = 0$ to inequalities $-\epsilon_{\lambda_H} \leq h(\cdot) \leq \epsilon_{\lambda_H}$ with

$$\epsilon_{\lambda_H} = (1 - \lambda_H)\epsilon_H + \lambda_H \epsilon_L, \tag{9}$$

where $\epsilon_H \gg \epsilon_L \approx 0$. As $\lambda_H \to 1$, equalities are enforced tightly.

We combine CObj, SBnds, SaS, and GPen into (5) to obtain a smooth transformation from relaxed to strict constraints.

### D. Type II: AC-OPF-Aware Homotopy with Trivial Solutions

Type II heuristics exploit power-system structures to construct homotopies for problems like ACOPF.

*Load-Stepping:* We gradually increase loads from zero to their original values:

$$h_{\lambda_H} : (P_g - \lambda_H P_d) + j(Q_g - \lambda_H Q_d) - v \odot (Y_{\text{bus}} v)^* = 0. \tag{10}$$

At $\lambda_H = 0$, a trivial no-load solution $x_0$ exists (intuitively, voltage and power outputs should be all zeros when loads are zero); the network is first trained to satisfy this easy condition (optionally regularized toward some labels $x_0$), and then moves back to the original load condition as $\lambda_H$ increases.

*Tx-Stepping:* We deform the system admittance matrix:

$$Y_{\lambda_H} = (1 - \lambda_H) Y_0 + \lambda_H Y_{\text{bus}}, \tag{11}$$

where $Y_0$ corresponds to low-impedance (nearly shorted) lines. At $\lambda_H = 0$, the system is well-conditioned with voltages close to the reference at all buses, yielding an easy feasible solution. As $\lambda_H$ grows, the NN is trained to maintain feasibility under the gradually restored physical network.

Both load-stepping and Tx-stepping can be combined with Type I relaxations inside the same homotopy schedule, enabling a robust training procedure for generic L2O.

## V. Numerical Results

We evaluate the efficacy of homotopy heuristics on both the power grid optimal power flow problem and general random non-convex constrained optimization problems to assess generality. The use of homotopy is expected to enable a more reliable convergence of the neural network models to outperform non-homotopy results on the following criteria:

- **Optimality**: the objective function $f_{obj}$ achieved by the solution. For the ACOPF problem, it represents the per-hour cost ($/h) of the generation dispatch.
- **Feasibility**: how much the solution $\hat{x}$ violates the equality and inequality constraints. Feasibility is quantified by the mean and maximum violation of constraints: $\text{mean}(h(\hat{x}))$, $\max(h(\hat{x}))$, $\text{mean}(relu(g(\hat{x})))$, $\max(relu(g(\hat{x})))$. In real-world tasks, smaller violation of constraints means the neural network outputs a more practical solution for real-world optimization and control.

We compare the different versions of homotopy L2O with a non-homotopy baseline represented by the penalty method (3). Appendix A describes the details on experiment settings and hyper-parameter tuning, for a fair comparison.

### A. Power Grid AC Optimal Power Flow Problem

This subsection evaluates the real-world task of the ACOPF problem (defined in Section III). Table I shows ACOPF results on a 30-bus system. Homotopy methods improve the feasibility of NN outputs with smaller violations of equality and inequality constraints. See Appendix A for our experiment settings and hyper-parameter tuning.

### B. Non-convex Optimization with Random System Constraints

First, consider a problem with a non-convex objective and random linear inequality constraints:

$$\min_x \sum_{i=1}^{n-1} (1 - x_i)^2 + 2(x_{i+1} - x_i^2)^2 \ \text{ s.t. } \ Ax \leq b + C\xi \tag{12}$$

where $n$ is the problem size (complexity), $x$ is the solution vector representing $n$ variables to solve, $\xi$ is an $round(0.4{*}n) \times 1$ vector representing the (known) input parameter that varies across instances, $A^{n\times n}, b^{n\times 1}, C^{dim(\xi)}$ are randomly generated matrices representing $n$ random linear constraints.

Table II shows results on test data for problems with varying complexity. Results demonstrate that adding homotopy heuristics onto the penalty method enables neural network outputs to have a smaller violation of constraints. See Appendix A for details on experiment settings.

## VI. Conclusion

This work introduced a homotopy-guided training heuristic for self-supervised Learning to Optimize (L2O) in highly nonconvex constrained optimization problems, with a focus on AC Optimal Power Flow. The proposed homotopy heuristics reshape the optimization landscape by (i) gradually relaxing and tightening objectives and constraint bounds to expand and guide the basin of attraction, and (ii) exploiting power-system

| Method | Obj | Mean eq. | Max eq. | Mean ineq. | Max ineq. |
|---|---|---|---|---|---|
| **SaS, SBnds** | 666 | 0.0027 (0.0015) | 0.010 (0.005) | 0.0000 (0.0000) | 0.000 (0.001) |
| **SaS, GPen** | 666 | 0.0023 (0.0014) | 0.008 (0.005) | 0.0000 (0.0000) | 0.000 (0.000) |
| **Tx-stepping, SBnds** | 665 | 0.0024 (0.0013) | 0.008 (0.005) | 0.0000 (0.0000) | 0.000 (0.000) |
| **Tx-stepping, GPen** | 665 | 0.0017 (0.0009) | 0.006 (0.003) | 0.0000 (0.0000) | 0.000 (0.000) |
| **Load-stepping, SBnds** | 666 | 0.0020 (0.0012) | 0.007 (0.004) | 0.0000 (0.0000) | 0.000 (0.000) |
| **Load-stepping, GPen** | 667 | 0.0024 (0.0016) | 0.009 (0.005) | 0.0000 (0.0000) | 0.000 (0.000) |
| **Original penalty** | 673 | 0.0036 (0.0027) | 0.013 (0.010) | 0.0000 (0.0000) | 0.001 (0.003) |

TABLE I: Results of ACOPF problem on case30, over 100 test instances. Mean and max violations are analyzed across instances, we list the average value (std). The original penalty method has larger violations of constraints, whereas homotopy methods have smaller violations.

| Method | | Complexity $n$ | | | |
|---|---|---|---|---|---|
| | | 5<br>$\Delta\lambda_H = 0.05$ | 25<br>$\Delta\lambda_H = 0.05$ | 50<br>$\Delta\lambda_H = 0.05$ | 100<br>$\Delta\lambda_H = 0.005$, $w_{ineq} = 500$ |
| **CObj, SaS, SBnds** | obj | 0.48 | 31.71 | 62.03 | 276.16 |
| | mean viol | 0.0000(0.0000) | 0.0001(0.0002) | 0.0003(0.0011) | 0.0002(0.0009) |
| | max viol | 0.0000(0.0000) | 0.0016(0.0048) | 0.0129(0.0526) | 0.0246(0.0947) |
| **CObj, SaS, GPen** | obj | 0.48 | 31.69 | 62.59 | 253.42 |
| | mean viol | 0.0000(0.0000) | 0.0001(0.0005) | 0.0002(0.0005) | 0.0004(0.0018) |
| | max viol | 0.0000(0.0000) | 0.0029(0.0121) | 0.0061(0.0141) | 0.0297(0.1032) |
| **SaS, SBnds** | obj | 0.48 | 33.35 | 62.33 | 272.36 |
| | mean viol | 0.0000(0.0000) | 0.0001(0.0003) | 0.0002(0.0009) | 0.0002(0.0009) |
| | max viol | 0.0000(0.0000) | 0.0023(0.0066) | 0.0082(0.0222) | 0.0180(0.0711) |
| **SaS, GPen** | obj | 0.47 | 32.15 | 62.20 | 447.32 |
| | mean viol | 0.0000(0.0000) | 0.0001(0.0004) | 0.0006(0.0035) | 0.0031(0.0067) |
| | max viol | 0.0000(0.0000) | 0.0024(0.0080) | 0.0095(0.0347) | 0.1990(0.3583) |
| **Original** (penalty method) | obj | 0.46 | 31.84 | 62.44 | 208.71 |
| | mean viol | 0.0000(0.0000) | 0.0005(0.0010) | 0.0005(0.0011) | 0.0015(0.0023) |
| | max viol | 0.0000(0.0000) | 0.0095(0.0160) | 0.0193(0.0470) | 0.1249(0.1877) |

TABLE II: Non-convex problem with $n$ variables and $n$ random linear constraints as $n$ varies as $5, 25, 50, 100$. Results over 100 test instances are listed, using metrics of objective, mean, and max inequality constraint violations. The violations are formatted as average value (std) in this Table. Results show that the homotopy heuristics enable a smaller violations than the original penalty method.

structure to construct domain-aware continuation paths with trivial feasible starting points. These mechanisms are general and can be integrated into a broad range of L2O methods.

Numerical experiments on AC-OPF test cases and synthetic nonconvex constrained problems show that, compared to the penalty-based approach, homotopy-guided training consistently improves convergence reliability and significantly reduces constraint violations, while maintaining near-optimal objective performance. These results highlight homotopy continuation as an effective heuristic for improving the convergence of self-supervised L2O in power system applications.


## Acknowledgments

This research was partially supported by the Data Model Convergence (DMC) initiative at Pacific Northwest National Laboratory (PNNL), and by the Ralph O'Connor Sustainable Energy Institute at Johns Hopkins University.


## Appendix

### A. Experiment settings and hyper-parameter tuning

To create a fair comparison in each optimization problem, experiments with and without homotopy heuristics will train with the same NN architecture, Adam optimizer, and learning rate scheduler (StepLR with step=100, gamma=0.1, and a minimal learning rate $10^{-5}$; in homotopy methods, lr scheduler is only applied to the last homotopy step). Early stopping is also applied in each experiment to avoid overfitting: each original penalty method trains for 1,000 epochs with warmup=50, patience=200, and each homotopy method trains 100 epochs in each homotopy step with warmup=50, patience=50. All NNs are trained on PyTorch.

Experiment settings for the non-convex problem with random linear constraints, see problem definition (12) in Section V-B, are listed below:

- Dataset: 50,000 instances (with train/validation/test ratio 8:1:1)
- NN architecture: cylinder NN with 4 layers, hidden layer size increases with problem size $n$ by $hiddenlayersize = 30\sqrt{n/5}$
- Penalty weights: $w_{eq} = 50$, $w_{ineq} = 50$
- Homotopy settings: CObj has $f_{cvx} = \sum_{i=1}^{n-1}(1 - x_i)^2 + 2(x_{i+1}^2 + x_i^4)$, SaS has $\epsilon_H = 0.01, \epsilon_L = 0$, SBnds has $\epsilon^+ = 1, \epsilon^- = 0$.

These hyper-parameters are tuned and kept fixed across all experiments of this non-convex problem.

For the ACOPF problem, we propose an additional trick of $P_g$ pull-up and used it on all experiments.

**($P_g$ pull-up)** Based on power system domain knowledge, any decision with supply lower than demand is always technically infeasible. To avoid bad predictions of this type, we apply the heuristics of $P_g$ pull-up where an additional domain-specific constraint $\sum P_g - \sum P_d \geq \epsilon$ is added to pull the generation up and thus promote convergence to a point with total supply higher than demand. This constraint is not subject to homotopy heuristics and remains the same in the homotopy path. experiment settings are as follows. Similarly as for other constraints, we have a weight $w_{pullup}$ to impose the additional constraint in penalty method.

- Dataset: 50,000 instances (with train/validation/test ratio 8:1:1), data are generated by randomly sampling load profiles in the range of $75\% - 150\%$ of the base load profile (base load is the load in case data).
- Batchsize: 1024
- NN architecture: cylinder NN with 2 layers and hidden layer size = 200,
- Penalty weights: $w_{eq} = 10^5$, $w_{ineq} = 10^6$
- General homotopy settings: $\Delta\lambda_H = 0.05$, SaS has $\epsilon_H = 0.01, \epsilon_L = 0$, SBnds has $\epsilon^+ = 1, \epsilon^- = 0$.
- Domain specific homotopy settings: warm homotopy loss has $w_{warm} = 10^6$
- others: $P_g$ pull up has $w_{pullup} = 10^6, \epsilon = 0.01$

The hyper-parameters are kept fixed across all experiments.

## References


[1] P. Van Hentenryck, "Optimization learning," 2025. [Online]. Available: https://arxiv.org/abs/2501.03443

[2] A. S. Zamzam and K. Baker, "Learning optimal solutions for extremely fast ac optimal power flow," in *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, 2020, pp. 1–6.

[3] S. Park, W. Chen, T. W. Mak, and P. Van Hentenryck, "Compact optimization learning for ac optimal power flow," *IEEE Transactions on Power Systems*, vol. 39, no. 2, pp. 4350–4359, 2024.

[4] J. Kotary, F. Fioretto, and P. Van Hentenryck, "Learning hard optimization problems: A data generation perspective," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 24 981–24 992. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/file/d1942a3ab01eb59220e2b3a46e7ef09d-Paper.pdf

[5] R. Nellikkath and S. Chatzivasileiadis, "Physics-informed neural networks for ac optimal power flow," *Electric Power Systems Research*, vol. 212, p. 108412, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378779622005636

[6] W. Huang, M. Chen, and S. H. Low, "Unsupervised learning for solving ac optimal power flows: Design, analysis, and experiment," *IEEE Transactions on Power Systems*, vol. 39, no. 6, pp. 7102–7114, 2024.

[7] 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.

[8] X. Hu, H. Hu, S. Verma, and Z.-L. Zhang, "Physics-guided deep neural networks for power flow analysis," *IEEE Transactions on Power Systems*, vol. 36, no. 3, pp. 2082–2092, 2020.

[9] Y. Nandwani, A. Pathak, and P. Singla, "A primal dual formulation for deep learning with constraints," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[10] F. Fioretto, T. W. Mak, and P. Van Hentenryck, "Predicting ac optimal power flows: Combining deep learning and lagrangian dual methods," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, pp. 630–637, Apr. 2020.

[11] L. Zhang and B. Zhang, "Learning to solve the ac optimal power flow via a lagrangian approach," in *2022 North American Power Symposium (NAPS)*, 2022, pp. 1–6.

[12] X. Pan, W. Huang, M. Chen, and S. H. Low, "Deepopf-al: Augmented learning for solving ac-opf problems with a multi-valued load-solution mapping," in *Proceedings of the 14th ACM International Conference on Future Energy Systems*, ser. e-Energy '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 42–47. [Online]. Available: https://doi.org/10.1145/3575813.3576874

[13] M. K. Singh, V. Kekatos, and G. B. Giannakis, "Learning to solve the ac-opf using sensitivity-informed deep neural networks," *IEEE Transactions on Power Systems*, vol. 37, no. 4, pp. 2833–2846, 2022.

[14] P. L. Donti, D. Rolnick, and J. Z. Kolter, "Dc3: A learning method for optimization with hard constraints," *arXiv preprint arXiv:2104.12225*, 2021.

[15] P. Márquez-Neila, M. Salzmann, and P. Fua, "Imposing hard constraints on deep networks: Promises and limitations," *arXiv preprint arXiv:1706.02025*, 2017.

[16] X. Pan, T. Zhao, and M. Chen, "Deepopf: Deep neural network for dc optimal power flow," in *2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. IEEE, 2019, pp. 1–6.

[17] T. Beucler, M. Pritchard, S. Rasp, J. Ott, P. Baldi, and P. Gentine, "Enforcing analytic constraints in neural networks emulating physical systems," *Physical Review Letters*, vol. 126, no. 9, p. 098302, 2021.

[18] T. Gal and J. Nedoma, "Multiparametric linear programming," *Management Science*, vol. 18, no. 7, pp. 406–422, 1972. [Online]. Available: https://doi.org/10.1287/mnsc.18.7.406

[19] T. Gal and H. Greenberg, *Advances in Sensitivity Analysis and Parametric Programming*, ser. International Series in Operations Research & Management Science. Springer US. [Online]. Available: https://books.google.com/books?id=PufhfbsBq24C

[20] A. Bemporad, M. Morari, V. Dua, and E. Pistikopoulos, "The explicit solution of model predictive control via multiparametric quadratic programming," in *Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No.00CH36334)*, vol. 2, 2000, pp. 872–876 vol.2.

[21] M. Herceg, M. Kvasnica, C. Jones, and M. Morari, "Multi-Parametric Toolbox 3.0," in *Proc. of the European Control Conference*, Zürich, Switzerland, July 17–19 2013, pp. 502–510, http://control.ee.ethz.ch/~mpt.

[22] S. Gould, R. Hartley, and D. Campbell, "Deep declarative networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 8, pp. 3988–4004, 2021.

[23] J. Kotary, F. Fioretto, P. Van Hentenryck, and B. Wilder, "End-to-end constrained optimization learning: A survey," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, 2021, pp. 4475–4482. [Online]. Available: https://doi.org/10.24963/ijcai.2021/610

[24] J.-H. He, "Homotopy perturbation technique," *Computer methods in applied mechanics and engineering*, vol. 178, no. 3-4, pp. 257–262, 1999.

[25] S. Liao, "On the homotopy analysis method for nonlinear problems," *Applied Mathematics and Computation*, vol. 147, no. 2, pp. 499–513, 2004.

[26] C. Hillermeier *et al.*, *Nonlinear multiobjective optimization: a generalized homotopy approach*. Springer Science & Business Media, 2001, vol. 135.

[27] F. N. Najm, *Circuit simulation*. John Wiley & Sons, 2010.

[28] A. Pandey, M. Jereminov, M. R. Wagner, G. Hug, and L. Pileggi, "Robust convergence of power flow using tx stepping method with equivalent circuit formulation," in *2018 Power Systems Computation Conference (PSCC)*. IEEE, 2018, pp. 1–7.

[29] A. Pandey, A. Agarwal, and L. Pileggi, "Incremental model building homotopy approach for solving exact ac-constrained optimal power flow," *arXiv preprint arXiv:2011.00587*, 2020.

[30] M. Jereminov, A. Terzakis, M. Wagner, A. Pandey, and L. Pileggi, "Robust and efficient power flow convergence with g-min stepping homotopy method," in *2019 IEEE International Conference on Environment and Electrical Engineering and 2019 IEEE Industrial and Commercial Power Systems Europe (EEEIC/I&CPS Europe)*. IEEE, 2019, pp. 1–6.

[31] D. M. Dunlavy and D. P. O'Leary, "Homotopy optimization methods for global optimization." Sandia National Laboratories (SNL), Albuquerque, NM, and Livermore, CA ..., Tech. Rep., 2005.

[32] F. Capitanescu, "Critical review of recent advances and further developments needed in ac optimal power flow," *Electric Power Systems Research*, vol. 136, pp. 57–68, 2016.