Differentiable Optimization for Deep Learning-Enhanced DC Approximation of AC Optimal Power Flow

Andrew Rosemberg
NSF AI Institute for Advances in Optimization
Georgia Institute of Technology
Atlanta, GA
arosemberg3@gatech.edu

Michael Klamkin
NSF AI Institute for Advances in Optimization
Georgia Institute of Technology
Atlanta, GA
klam@gatech.edu

Pascal Van Hentenryck
NSF AI Institute for Advances in Optimization
Georgia Institute of Technology
Atlanta, GA
pvh@gatech.edu

Abstract

The growing scale of power systems and the increasing uncertainty introduced by renewable energy sources necessitates novel optimization techniques that are significantly faster and more accurate than existing methods. The AC Optimal Power Flow (AC-OPF) problem, a core component of power grid optimization, is often approximated using linearized DC Optimal Power Flow (DC-OPF) models for computational tractability, albeit at the cost of suboptimal and inefficient decisions. To address these limitations, we propose a novel deep learning-based framework for network equivalency that enhances DC-OPF to more closely mimic the behavior of AC-OPF. The approach utilizes recent advances in differentiable optimization, incorporating a neural network trained to predict adjusted nodal shunt conductances and branch susceptances in order to account for nonlinear power flow behavior. The model can be trained end-to-end using modern deep learning frameworks by leveraging the implicit function theorem. Results demonstrate the framework's ability to significantly improve prediction accuracy.

1 Introduction

Power systems optimization has gained significant attention in recent years, driven by growing emphasis on expansion, the integration of renewable energy sources, and the general need for cleaner, more sustainable energy solutions [1–4]. A fundamental component of optimal power grid control is the constrained optimization problem known as AC Optimal Power Flow (AC-OPF, Model 3, see Table 1 for notations), which includes the nonconvex AC Power Flow (AC-PF) equations

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: MLxOR: Mathematical Foundations and Operational Integration of Machine Learning for Uncertainty-Aware Decision-Making.

(3b)−(3g) to model how power is transmitted through the grid's transmission lines and transformers, engineering constraints such as transmission line thermal limits (3h)−(3i), and minimizes total power generation cost (3a). Obtaining accurate OPF solutions is critical to many downstream tasks across time scales, including real-time risk-aware market clearing [5, 6], day-ahead security-constrained unit commitment (SCUC) [7], transmission switching optimization [8], and expansion planning [9]. The main challenge precluding more widespread use of AC-OPF in practice lies in the non-linear and non-convex nature of the physics and engineering constraints, making solving time scale poorly as network size and uncertainty grows. Consequently, AC-(O)PF is often not directly utilized in practice [10]. Instead, transmission system operators (TSOs) opt for simplified problems such as DC-(O)PF [10, 11], where assumptions are made to approximate the physics using only linear functions, facilitating the use of more computationally efficient convex optimization techniques. While this approach is tractable, the approximations currently in use lead to sub-optimal decisions that incur significant operational inefficiency and thus increased cost [12].

Recent research has focused on leveraging deep learning to address these challenges. Most approaches (as comprehensively reviewed in [13]) employ "proxy models" — surrogate models trained via supervised learning, often with additional loss function terms to reduce constraint violations. These models directly approximate the power demand to optimal solution mapping. While such models have shown promising results, achieving high-fidelity predictions with minimal constraint violations for AC-OPF [14–17], they typically lack guarantees of optimality or feasibility, particularly when applied to scenarios outside their training distribution. This limitation poses a significant challenge in ensuring reliable operations in the face of uncertainty.

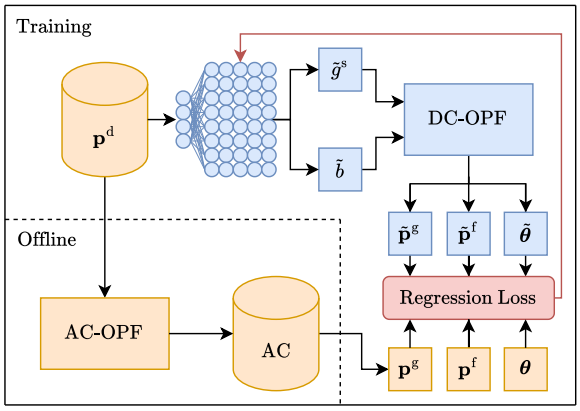


Figure 1: The proposed framework for deep learning-based DC-OPF adjustment.

This work addresses this challenge by proposing a novel deep learning framework that leverages recent advances in the differentiation of constrained optimization with respect to problem parameters [18] in order to *learn how to conditionally modify the problem data of DC-OPF such that its optimal solution mimics locally the behavior of AC-OPF*. Figure 1 summarizes the approach. First, a dataset of power demands $\mathbf{p}^{\mathrm{d}}$ is sampled and the corresponding AC-OPF solutions are obtained using a non-linear programming solver such as Ipopt [19]. The neural network model is created by appending a differentiable DC-OPF layer to a feed-forward neural network that predicts model parameters. The feed-forward portion of the neural network takes as input the power demand $\mathbf{p}^{\mathrm{d}}$ and outputs two vectors: adjusted nodal shunt conductances $\tilde{g}^{\mathrm{s}}$ and adjusted branch susceptances $\tilde{b}$, which are then used to formulate an "adjusted" DC-OPF. The final layer solves this adjusted DC-OPF to obtain its optimal solution $\tilde{\mathbf{p}}^{\mathrm{g}}$, $\tilde{\mathbf{p}}^{\mathrm{f}}$, $\tilde{\boldsymbol{\theta}}$ which is the overall model's prediction. Adjusting g^{s} and b allows the model to account for power losses and variations in effective line impedance that would otherwise be neglected in a standard DC-OPF formulation (which assumes loss-less power flow and linearized Kirchhoff laws), making it possible for the adjusted DC-OPF to better approximate the nonlinear characteristics of AC-OPF while retaining interpretability.

2 Related Work

The need for accurate yet computationally tractable models for power system optimization is well-documented. Particularly, Rosemberg et al. [12] underscores the economic consequences of relying on simplified models, examining the trade-offs associated with convex relaxations and approximations in hydrothermal dispatch planning. Notwithstanding, the Federal Energy Regulatory Commission (FERC) report [10] details current practices in the U.S., where transmission system operators often adopt linear relaxations, despite their potential to yield infeasible or inefficient solutions.

Classical approaches to power system reduction trace back to [20], which introduced network reduction models that remain foundational in simplifying power systems. However, these methods lack the flexibility needed to address the variability and uncertainty inherent in modern, renewable-integrated grids. Recent works have proposed new methods to overcome these limitations. For example, [21] explores network reduction models using nonlinear basis functions, leveraging the fact that basis function fitting can be formulated as linear constraints, offering a computationally efficient alternative. Additionally, [13] provides a comprehensive review of deep learning applications in OPF, highlighting the growing interest in data-driven approaches that aim to enhance the scalability and robustness of OPF solutions. Constante-Flores et al. [22] proposes an approach that first solves the bilevel problem of finding the best DC-OPF parameters to exactly match AC-OPF solutions for each instance in a dataset, then fitting a neural network to map AC-OPF loads to these parameters. While this approach minimizes the error between the AC and DC formulations, the requirement for exact bilevel solutions severely limits scalability.

3 Technical Approach

Problem Setup The learning problem can be posed as finding the neural network weights ω that approximately solve the bilevel optimization problem:

$$\underset{\omega}{\operatorname{argmin}}\ \mathbb{E}_{\mathbf{p}^{\mathrm{d}}}\left[\mathrm{MSELoss}\left(\begin{bmatrix}\mathbf{p}^{\mathrm{g}}\\ \mathbf{p}^{\mathrm{f}}\\ \boldsymbol{\theta}\end{bmatrix},\ \begin{bmatrix}\mathbf{p}^{\mathrm{g}}_{\mathrm{AC}}\\ \mathbf{p}^{\mathrm{f}}_{\mathrm{AC}}\\ \boldsymbol{\theta}_{\mathrm{AC}}\end{bmatrix}\right)\quad \text{where}\ \begin{bmatrix}\mathbf{p}^{\mathrm{g}}\\ \mathbf{p}^{\mathrm{f}}\\ \boldsymbol{\theta}\end{bmatrix}\in \begin{array}{r}\operatorname{argmin}\ \mathrm{DC\text{-}OPF}(\mathbf{p}^{\mathrm{d}};\ g^{\mathrm{s}},\ b)\\ \text{s.t.}\quad g^{\mathrm{s}},\ b=\mathrm{NN}_{\omega}(\mathbf{p}^{\mathrm{d}})\end{array}\right]$$

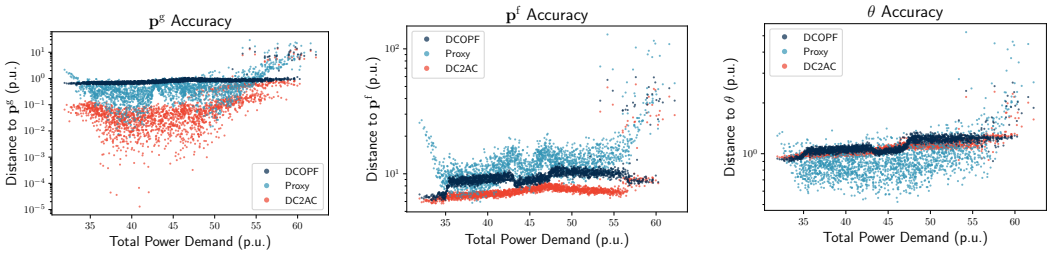
In order to find the mapping from power demand to optimally adjusted nodal shunt conductances and adjusted branch susceptances, $\mathbf{p}^{\mathrm{d}} \to (\tilde{g}^{\mathrm{s}}, \tilde{b})$, one needs to solve this complicated bilevel problem, i.e. taking into account the optimal solution of the lower level problem (DC-OPF) for any possible choice of upper level solution. The optimality of the lower level can be characterized using the Karush-Kuhn-Tucker (KKT) conditions which consist of guaranteeing: (a) Primal Feasibility, (b) Stationarity, (c) Dual Feasibility, and (d) Complementary Slackness. The exact mathematical forms of these conditions for DC-OPF is included in A.1.

Offline AC-OPF Data Generation The proposed method is a "semi-self-supervised" learning approach; it relies on a dataset of power demands and corresponding AC-OPF solutions, but solves DC-OPF in-the-loop. Thus, before training begins, a number of AC-OPF instances must be solved offline. These will be used in an ordinary regression loss, minimizing the distance between the adjusted DC-OPF optimal solution and the AC-OPF optimal solution.

Differentiable Optimization Layers The core of the proposed method is the use of a differentiable optimization layer to enable backpropagation through the solving of the DC-OPF, then differentiating the solution with respect to problem parameters which were predicted by the feed-forward neural network portion. Although automatic differentiation can in principle be used to differentiate through the iterations of an LP solver, this technique scales poorly with the number of iterations, resulting in massive computational graphs. It is also cumbersome to implement in practice since most modern solvers do not natively integrate with automatic differentiation systems. Instead of differentiating through the solution iterates, the differentiable optimization layer approach uses the implicit function theorem to solve for the gradient of the solution map with respect to problem data in terms of the optimal solution, which is obtained in the forward pass by calling a solver [18, 23].

Specifically, the sensitivities are calculated by applying the implicit function theorem applied to the KKT conditions of the optimization problem. These KKT conditions encapsulate the solution of the DC-OPF problem as a system of equations, i.e. $F(\mathbf{x}, \phi) = 0$, where $\mathbf{x} = (\mathbf{p}^{\mathrm{g}}, \mathbf{p}^{\mathrm{f}}, \boldsymbol{\theta}, \lambda)$ represents the decision variables of the OPF, including power generation ($\mathbf{p}^{\mathrm{g}}$), power flow ($\mathbf{p}^{\mathrm{f}}$), voltage angles ($\boldsymbol{\theta}$), and Lagrange multipliers (λ); and $\phi = (\tilde{g}^{\mathrm{s}}, \tilde{b})$ represents the adjustable problem data: nodal shunt conductances and branch susceptances. The exact KKT conditions are given in Section A.1.

This capability allows for the proposed framework to focus on learning how to account for the nonlinearities only rather than also trying to learn how to solve the DC-OPF itself (an end-to-end proxy has to learn both simultaneously). This is achieved by having the neural network predict DC-OPF problem data (nodal shunt conductance and branch susceptance) then using a linear optimization solver to obtain the corresponding DC-OPF solution – rather than predicting the solution directly.



(a) $\|\tilde{\mathbf{p}}^{\mathrm{g}} - \mathbf{p}^{\mathrm{g}}\|_1$ (b) $\|\tilde{\mathbf{p}}^{\mathrm{f}} - \mathbf{p}^{\mathrm{f}}\|_1$ (c) $\|\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta}\|_1$

Figure 2: Accuracy vs. total demand across methods.

Baseline Methods The proposed method is compared to the standard proxy approach [13] where a feed-forward neural network is trained using MSE loss to learn the map from power demand to AC-OPF solutions directly. The proxy and the proposed method are also compared to using the DC-OPF solution directly. The methods are evaluated using solution accuracy; that is, the L_1 distance between the predicted $\tilde{\mathbf{p}}^{\mathrm{g}}$, $\tilde{\mathbf{p}}^{\mathrm{f}}$, $\tilde{\boldsymbol{\theta}}$ and the true (i.e. found by Ipopt [19]) $\mathbf{p}^{\mathrm{g}}$, $\mathbf{p}^{\mathrm{f}}$, $\boldsymbol{\theta}$.

4 Results

This section includes experimental results comparing the baseline optimization proxy method, the current approach of using the DC-OPF solution itself, and the proposed neural-adjusted DC-OPF (hereafter referred to as DC2AC). Additional details on experiment settings are given in Section B.1

4.1 Accuracy Analysis

Table 2 summarizes the validation set accuracy of each approach, broken down by type of decision variable. Evidently, the DC2AC approach outperforms both baselines for $\mathbf{p}^{\mathrm{g}}$ and $\mathbf{p}^{\mathrm{f}}$, though the proxy is the best for $\boldsymbol{\theta}$. The columns of Table 2 are further visualized in Figure 2.

Notably, across all decision variables, all methods struggle in the high-load region corresponding to the power system being congested. Interestingly, the DC-OPF often outperforms the proxy in the high-load region, and DC2AC typically improves slightly on DC-OPF. For $\mathbf{p}^{\mathrm{g}}$ and $\mathbf{p}^{\mathrm{f}}$, especially in the lower-load portion of the domain, DC2AC is usually the best method of the three, achieving the lowest errors compared to the true AC solution. Overall, DC2AC outperforms all other methods on 95% of the validation set samples for $\mathbf{p}^{\mathrm{g}}$ and 97% of samples for $\mathbf{p}^{\mathrm{f}}$, but only 13% of samples for $\boldsymbol{\theta}$. Remarkably, DC2AC is able to almost exactly reproduce the AC-optimal $\mathbf{p}^{\mathrm{g}}$ in mid-range cases, achieving errors as low as 10^{-5}. The relative flexibility of the proxy is shown to help to learn the more complicated $\boldsymbol{\theta}$ variables, with DC2AC struggling to significantly outperform the vanilla DC-OPF. We hypothesize that this is due to the fact that the DC-OPF formulation we use neglects reactive power [24, Appendix A.2.3], a key limitation that is, evidently, hard to overcome for DC2AC.

Figure 3 shows the convergence plots for the proxy and DC2AC learning approaches, where solid lines denote validation loss and dashed lines denote training loss. Note that the networks have identical initialization, and that the DC2AC training was stopped early due to lack of improvement. The figure shows that DC2AC converges extremely quickly, mostly within the first and second epochs.

5 Conclusion

This paper introduced DC2AC, an interpretable deep learning approach to approximate AC-OPF solutions. The core idea is to use a differentiable optimization layer to solve a parametrized DC-OPF in the forward pass, leveraging the implicit function theorem to enable end-to-end training. Overall, the results show that the DC2AC approach is promising, with clear directions for further improvement. We hypothesize that since output of DC2AC is directly the (parametrized) DC-OPF solution, it struggles to capture the active-reactive power relationship, explaining the poor performance on the $\boldsymbol{\theta}$ variables. Future work includes exploring the use of different approximations of AC-OPF to address this and integration with GPU solvers [25] to speed up training.

Acknowledgements

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-2039655 and the National Science Foundation Artificial Intelligence Institute for Advances in Optimization under Grant No. 2112533. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] Alessandro Soares, Alexandre Street, Tiago Andrade, and Joaquim Dias Garcia. An integrated progressive hedging and benders decomposition with multiple master method to solve the brazilian generation expansion problem. *IEEE Transactions on Power Systems*, 37(5):4017–4027, 2022.

[2] David Pozo, Enzo E Sauma, and Javier Contreras. A three-level static milp model for generation and transmission expansion planning. *IEEE Transactions on Power systems*, 28(1):202–210, 2012.

[3] David Pozo and Javier Contreras. A chance-constrained unit commitment with an nk security criterion and significant wind generation. *IEEE Transactions on Power systems*, 28(3):2842–2851, 2012.

[4] Neil Barry, Minas Chatzos, Wenbo Chen, Dahye Han, Chaofan Huang, Roshan Joseph, Michael Klamkin, Seonho Park, Mathieu Tanneau, Pascal Van Hentenryck, et al. Risk-aware control and optimization for high-renewable power grids. *arXiv preprint arXiv:2204.00950*, 2022.

[5] Simon Tam. Real-time security-constrained economic dispatch and commitment in the pjm: Experiences and challenges. In *FERC Software Conference*, 2011.

[6] Wenbo Chen. End-to-end feasible optimization proxies for large-scale economic dispatch. INFORMS, 2023.

[7] Xiaorong Sun, Peter B. Luh, Mikhail A. Bragin, Yonghong Chen, Fengyu Wang, and Jie Wan. A decomposition and coordination approach for large-scale security constrained unit commitment problems with combined cycle units. In *IEEE Power & Energy Society*, pages 1–5, 2017.

[8] E. B. Fisher, R. P. O'Neill, and M. C. Ferris. Optimal transmission switching. *IEEE Transactions on Power Systems (TPWRS)*, 23(3):1346–1355, Aug 2008.

[9] Sumit Verma, Vivekananda Mukherjee, et al. Transmission expansion planning: A review. In *3rd International Conference on Energy Efficient Technologies for Sustainability (ICEETS 2016)*, pages 350–355. IEEE, 2016.

[10] Richard P O'Neill, Thomas Dautel, and Eric Krall. Recent iso software enhancements and future software and modeling plans. *Federal Energy Regulatory Commission, Tech. Rep*, 2011.

[11] Brent Eldridge, Richard P O'Neill, and Andrea R Castillo. Marginal loss calculations for the dcopf. Technical report, Sandia National Lab.(SNL-NM), Albuquerque, NM (United States), 2016.

[12] Andrew W Rosemberg, Alexandre Street, Joaquim Dias Garcia, Davi M Valladão, Thuener Silva, and Oscar Dowson. Assessing the cost of network simplifications in long-term hydrothermal dispatch planning models. *IEEE Transactions on Sustainable Energy*, 13(1):196–206, 2021.

[13] Hooman Khaloie, Mihaly Dolanyi, Jean-Francois Toubeau, and François Vallée. Review of machine learning techniques for optimal power flow. *Available at SSRN 4681955*, May 2024. doi: 10.2139/ssrn.4681955.

[14] Pascal Van Hentenryck. Machine learning for optimal power flows. *Tutorials in Operations Research: Emerging Optimization Methods and Modeling Techniques with Applications*, pages 62–82, 2021.

[15] Wenbo Chen, Seonho Park, Mathieu Tanneau, and Pascal Van Hentenryck. Learning optimization proxies for large-scale security-constrained economic dispatch. *Electric Power Systems Research*, 213:108566, 2022.

[16] Luis Piloto, Sofia Liguori, Sephora Madjiheurem, Miha Zgubic, Sean Lovett, Hamish Tomlinson, Sophie Elster, Chris Apps, and Sims Witherspoon. Canos: A fast and scalable neural ac-opf solver robust to n-1 perturbations. *arXiv preprint arXiv:2403.17660*, 2024.

[17] Michael Klamkin, Mathieu Tanneau, Terrence WK Mak, and Pascal Van Hentenryck. Bucketized active sampling for learning acopf. *Electric Power Systems Research*, 235:110697, 2024.

[18] Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, and J Zico Kolter. Differentiable convex optimization layers. *Advances in neural information processing systems*, 32, 2019.

[19] Lorenz T Biegler and Victor M Zavala. Large-scale nonlinear programming using Ipopt: An integrating framework for enterprise-wide dynamic optimization. *Computers & Chemical Engineering*, 33(3):575–582, 2009.

[20] James B Ward. *Equivalent Circuits in Power System Studies*. Purdue University, 1949.

[21] Raul Ribeiro, Alexandre Street, Fernando Mancilla-David, and Alejandro Angulo. Equivalent reduced dc network models with nonlinear load functions: A data-driven approach. *IEEE Transactions on Power Systems*, 39(2):3021–3032, 2023.

[22] Gonzalo E Constante-Flores, André H Quisaguano, Antonio J Conejo, and Can Li. Ac-network-informed dc optimal power flow for electricity markets. *arXiv preprint arXiv:2410.18413*, 2024.

[23] Mathieu Besançon, Joaquim Dias Garcia, Benoît Legat, and Akshay Sharma. Flexible differentiable optimization via model transformations. *INFORMS Journal on Computing*, 2022. doi: 10.1287/ijoc.2022.0283.

[24] Michael Klamkin, Mathieu Tanneau, and Pascal Van Hentenryck. PGLearn–an open-source learning toolkit for optimal power flow. *arXiv preprint arXiv:2505.22825*, 2025.

[25] Sungho Shin, François Pacaud, and Mihai Anitescu. Accelerating optimal power flow with GPUs: SIMD abstraction of nonlinear programs and condensed-space interior-point methods. *arXiv preprint arXiv:2307.16830*, 2023.

[26] PACE. *Partnership for an Advanced Computing Environment (PACE)*, 2017. URL `http://www.pace.gatech.edu`.

[27] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[28] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM Review*, 59(1):65–98, 2017. doi: 10.1137/141000671.

[29] Miles Lubin, Oscar Dowson, Joaquim Dias Garcia, Joey Huchette, Benoît Legat, and Juan Pablo Vielma. JuMP 1.0: Recent improvements to a modeling language for mathematical optimization. *Mathematical Programming Computation*, 2023. doi: 10.1007/s12532-023-00239-3.

[30] Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, Geeta Chauhan, Anjali Chourdia, Will Constable, Alban Desmaison, Zachary DeVito, Elias Ellison, Will Feng, Jiong Gong, Michael Gschwind, Brian Hirsh, Sherlock Huang, Kshiteej Kalambarkar, Laurent Kirsch, Michael Lazos, Mario Lezcano, Yanbo Liang, Jason Liang, Yinghai Lu, CK Luk, Bert Maher, Yunjie Pan, Christian Puhrsch, Matthias Reso, Mark Saroufim, Marcos Yukio Siraichi, Helen Suk, Michael Suo, Phil Tillet, Eikan Wang, Xiaodong Wang, William Wen, Shunting Zhang, Xu Zhao, Keren Zhou, Richard Zou, Ajit Mathews, Gregory Chanan, Peng Wu, and Soumith Chintala. PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph

Compilation. In *29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2 (ASPLOS '24)*. ACM, April 2024. doi: 10.1145/3620665.3640366.

[31] Michael Klamkin. `juliafunction`: Batch-distributed differentiable Julia functions in PyTorch. `https://github.com/klamike/juliafunction`, 2024.

[32] Q. Huangfu and J. A. J. Hall. Parallelizing the dual revised simplex method. *Mathematical Programming Computation*, 10(1):119–142, 2018. doi: 10.1007/s12532-017-0130-5.

[33] Stéphane Fliscounakis, Patrick Panciatici, Florin Capitanescu, and Louis Wehenkel. Contingency ranking with respect to overloads in very large power systems taking into account uncertainty, preventive, and corrective actions. *IEEE Transactions on Power Systems*, 28(4): 4909–4917, 2013.

A Formulations and Notations

This section includes the detailed mathematical programming formulations referenced throughout the paper. Table 1 summarizes the notations, Model 1 presents the DC-OPF, 3 presents the AC-OPF, and 2 presents the dual of DC-OPF. The KKT conditions of DC-OPF are also included in section A.1.

Table 1: OPF Data Notations

Description	Size	Symbol
Set of buses	–	$\mathcal{N}$
Set of edges	–	$\mathcal{E}$
Set of loads	–	$\mathcal{L}$
Set of generators	–	$\mathcal{G}$
Reference bus voltage angle	1	$\boldsymbol{\theta}_{\text{ref}}$
Active power load	$\lvert\mathcal{L}\rvert$	$\mathbf{p}^{\text{d}}$
Reactive power load	$\lvert\mathcal{L}\rvert$	$\mathbf{q}^{\text{d}}$
Active power generation	$\lvert\mathcal{G}\rvert$	$\mathbf{p}^{\text{g}}$
Reactive power generation	$\lvert\mathcal{G}\rvert$	$\mathbf{p}^{\text{g}}$
Active power flow from	$\lvert\mathcal{E}\rvert$	$\mathbf{p}^{\text{f}}$
Active power flow to	$\lvert\mathcal{E}\rvert$	$\mathbf{p}^{\text{t}}$
Reactive power flow from	$\lvert\mathcal{E}\rvert$	$\mathbf{q}^{\text{f}}$
Reactive power flow to	$\lvert\mathcal{E}\rvert$	$\mathbf{q}^{\text{t}}$
Voltage magnitude	$\lvert\mathcal{N}\rvert$	$\mathbf{v}$
Voltage angle	$\lvert\mathcal{N}\rvert$	$\boldsymbol{\theta}$
Nodal load shedding	$\lvert\mathcal{N}\rvert$	φ
Indices of branches leaving each bus	$\lvert\mathcal{N}\rvert$	$\mathcal{E}_i$
Indices of branches entering each bus	$\lvert\mathcal{N}\rvert$	$\mathcal{E}_i^R$
Indices of generators at each bus	$\lvert\mathcal{N}\rvert$	$\mathcal{G}_i$
Indices of loads at each bus	$\lvert\mathcal{N}\rvert$	$\mathcal{L}_i$
Nodal shunt conductance	$\lvert\mathcal{N}\rvert$	g^{s}
Nodal shunt susceptance	$\lvert\mathcal{N}\rvert$	b^{s}
Voltage magnitude lower bound	$\lvert\mathcal{N}\rvert$	$\underline{\mathbf{v}}$
Voltage magnitude upper bound	$\lvert\mathcal{N}\rvert$	$\overline{\mathbf{v}}$
Minimum voltage angle difference	$\lvert\mathcal{E}\rvert$	$\underline{\Delta}\theta$
Maximum voltage angle difference	$\lvert\mathcal{E}\rvert$	$\overline{\Delta}\theta$
Branch thermal limit	$\lvert\mathcal{E}\rvert$	$\overline{S}$
Minimum active power generation	$\lvert\mathcal{G}\rvert$	$\underline{\mathbf{p}}^{\text{g}}$
Maximum active power generation	$\lvert\mathcal{G}\rvert$	$\overline{\mathbf{p}}^{\text{g}}$
Minimum reactive power generation	$\lvert\mathcal{G}\rvert$	$\underline{\mathbf{q}}^{\text{g}}$
Maximum reactive power generation	$\lvert\mathcal{G}\rvert$	$\overline{\mathbf{q}}^{\text{g}}$
Generator cost coefficient	$\lvert\mathcal{G}\rvert$	c
Load shedding cost	1	c_{φ}
From bus index for each branch	$\lvert\mathcal{E}\rvert$	i
To bus index for each branch	$\lvert\mathcal{E}\rvert$	j
Branch conductance	$\lvert\mathcal{E}\rvert$	g
Branch susceptance	$\lvert\mathcal{E}\rvert$	b
From-side branch conductance	$\lvert\mathcal{E}\rvert$	g^{ff}
From-to branch conductance	$\lvert\mathcal{E}\rvert$	g^{ft}
To-from branch conductance	$\lvert\mathcal{E}\rvert$	g^{tf}
To-side branch conductance	$\lvert\mathcal{E}\rvert$	g^{tt}
From-side branch susceptance	$\lvert\mathcal{E}\rvert$	b^{ff}
From-to branch susceptance	$\lvert\mathcal{E}\rvert$	b^{ft}
To-from branch susceptance	$\lvert\mathcal{E}\rvert$	b^{tf}
To-side branch susceptance	$\lvert\mathcal{E}\rvert$	b^{tt}

$$
\begin{aligned}
\min_{\mathbf{p}^{\mathrm{g}},\mathbf{p}^{\mathrm{f}},\boldsymbol{\theta},\boldsymbol{\varphi}} \quad & \sum_{i\in\mathcal{N}} c_{\varphi}\varphi_i + \sum_{j\in\mathcal{G}_i} c_j \mathbf{p}_j^{\mathrm{g}} \\
\text{s.t.} \quad & \sum_{j\in\mathcal{G}_i} \mathbf{p}_j^{\mathrm{g}} - \sum_{e\in\mathcal{E}_i} \mathbf{p}_e^{\mathrm{f}} + \sum_{e\in\mathcal{E}_i^R} \mathbf{p}_e^{\mathrm{f}} + \varphi_i = \sum_{j\in\mathcal{L}_i} \mathbf{p}_j^{\mathrm{d}} + \boxed{g_i^{\mathrm{s}}} && \forall i\in\mathcal{N} && [\lambda^{\mathrm{p}}] \\
& \mathbf{p}_e^{\mathrm{f}} = -\boxed{b_e}(\boldsymbol{\theta}_i - \boldsymbol{\theta}_j) && \forall e=(i,j)\in\mathcal{E} && [\lambda^{\mathrm{pf}}] \\
& \underline{\Delta}\theta_e \le \boldsymbol{\theta}_i - \boldsymbol{\theta}_j \le \overline{\Delta}\theta_e && \forall e=(i,j)\in\mathcal{E} && [\underline{\mu}^{\theta}, \bar{\mu}^{\theta}] \\
& \boldsymbol{\theta}_{\mathrm{ref}} = 0 \\
& \underline{\mathbf{p}_i^{\mathrm{g}}} \le \mathbf{p}_i^{\mathrm{g}} \le \overline{\mathbf{p}_i^{\mathrm{g}}} && \forall i\in\mathcal{G} && [\underline{\mu}^{\mathrm{pg}}, \bar{\mu}^{\mathrm{pg}}] \\
& -\overline{S_e} \le \mathbf{p}_e^{\mathrm{f}} \le \overline{S_e} && \forall e\in\mathcal{E} && [\underline{\mu}^{\mathrm{pf}}, \bar{\mu}^{\mathrm{pf}}]
\end{aligned}
$$

Model 1: DC Optimal Power Flow (DC-OPF)

$$
\begin{aligned}
\max_{\substack{\lambda^{\mathrm{p}},\lambda^{\mathrm{pf}},\underline{\mu}^{\theta},\bar{\mu}^{\theta},\\ \underline{\mu}^{\mathrm{pg}},\bar{\mu}^{\mathrm{pg}},\underline{\mu}^{\mathrm{pf}},\bar{\mu}^{\mathrm{pf}}}} \quad & \sum_{i\in\mathcal{N}} \lambda_i^{\mathrm{p}}\Big(\boxed{g_i^{\mathrm{s}}} + \varphi_i + \sum_{j\in\mathcal{L}_i} \mathbf{p}_j^{\mathrm{d}}\Big) + \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{G}_i} \Big(\underline{\mathbf{p}}_j^{\mathrm{g}}\underline{\mu}_j^{\mathrm{pg}} - \overline{\mathbf{p}}_j^{\mathrm{g}}\bar{\mu}_j^{\mathrm{pg}}\Big) \\
& + \sum_{e\in\mathcal{E}} \Big(\underline{\Delta}\theta_e\underline{\mu}_e^{\theta} - \overline{\Delta}\theta_e\bar{\mu}_e^{\theta} - \bar{s}_e\underline{\mu}_e^{\mathrm{pf}} - \bar{s}_e\bar{\mu}_e^{\mathrm{pf}}\Big) && && (2a) \\
\text{s.t.} \quad & \lambda_i^{\mathrm{p}} + \underline{\mu}_g^{\mathrm{pg}} - \bar{\mu}_g^{\mathrm{pg}} = c_g && \forall i\in\mathcal{N}, \forall g\in\mathcal{G}_i && (2b) \\
& -\lambda_i^{\mathrm{p}} + \lambda_j^{\mathrm{p}} - \lambda_e^{\mathrm{pf}} + \underline{\mu}_e^{\mathrm{pf}} - \bar{\mu}_e^{\mathrm{pf}} = 0 && \forall e=(i,j)\in\mathcal{E} && (2c) \\
& \sum_{e\in\mathcal{E}_i^+} \Big(\underline{\mu}_e^{\theta} - \boxed{b_e}\lambda_e^{\mathrm{pf}}\Big) + \sum_{e\in\mathcal{E}_i^-} \Big(\boxed{b_e}\lambda_e^{\mathrm{pf}} - \bar{\mu}_e^{\theta}\Big) = 0 && \forall i\in\mathcal{N} && (2d) \\
& \underline{\mu}^{\theta}, \underline{\mu}^{\mathrm{pg}}, \underline{\mu}^{\mathrm{pf}} \ge 0 && && (2e) \\
& \bar{\mu}^{\theta}, \bar{\mu}^{\mathrm{pg}}, \bar{\mu}^{\mathrm{pf}} \ge 0 && && (2f)
\end{aligned}
$$

Model 2: Dual of DC-OPF

$$
\begin{aligned}
\min_{\substack{\mathbf{p}^{\mathrm{g}},\mathbf{q}^{\mathrm{g}},\mathbf{v},\boldsymbol{\theta}\\ \mathbf{p}^{\mathrm{f}},\mathbf{q}^{\mathrm{f}},\mathbf{p}^{\mathrm{t}},\mathbf{q}^{\mathrm{t}}}} \quad & \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{G}_i} c_j \mathbf{p}_j^{\mathrm{g}} && \text{(3a)}\\
\text{s.t.} \quad & \sum_{j\in\mathcal{G}_i}\mathbf{p}_j^{\mathrm{g}} - \sum_{j\in\mathcal{L}_i}\mathbf{p}_j^{\mathrm{d}} - g_i^{\mathrm{s}}\mathbf{v}_i^2 = \sum_{e\in\mathcal{E}_i}\mathbf{p}_e^{\mathrm{f}} + \sum_{e\in\mathcal{E}_i^R}\mathbf{p}_e^{\mathrm{t}} && \forall i\in\mathcal{N} \quad \text{(3b)}\\
& \sum_{j\in\mathcal{G}_i}\mathbf{q}_j^{\mathrm{g}} - \sum_{j\in\mathcal{L}_i}\mathbf{q}_j^{\mathrm{d}} + b_i^{\mathrm{s}}\mathbf{v}_i^2 = \sum_{e\in\mathcal{E}_i}\mathbf{q}_e^{\mathrm{f}} + \sum_{e\in\mathcal{E}_i^R}\mathbf{q}_e^{\mathrm{t}} && \forall i\in\mathcal{N} \quad \text{(3c)}\\
& \mathbf{p}_e^{\mathrm{f}} = g_e^{\mathrm{ff}}\mathbf{v}_i^2 + g_e^{\mathrm{ft}}\mathbf{v}_i\mathbf{v}_j\cos(\boldsymbol{\theta}_i-\boldsymbol{\theta}_j) + b_e^{\mathrm{ft}}\mathbf{v}_i\mathbf{v}_j\sin(\boldsymbol{\theta}_i-\boldsymbol{\theta}_j) && \forall e=(i,j)\in\mathcal{E} \quad \text{(3d)}\\
& \mathbf{q}_e^{\mathrm{f}} = -b_e^{\mathrm{ff}}\mathbf{v}_i^2 - b_e^{\mathrm{ft}}\mathbf{v}_i\mathbf{v}_j\cos(\boldsymbol{\theta}_i-\boldsymbol{\theta}_j) + g_e^{\mathrm{ft}}\mathbf{v}_i\mathbf{v}_j\sin(\boldsymbol{\theta}_i-\boldsymbol{\theta}_j) && \forall e=(i,j)\in\mathcal{E} \quad \text{(3e)}\\
& \mathbf{p}_e^{\mathrm{t}} = g_e^{\mathrm{tt}}\mathbf{v}_j^2 + g_e^{\mathrm{tf}}\mathbf{v}_i\mathbf{v}_j\cos(\boldsymbol{\theta}_i-\boldsymbol{\theta}_j) - b_e^{\mathrm{tf}}\mathbf{v}_i\mathbf{v}_j\sin(\boldsymbol{\theta}_i-\boldsymbol{\theta}_j) && \forall e=(i,j)\in\mathcal{E} \quad \text{(3f)}\\
& \mathbf{q}_e^{\mathrm{t}} = -b_e^{\mathrm{tt}}\mathbf{v}_j^2 - b_e^{\mathrm{tf}}\mathbf{v}_i\mathbf{v}_j\cos(\boldsymbol{\theta}_i-\boldsymbol{\theta}_j) - g_e^{\mathrm{tf}}\mathbf{v}_i\mathbf{v}_j\sin(\boldsymbol{\theta}_i-\boldsymbol{\theta}_j) && \forall e=(i,j)\in\mathcal{E} \quad \text{(3g)}\\
& (\mathbf{p}_e^{\mathrm{f}})^2 + (\mathbf{q}_e^{\mathrm{f}})^2 \le \overline{S_e}^2 && \forall e\in\mathcal{E} \quad \text{(3h)}\\
& (\mathbf{p}_e^{\mathrm{t}})^2 + (\mathbf{q}_e^{\mathrm{t}})^2 \le \overline{S_e}^2 && \forall e\in\mathcal{E} \quad \text{(3i)}\\
& \underline{\Delta}\theta_e \le \boldsymbol{\theta}_i - \boldsymbol{\theta}_j \le \overline{\Delta}\theta_e && \forall e=(i,j)\in\mathcal{E} \quad \text{(3j)}\\
& \boldsymbol{\theta}_{\text{ref}} = 0 && \text{(3k)}\\
& \underline{\mathbf{v}_i} \le \mathbf{v}_i \le \overline{\mathbf{v}_i} && \forall i\in\mathcal{N} \quad \text{(3l)}\\
& \underline{\mathbf{p}}_i^{\mathrm{g}} \le \mathbf{p}_i^{\mathrm{g}} \le \overline{\mathbf{p}}_i^{\mathrm{g}},\ \underline{\mathbf{q}}_i^{\mathrm{g}} \le \mathbf{q}_i^{\mathrm{g}} \le \overline{\mathbf{q}}_i^{\mathrm{g}} && \forall i\in\mathcal{G} \quad \text{(3m)}\\
& -\overline{S_e} \le \mathbf{p}_e^{\mathrm{f}} \le \overline{S_e},\ -\overline{S_e} \le \mathbf{q}_e^{\mathrm{f}} \le \overline{S_e},\ -\overline{S_e} \le \mathbf{p}_e^{\mathrm{t}} \le \overline{S_e},\ -\overline{S_e} \le \mathbf{q}_e^{\mathrm{t}} \le \overline{S_e} && \forall e\in\mathcal{E} \quad \text{(3n)}
\end{aligned}
$$

Model 3: AC Optimal Power Flow (AC-OPF)

A.1 DC-OPF KKT conditions

(a) Primal Feasibility: These conditions ensure that the primal problem constraints are satisfied:

$$
\begin{aligned}
&\sum_{j\in\mathcal{G}_i}\mathbf{p}^{\mathrm{g}}_j - \sum_{e\in\mathcal{E}_i}\mathbf{p}^{\mathrm{f}}_e + \sum_{e\in\mathcal{E}^R_i}\mathbf{p}^{\mathrm{f}}_e + \varphi_i = \sum_{j\in\mathcal{L}_i}\mathbf{p}^{\mathrm{d}}_j + g^{\mathrm{s}}_i, && \forall i\in\mathcal{N}\\
&\mathbf{p}^{\mathrm{f}}_e = -b_e(\boldsymbol{\theta}_i - \boldsymbol{\theta}_j), && \forall e=(i,j)\in\mathcal{E}\\
&\underline{\Delta}\theta_e \le \boldsymbol{\theta}_i - \boldsymbol{\theta}_j \le \overline{\Delta}\theta_e, && \forall e=(i,j)\in\mathcal{E}\\
&\boldsymbol{\theta}_{\mathrm{ref}} = 0, && \text{(reference bus)}\\
&\underline{\mathbf{p}^{\mathrm{g}}_i} \le \mathbf{p}^{\mathrm{g}}_i \le \overline{\mathbf{p}^{\mathrm{g}}_i}, && \forall i\in\mathcal{G}\\
&-\overline{S_e} \le \mathbf{p}^{\mathrm{f}}_e \le \overline{S_e}, && \forall e\in\mathcal{E}
\end{aligned}
$$

(b) Stationarity: Stationarity sets the Lagrangian derivatives with respect to primal variables to zero:

$$
\begin{aligned}
&\frac{\partial\mathcal{L}}{\partial\mathbf{p}^{\mathrm{g}}_j} = c_j + \lambda^{\mathrm{p}}_i - \underline{\mu}^{\mathrm{pg}}_j + \bar{\mu}^{\mathrm{pg}}_j = 0 \quad \forall j\in\mathcal{G},\\
&\frac{\partial\mathcal{L}}{\partial\mathbf{p}^{\mathrm{f}}_e} = -\lambda^{\mathrm{p}}_i + \lambda^{\mathrm{p}}_j + \lambda^{\mathrm{pf}}_e - \underline{\mu}^{\mathrm{pf}}_e + \bar{\mu}^{\mathrm{pf}}_e = 0\\
&\qquad \forall e=(i,j)\in\mathcal{E},\\
&\frac{\partial\mathcal{L}}{\partial\boldsymbol{\theta}_i} = \sum_{e\in\mathcal{E}_i} b_e\lambda^{\mathrm{pf}}_e - \sum_{e\in\mathcal{E}^R_i} b_e\lambda^{\mathrm{pf}}_e + \sum_{e\in\mathcal{E}_i}(-\underline{\mu}^{\theta}_e + \bar{\mu}^{\theta}_e) = 0\\
&\qquad \forall i\in\mathcal{N},\\
&\frac{\partial\mathcal{L}}{\partial\varphi_i} = c_\varphi + \lambda^{\mathrm{p}}_i = 0 \quad \forall i\in\mathcal{N}.
\end{aligned}
$$

(c) Dual Feasibility: Dual variables associated with inequality constraints must remain non-negative:

$$
\begin{aligned}
&\underline{\mu}^{\mathrm{pg}}_j \ge 0, \quad \bar{\mu}^{\mathrm{pg}}_j \ge 0, && \forall j\in\mathcal{G}\\
&\underline{\mu}^{\mathrm{pf}}_e \ge 0, \quad \bar{\mu}^{\mathrm{pf}}_e \ge 0, && \forall e\in\mathcal{E}\\
&\underline{\mu}^{\theta}_e \ge 0, \quad \bar{\mu}^{\theta}_e \ge 0, && \forall e\in\mathcal{E}
\end{aligned}
$$

(d) Complementary Slackness: Complementary slackness ensures that either the inequality constraint is active or its corresponding dual variable is zero:

$$
\begin{aligned}
&\underline{\mu}^{\mathrm{pg}}_j \perp (\underline{\mathbf{p}^{\mathrm{g}}_j} - \mathbf{p}^{\mathrm{g}}_j), && \forall j\in\mathcal{G}\\
&\bar{\mu}^{\mathrm{pg}}_j \perp (\mathbf{p}^{\mathrm{g}}_j - \overline{\mathbf{p}^{\mathrm{g}}_j}), && \forall j\in\mathcal{G}\\
&\underline{\mu}^{\mathrm{pf}}_e \perp (-\overline{S_e} - \mathbf{p}^{\mathrm{f}}_e), && \forall e\in\mathcal{E}\\
&\bar{\mu}^{\mathrm{pf}}_e \perp (\mathbf{p}^{\mathrm{f}}_e - \overline{S_e}), && \forall e\in\mathcal{E}\\
&\underline{\mu}^{\theta}_e \perp (\underline{\Delta}\theta_e - (\boldsymbol{\theta}_i - \boldsymbol{\theta}_j)), && \forall e=(i,j)\in\mathcal{E}\\
&\bar{\mu}^{\theta}_e \perp ((\boldsymbol{\theta}_i - \boldsymbol{\theta}_j) - \overline{\Delta}\theta_e), && \forall e=(i,j)\in\mathcal{E}
\end{aligned}
$$

B Experiment Setup and Additional Results

This section includes details on experiment setup and additional results, namely the convergence plot in Figure 3 and prediction accuracies in Table 2.

B.1 Experiment Setup

The baseline and the proposed method both use a feed-forward neural network architecture with 3 layers each of width 64 and Softplus activations, a constant learning rate of 10^{-4}, and the Adam [27] optimizer. Bounds on predictions are enforced using a double-sided Softplus, i.e. $\text{softplus}(x - l) - \text{softplus}(x - u) + l$ to enforce $l \leq x \leq u$. The differentiable optimization layer implementation uses the Julia programming language [28]; its main dependencies are the JuMP modeling language [29] and the DiffOpt differentiable optimization framework [23]. The PGLearn.jl library [24] is used to build the JuMP model. The learning approaches are implemented in PyTorch [30] using ML4OPF [24], with DC2AC relying on Julia interface `juliafunction` [31] and the LP solver HiGHS [32].

B.1.1 Data Generation Procedure

This work uses PGLearn.jl [24] for data generation, sampling loads using both a global (one per sample) correlation factor as well as local noise (one per sample per load). Both are sampled from Uniform distributions; the local noise range is set to $\pm 15\%$ around the each load's reference value and the global range is set to $70\% - 110\%$. This ensures the dataset captures a wide range of operating conditions, especially in the challenging high-load regime. Results are provided for a dataset of 10,000 samples (80% training, 20% validation) generated around `89_pegase` [33], a benchmark based on the European power grid that has 89 buses (nodes), 35 loads, 12 generators, and 210 transmission lines/transformers (edges).

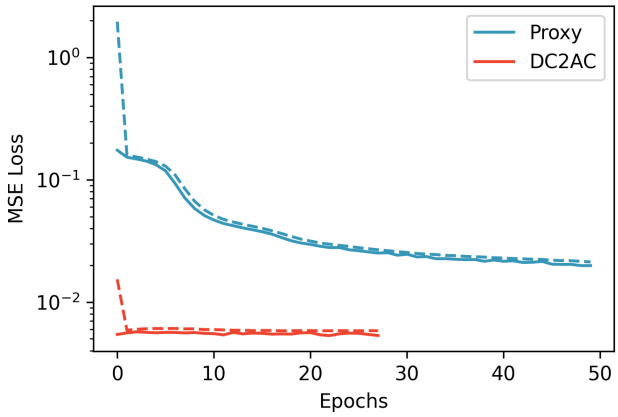


Figure 3: Training (dashed) and validation (solid) loss convergence throughout training for Proxy and DC2AC.

Table 2: Prediction accuracies compared to AC-OPF (per-unit).

Method	$\lVert\tilde{\mathbf{p}}^{\mathrm{g}} - \mathbf{p}^{\mathrm{g}}\rVert_1$	$\lVert\tilde{\mathbf{p}}^{\mathrm{f}} - \mathbf{p}^{\mathrm{f}}\rVert_1$	$\lVert\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta}\rVert_1$
DC-OPF	0.90	9.72	1.12
Proxy	0.74	12.83	**0.95**
DC2AC	**0.22**	**7.57**	1.08