# Learning Optimal Power Flow Value Functions with Input-Convex Neural Networks

Andrew Rosemberg*, Mathieu Tanneau*, Bruno Fanzeres†, Joaquim Garcia‡ and Pascal Van Hentenryck*
**Georgia Institute of Technology*, arosemberg3@gatech.edu, {mathieu.tanneau, pascal.vanhentenryck}@isye.gatech.edu
† *Industrial Engineering Department*, *Pontifical Catholic University of Rio de Janeiro*, bruno.santos@puc-rio.br
‡ Research and Development, PSR - Energy Consulting and Analytics, joaquimgarcia@psr-inc.com

***Abstract*—The Optimal Power Flow (OPF) problem is integral to the functioning of power systems, aiming to optimize generation dispatch while adhering to technical and operational constraints. These constraints are far from straightforward; they involve intricate, non-convex considerations related to Alternating Current (AC) power flow, which are essential for the safety and practicality of electrical grids. However, solving the OPF problem for varying conditions within stringent time-frames poses practical challenges. To address this, operators often resort to model simplifications of varying accuracy. Unfortunately, better approximations (tight convex relaxations) are often still computationally intractable. This research explores machine learning (ML) to learn convex approximate solutions for faster analysis in the online setting while still allowing for coupling into other convex dependent decision problems. By trading off a small amount of accuracy for substantial gains in speed, they enable the efficient exploration of vast solution spaces in these complex problems.**

***Index Terms*—Optimal Power Flow, Renewable Energy, Deep Learning, Convex Relaxations, Learn-to-Optimize.**

## Nomenclature

| | |
|---|---|
| $\mathcal{N}$ | Set of buses |
| $\mathcal{E}$ | Set of branches |
| $\mathcal{E}^R$ | Set of reverse branches |
| $\mathbf{j}$ | Imaginary unit $\mathbf{j}^2 = -1$ |
| $z^\star$ | Complex conjugate of $z$ |
| $\mathrm{S}^\mathrm{d}$ | Complex power demand; $\mathrm{S}^\mathrm{d} = \mathrm{p}^\mathrm{d} + \mathbf{j}\mathrm{q}^\mathrm{d}$ |
| $Y^s$ | Bus shunt admittance |
| $Y$ | Complex branch line admittance |
| $Y^c$ | Complex branch shunt admittance |
| $\bar{s}$ | Thermal branch limit |
| $\underline{\mathrm{p}}^\mathrm{g}, \overline{\mathrm{p}}^\mathrm{g}$ | Active power generation bounds |
| $\underline{\mathrm{q}}^\mathrm{g}, \overline{\mathrm{q}}^\mathrm{g}$ | Reactive power generation bounds |
| $\underline{\mathrm{v}}, \overline{\mathrm{v}}$ | Voltage magnitude bounds |
| $b$ | Branch susceptance |
| $\mathbf{S}^\mathrm{g}$ | Complex power generation; $\mathbf{S}^\mathrm{g} = \mathbf{p}^\mathrm{g} + \mathbf{j}\mathbf{q}^\mathrm{g}$ |
| $\mathbf{S}^\mathrm{f}$ | Complex power flow; $\mathbf{S}^\mathrm{f} = \mathbf{p}^\mathrm{f} + \mathbf{j}\mathbf{q}^\mathrm{f}$ |
| $\mathbf{V}$ | Complex voltage; $\mathbf{V} = \mathbf{v}\angle\theta$ |

This research is partly funded by NSF award 2112533.

## I. Introduction

The Optimal Power Flow (OPF) problem optimizes generation dispatch while satisfying physical and engineering constraints. It is therefore fundamental for many aspects of power systems operations: market-clearing, unit commitment, optimal transmission switching, transmission expansion planning, to name a few. Its alternating current (AC) form [1], [2], AC-OPF, is a nonlinear, non-convex problem which makes it challenging to solve in practice, especially when combined with discrete decisions like unit commitment, line switching, and bus splitting. Therefore, operators rely on approximate OPF formulations, typically the DC-OPF approximation which, although more tractable, may lead to sub-optimal or unsafe decisions when far from the traditional operating point, because it does not capture the complexity of AC systems.

These computational limitations have spurred interest in optimization proxies for power systems, and OPF problems in particular. Optimization proxies [3]–[5] are machine learning (ML) models that approximate the input-to-output mapping of an optimization solver; once trained, they produce predictions in milliseconds. A large body of work has focused on predicting solutions to OPF problems, especially DC-OPF and AC-OPF. In this case, the proxy takes the input data of the OPF as input and outputs a near-feasible, near-optimal solution. This enables real-time risk assessment at massive scales.

Another stream of research uses optimization proxies to capture complex interactions, e.g., AC power flow equations, then embeds the trained proxy in a larger optimization problem, e.g., an unit-commitment problem [6], [7]. This strategy replaces the nonlinear component. e.g., the AC power flow equations, with a mixed-integer representation of a trained neural network. Although it removes the nonconvexity stemming from the physics, this approach requires the use of discrete variables, introducing another type of non-convexity, which reduces its tractability for large-scale systems.

To address this challenge, this paper explores the use of *input-convex neural networks (ICNN)* [8] as an alternative to non-convex Deep Neural Networks (DNNs) for applications where the neural network must be embedded in a larger optimization. More specifically, we highlight that DNNs are typically non-convex in the input parameters, which makes it difficult to embed them in an efficient way in an optimization model. Even state-of-the-art reformulation procedures of the DNN architecture using mathematical programming techniques (e.g., with a representation using binary variables) raise significant computational challenges. ICNNs, on the other hand, are convex functions of the input parameters by construction, thus suitable for embedding within optimization models, even for large-scale instances. *The open question, which is addressed in the paper, is whether these computational benefits come with a loss of accuracy or whether ICCNs preserve high prediction capabilities for a range of applications. This work is an attempt to answer this question when the learning task consists in building a tractable approximation of the value function of an OPF problem.* Such ICNNs, if accurate enough, would provide a highly valuable tool for a broad range of applications in power systems, including unit commitment and transmission switching (e.g., through Branch-and-Cut [8]), as well as a wealth of stochastic optimization and reinforcement learning [9] methods that implicitly rely on value functions and their gradients.

*The main objective of the paper is thus to determine whether ICNNs, despite their more limited expressive power, can match the performance of DNNs for approximating the value of OPF problems*. This is a pre-requisite to using ICNNs in larger optimizations and an open issue in the representation power of neural networks. The main contributions of the paper can be summarized as follows.

1) The paper derives strong theoretical guarantees on the performance of ICNNs. In particular, it provides bounds on the generalization error of ICNNs that only depend on the ICNN performance on the training data.
2) The paper explores specific ICNN architectures and trains it to learn three OPF formulations: the AC-OPF, the SOC relaxation, and the DC-OPF.
3) The paper reports the performance of the resulting ICNNs on large-scale systems, that are 50 times larger than prior research. The results demonstrate that ICCNs are capable of learning the value function of OPF problems, at least as effectively as DNNs on traditional test cases, with optimality gaps almost always lower than 0.5%.

The rest of the paper is organized as follows. Section II reviews related works in the literature, Section III describes the OPF formulations considered in the paper. Section IV presents the input-convex architecture used in the paper, and provides strong generalization bounds for this class of models. Section V presents and analyses the results of the proposed comparison, and Section VI concludes the paper.

## II. Related Work

The decomposition of intricate problems through value function approximations has found widespread application both in industry and academic literature. This approach has been instrumental in achieving tractable solutions for various practical scenarios, including multistage decision-making problems.

In multistage problems, such as those encountered in storage management and long-term asset investment, decision-makers often seek optimal policies. They do so by employing a spectrum of function approximations that range from simple parametric forms [10] to more intricate piece-wise models [11], [12]. Some of these advanced models may involve a substantial number of individual function evaluations to reach convergence [13]–[15].

DNNs have emerged as a standout player in approximating solutions, particularly in the domain of Optimal Power Flow (OPF) problems [16]–[18]. Furthermore, He et al. [19] have successfully employed neural networks to approximate the cost function of unit commitment problems, streamlining constraint screening processes and improving solution efficiency.

ICNNs have also found applications in energy-related challenges, such as unit commitment [7] and voltage regulation [20]. These networks offer a unique advantage by ensuring convexity within specific regions of their input domains through parameter constraints, thereby reducing the complexity of identifying convex mappings.

A notable contribution by Zhang et al. [21] involves training convex neural networks to predict the objective values of DC-OPF. This innovative approach enables the derivation of dual solutions, aiding in the identification of active sets of constraints. Leveraging the convexity of these ICNNs, this method augments the training process and provides valuable generalization bounds. Similar investigations by Chen et al. [22] extend this approach to systems with up to 118 buses, highlighting its applicability.

In another pioneering effort, Wu et al. [7] harness the power of ICNNs to map pre-fault operation conditions to transient stability indices. This enables the formulation of transient stability constraints, with numerical experiments conducted on systems featuring 39 and 118 buses validating the effectiveness of their methodology.

Machine Learning (ML) techniques, including ICNN-based approaches, have shown promise in discovering convex approximations and relaxations for optimization problems. In the context of Optimal Power Flow (OPF) applications [23], methodologies utilizing ICNNs have exhibited significant potential [24]. These collective advances underscore the growing role of advanced neural network techniques in enhancing optimization methodologies, promising more efficient and effective problem-solving strategies.

*This paper extends these lines of research in several directions. First, it demonstrates, for the first time, that ICNNs can provide state-of-the-art results in predicting value functions for large-scale OPF problems involving thousands of buses. The OPF problems studied in the paper also go beyond the DC model and include the SOCP relaxation and the AC-OPF. Second, the paper contributes strong generalization bounds for ICNNs that significantly expand existing work.*

## III. Optimal Power Flow

The Optimal Power Flow (OPF) problem [25] is a fundamental problem in power systems operations. The OPF problem finds the most economical generation dispatch so as to serve electricity demand while satisfying physical and engineering constraints. The paper considers the AC-OPF

**Model 1** The AC-OPF Model

$$
\begin{aligned}
\min \quad & \sum_{i \in \mathcal{N}} c_i \mathbf{p}^{\mathrm{g}}_i && && \text{(1a)}\\
\text{s.t.} \quad & \mathbf{S}^{\mathrm{g}}_i - \mathrm{S}^{\mathrm{d}}_i - (Y^s_i)^\star |\mathbf{V}_i|^2 = \sum_{ij \in \mathcal{E} \cup \mathcal{E}^R} \mathbf{S}^{\mathrm{f}}_{ij} && \forall i \in \mathcal{N} && \text{(1b)}\\
& \mathbf{S}^{\mathrm{f}}_{ij} = (Y_{ij} + Y^c_{ij})^\star |\mathbf{V}_i|^2 - Y^\star_{ij} \mathbf{V}_i \mathbf{V}^\star_j && \forall ij \in \mathcal{E} && \text{(1c)}\\
& \mathbf{S}^{\mathrm{f}}_{ji} = (Y_{ij} + Y^c_{ji})^\star |\mathbf{V}_j|^2 - Y^\star_{ij} \mathbf{V}^\star_i \mathbf{V}_j && \forall ij \in \mathcal{E} && \text{(1d)}\\
& |\mathbf{S}^{\mathrm{f}}_{ij}|, |\mathbf{S}^{\mathrm{f}}_{ji}| \leq \bar{s}_{ij} && \forall ij \in \mathcal{E} && \text{(1e)}\\
& \underline{\mathrm{v}}_i \leq |\mathbf{V}_i| \leq \bar{\mathrm{v}}_i && \forall i \in \mathcal{N} && \text{(1f)}\\
& \underline{\mathrm{p}}^{\mathrm{g}}_i \leq \mathbf{p}^{\mathrm{g}}_i \leq \bar{\mathrm{p}}^{\mathrm{g}}_i && \forall i \in \mathcal{N} && \text{(1g)}\\
& \underline{\mathrm{q}}^{\mathrm{g}}_i \leq \mathbf{q}^{\mathrm{g}}_i \leq \bar{\mathrm{q}}^{\mathrm{g}}_i && \forall i \in \mathcal{N} && \text{(1h)}
\end{aligned}
$$

**Model 2** The SOC-OPF model

$$
\begin{aligned}
\min \quad & \sum_{i \in \mathcal{N}} c_i \mathbf{p}^{\mathrm{g}}_i && && \text{(2a)}\\
\text{s.t.} \quad & \mathbf{p}^{\mathrm{g}}_i - \mathrm{p}^{\mathrm{d}}_i - g^s_i \mathbf{w}_i = \sum_{ij \in \mathcal{E} \cup \mathcal{E}^R} \mathbf{p}^{\mathrm{f}}_{ij} && \forall i \in \mathcal{N} && \text{(2b)}\\
& \mathbf{q}^{\mathrm{g}}_i - \mathrm{q}^{\mathrm{d}}_i + b^s_i \mathbf{w}_i = \sum_{ij \in \mathcal{E} \cup \mathcal{E}^R} \mathbf{q}^{\mathrm{f}}_{ij} && \forall i \in \mathcal{N} && \text{(2c)}\\
& \mathbf{p}^{\mathrm{f}}_{ij} = \gamma^p_{ij} \mathbf{w}_i + \gamma^{p,r}_{ij} \mathbf{w}^{\mathrm{r}}_{ij} + \gamma^{p,i}_{ij} \mathbf{w}^{\mathrm{i}}_{ij} && \forall ij \in \mathcal{E} && \text{(2d)}\\
& \mathbf{q}^{\mathrm{f}}_{ij} = \gamma^q_{ij} \mathbf{w}_j + \gamma^{q,r}_{ij} \mathbf{w}^{\mathrm{r}}_{ij} + \gamma^{q,i}_{ij} \mathbf{w}^{\mathrm{i}}_{ij} && \forall ij \in \mathcal{E} && \text{(2e)}\\
& \mathbf{p}^{\mathrm{f}}_{ji} = \gamma^p_{ji} \mathbf{w}_i + \gamma^{p,r}_{ji} \mathbf{w}^{\mathrm{r}}_{ij} + \gamma^{p,i}_{ji} \mathbf{w}^{\mathrm{i}}_{ij} && \forall ij \in \mathcal{E} && \text{(2f)}\\
& \mathbf{q}^{\mathrm{f}}_{ji} = \gamma^q_{ji} \mathbf{w}_j + \gamma^{q,r}_{ji} \mathbf{w}^{\mathrm{r}}_{ij} + \gamma^{q,i}_{ji} \mathbf{w}^{\mathrm{i}}_{ij} && \forall ij \in \mathcal{E} && \text{(2g)}\\
& (\mathbf{p}^{\mathrm{f}}_{ij})^2 + (\mathbf{q}^{\mathrm{f}}_{ij})^2 \leq \bar{s}^2_{ij} && \forall ij \in \mathcal{E} \cup \mathcal{E}^R && \text{(2h)}\\
& (\mathbf{w}^{\mathrm{r}}_{ij})^2 + (\mathbf{w}^{\mathrm{i}}_{ij})^2 \leq \mathbf{w}_i \mathbf{w}_j && \forall ij \in \mathcal{E} && \text{(2i)}\\
& \underline{\mathrm{v}}^2_i \leq \mathbf{w}_i \leq \bar{\mathrm{v}}^2_i && \forall i \in \mathcal{N} && \text{(2j)}\\
& \underline{\mathrm{p}}^{\mathrm{g}}_i \leq \mathbf{p}^{\mathrm{g}}_i \leq \bar{\mathrm{p}}^{\mathrm{g}}_i && \forall i \in \mathcal{N} && \text{(2k)}\\
& \underline{\mathrm{q}}^{\mathrm{g}}_i \leq \mathbf{q}^{\mathrm{g}}_i \leq \bar{\mathrm{q}}^{\mathrm{g}}_i && \forall i \in \mathcal{N} && \text{(2l)}
\end{aligned}
$$

formulation, its second-order cone (SOC) relaxation, and its DC-OPF linear approximation, which are presented next. For ease of reading, the presentation omits transformer tap ratios, phase angle difference constraints, and reference voltage (slack bus) constraints. All are implemented and considered in the experiments of Section V

## A. The AC-OPF Formulation

Model 1 presents the AC-OPF formulation, in complex variables. The objective (1a) minimizes total generation costs. Constraints (1b) enforce power balance (Kirchhoff's current law) at each bus. Constraints (1c) and (1d) express Ohm's law on forward and reverse power flows, respectively. Constraints (1e) enforce thermal limits on forward and reverse power flows. Finally, constraints (1f)–(1h) enforce minimum and maximum limits on nodal voltage magnitude, active generation, and reactive generation, respectively. The AC-OPF problem is nonlinear and non-convex and is typically solved using interior-point algorithms.

**Model 3** The DC-OPF Model

$$
\begin{aligned}
\min \quad & \sum_{i \in \mathcal{N}} c_i \mathbf{p}^{\mathrm{g}}_i && && \text{(8a)}\\
\text{s.t.} \quad & \mathbf{p}^{\mathrm{g}}_i + \sum_{ji \in \mathcal{E}} \mathbf{p}^{\mathrm{f}}_{ji} - \sum_{ij \in \mathcal{E}} \mathbf{p}^{\mathrm{f}}_{ij} = \mathrm{p}^{\mathrm{d}}_i && \forall i \in \mathcal{N} && \text{(8b)}\\
& \mathbf{p}^{\mathrm{f}}_{ij} = b_{ij}(\theta_j - \theta_i) && \forall ij \in \mathcal{E} && \text{(8c)}\\
& |\mathbf{p}^{\mathrm{f}}_{ij}| \leq \bar{s}_{ij} && \forall ij \in \mathcal{E} && \text{(8d)}\\
& \underline{\mathrm{p}}^{\mathrm{g}}_i \leq \mathbf{p}^{\mathrm{g}}_i \leq \bar{\mathrm{p}}^{\mathrm{g}}_i && \forall i \in \mathcal{N} && \text{(8e)}
\end{aligned}
$$

## B. The SOC-OPF Formulation

The Second-Oder Cone (SOC) relaxation of AC-OPF proposed by Jabr [26] is obtained from AC-OPF by introducing the additional variables

$$
\begin{aligned}
\mathbf{w}_i &= \mathbf{v}^2_i, && \forall i \in \mathcal{N} && (3)\\
\mathbf{w}^{\mathrm{r}}_{ij} &= \mathbf{v}_i \mathbf{v}_j \cos(\theta_j - \theta_i), && \forall ij \in \mathcal{E} && (4)\\
\mathbf{w}^{\mathrm{i}}_{ij} &= \mathbf{v}_i \mathbf{v}_j \sin(\theta_j - \theta_i), && \forall ij \in \mathcal{E} && (5)
\end{aligned}
$$

and the non-convex constraint

$$
(\mathbf{w}^{\mathrm{r}}_{ij})^2 + (\mathbf{w}^{\mathrm{i}}_{ij})^2 = \mathbf{w}_i \mathbf{w}_j, \qquad \forall ij \in \mathcal{E}. \qquad (6)
$$

The SOC relaxation is then obtained by relaxing Eq. (6) into

$$
(\mathbf{w}^{\mathrm{r}}_{ij})^2 + (\mathbf{w}^{\mathrm{i}}_{ij})^2 \leq \mathbf{w}_i \mathbf{w}_j, \qquad \forall ij \in \mathcal{E}. \qquad (7)
$$

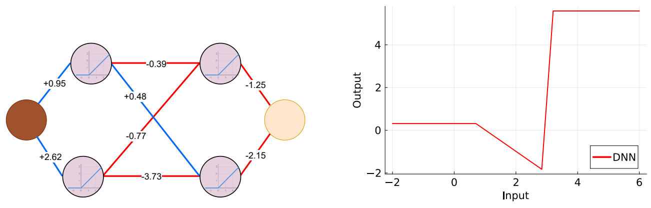
Model 2 presents the SOC-OPF formulation, in real variables. The objective function (2a) is equivalent to (1a). Constraints (2b) and (2c) enforce, at each node, Kirchhoff's current law for active and reactive power, respectively. Constraints (2d)–(2f) capture Ohm's law on active and reactive, forward and reverse power flows. The $\gamma$ parameters that appear in the constraints are constant terms derived from (1c)–(1d), after substituting variables $\mathbf{w}, \mathbf{w}^{\mathrm{r}}, \mathbf{w}^{\mathrm{i}}$. Constraints (2h) enforce thermal limits on forward and reverse power flows. Constraints (2i) is Jabr's inequality (7). Finally, constraints (2j)–(2l), like constraints (1f)–(1h), enforce minimum and maximum limits on nodal voltage magnitude, active and reactive generation.

The SOC-OPF formulation is nonlinear and convex. This makes it more tractable to solve than AC-OPF using, e.g., polynomial-time interior-point algorithms. Furthermore, since it is a relaxation of AC-OPF, solving SOC-OPF provides valid dual bounds on the optimal value of AC-OPF.

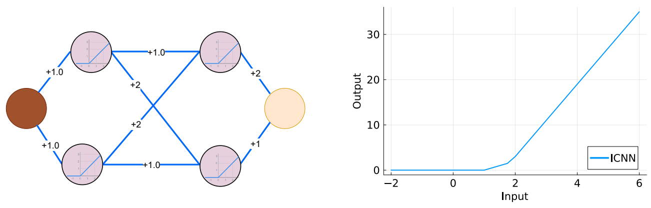
## C. The DC-OPF Formulation

The DC-OPF formulation is a linear approximation of AC-OPF. The approximation assumes that all voltage magnitudes are one per-unit, voltage angles are small and losses are negligible, and it ignores reactive power [27], [28]. The DC approximation underlies virtually all electricity markets and is widely used in, e.g., unit commitment and transmission network expansion planning problems.

Model 3 presents the resulting linear programming (LP) formulation of DC-OPF. The objective (8a) minimizes total generation costs. Constraints (8b) enforce (active) power balance at each node. Constraints (8c) approximate Ohm's



(a) Example DNN (left) and its Output (right). The DNN defines a non-convex function.



(b) Example ICNN (left) and its Output (right). All ICNN weights are positive and it defines a convex function.

Fig. 1. Illustration of Input-Convex Neural Networks.

law using a phase-angle formulation. Note that, DC-OPF does not consider reverse power flows, unlike AC-OPF and SOC-OPF; this is because losses are neglected in DC-OPF. Constraints (8d) enforce thermal constraints on each branch. Constraints (8e) enforce minimum and maximum limits on active power generation. Finally, recall that constraints on phase angle differences and slack bus are omitted from the presentation for readability, but are implemented in all the numerical experiments.

## IV. Methodology

The goal of the paper is to train an ML model that takes the nodal demand vector $\mathrm{S}^{\mathrm{d}}$ as input, and outputs the optimal solution of its corresponding OPF problem. This section presents the input-convex neural network (ICNN) architecture and its training, and establishes its generalization guarantees.

### A. The Input-Convex Neural Network Architecture

An ICNN is a special type of DNN that computes a convex function of its input [29]. ICNNs are well-suited when one seeks to represent or approximate convex functions since they are convex by design, unlike general DNNs. ICNNs achieve their convexity by combining convex activation functions with convexity-preserving operations [29].

The simplest ICNN architecture consists of fully-connected layers of the form $h(\mathbf{x}) = \mathrm{ReLU}\,(W\mathbf{x} + d)$, where $\mathbf{x} \in \mathbb{R}^n$ denotes the layer input vector, $d \in \mathbb{R}^m$ is the bias vector, and $W \in \mathbb{R}^{m \times n}$ is a weight matrix with non-negative coefficients, to ensure convexity. The Rectified Linear Unit ($\mathrm{ReLU}$) activation function $\mathrm{ReLU}(x) = \max(0, x)$ is applied element-wise. Figure 1 illustrates the difference between DNNs and ICNNs on a small example, and showcases ICNNs' convexity.

To learn the value function of OPF problems, this paper considers ICNN architectures with skip-connections, which allow the approximation of convex functions with decreasing slopes, and have been shown to improve performance [21], [29]. The overall architecture is illustrated in Figure 2. Its $k$-th layer is of the form

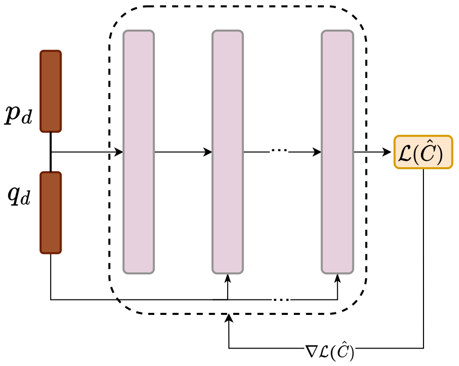


Fig. 2. Fully Connected ICNN

$$\mathbf{x}^k = h^k(\mathbf{x}^{k-1}) = \mathrm{ReLU}(W^k \mathbf{x}^{k-1} + H^k \mathbf{x}^0 + d^k), \tag{9}$$

where $\mathbf{x}^k$ and $\mathbf{x}^{k-1}$ denote the outputs of layer $k$ and $k-1$, $\mathbf{x}^0$ denotes the input of the ICNN, i.e., $\mathbf{x}^0 = (\mathrm{p}^{\mathrm{d}}, \mathrm{q}^{\mathrm{d}})$, $d^k$ is the bias vector, and $W^k, H^k$ are weight matrices. Skip-connections feed the ICNN input $\mathbf{x}^0$ to each layer. The coefficients of $W^k$ are non-negative, whereas $H^k$ may take positive or negative values without affecting convexity [29].

### B. The Training of ICNNs

The DNNs weights were optimized using the ADAM optimizer with a learning rate of 0.01 without incorporating any regularization techniques. ICNNs cannot directly be trained with traditional gradient descent algorithms, because some of their weights must be non-negative. So the training used projected gradient descent, which clips gradients before updating the weights (a common choice for bounding the weights). Each model underwent a maximum of 5000 epochs, with periodic validation checks at epochs 200, 600, 1500, and 3000 to ensure attainment of the predefined threshold (below a 0.3% gap) on the validation set. The best-performing model following these checks was retained. A batch size of 32 was utilized throughout training.

While it converges under the same assumptions as traditional gradient descent algorithms, projected gradient descent requires special attention to avoid slow convergence. Preliminary experiments showed that, for larger learning rates, gradient clipping results in a considerable slow-down, when compared to the training of regular DNNs. This stems from the greater impact on the loss that happens when projecting back onto the feasible region. In fact, appropriate learning rates (e.g., [30]) also depend on the relative distance to the constraint barriers - which can be very small when approaching the local optimum the algorithm is converging to. Unfortunately, dynamically adapting learning rates is not an easy task when post-clipping gradients from a standard gradient descent algorithms. Scheduling parameters had to be tuned in hyper-parameter optimization over the validation set. The best scheduling found was the following: every 100 epochs, when no progress is detected, the learning rate is reduced by 2%. Furthermore, if no progress is observed after 1000 epochs, the batch size was halved iteratively (starting at 32) until reaching a minimum size of 8.

### C. Generalization Guarantees

One fundamental benefit of using ICNNs is their stronger generalization guarantees. To the best of the authors' knowledge, the strongest theoretical guarantee for ICNNs in the context of OPF proxies is from [21] (Theorem 6.5): they show that the gradient of a perfectly-trained ICNN is bounded over the convex hull of the training set. While their proof uses convexity arguments, the result is not specific to ICNNs. For instance, it also holds for fully-connected DNNs with ReLU activation (DNN-ReLU for short). Indeed, DNN-ReLUs represent continuous, piece-wise linear functions, whose gradient is piece-wise constant. Since DNN-ReLUs have a finite number of pieces, their gradients are immediately bounded.

This paper improves on the generalization guarantees of [21]: it provides generalization bounds for *general ICNNs* (Theorem 1), and *explicit formulae* for *perfectly-trained ICNNs* (Theorem 2). The results are presented for a parametric convex optimization problem $\Phi(b)$ that the ICNN $f$ must learn, and its parametric strong dual $\Psi(b)$. For instance, for the case of a parametric linear program,

$$\Phi(b) = \min_x \left\{ c^\top x \;\middle|\; Ax \geq b \right\}, \tag{10}$$

$$\Psi(b) = \max_y \left\{ b^\top y \;\middle|\; A^\top y = c, y \geq 0 \right\}, \tag{11}$$

with primal and dual variables $x$ and $y$ respectively. When strong duality holds, $\Phi(b) = \Psi(b)$. $\Phi$ is convex in $b$ and for a given $b$, any dual optimal solution is a sub-gradient of $\Phi$ [31]. In addition, if the dual optimum $y^*(b)$ is unique, then $\nabla\Phi$ is defined and $\nabla\Phi(b) = y^*(b)$.

The results are expressed in terms of a dataset $\mathcal{D} = \{(b_i, y_i, z_i)\}_{i=1,\dots,N}$ where each $b_i, y_i, z_i$ are the right-hand side, optimal dual solution, and optimal value of instance $i = 1, \dots, N$, respectively. They are also expressed in terms of $\mathcal{B} = \text{conv}\{b_1, \dots, b_N\}$, the convex hull of the right-hand sides with diameter $\text{diam}(\mathcal{B})$. The first result is a generalization bound for arbitrary ICNNs.

**Theorem 1.** *Let $f$ be an ICNN, $\tilde{z}_i = f(b_i)$, and $\tilde{y}_i = \nabla f(b_i)$. There exists a constant $M$, whose value depends only on $\mathcal{D}$ and $\{\tilde{y}_i, \tilde{z}_i\}$, such that*

$$\forall b \in \mathcal{B}, |f(b) - \Phi(b)| \leq M. \tag{12}$$

*Proof.* Define $\check{f}$, $\hat{f}$ over $\mathcal{B}$ as

$$\check{f}(b) = \max \left\{ \tilde{z}_i + \tilde{y}_i^\top (b - b_i) \;\middle|\; i = 1, \dots, N \right\}$$

$$\hat{f}(b) = \min_\lambda \left\{ \sum_i \lambda_i \tilde{z}_i \;\middle|\; \lambda \geq 0, e^\top \lambda = 1 \right\}$$

Note that $\check{f}, \hat{f}$ are convex lower and upper envelopes of $f$. The constant $M$ is obtained by solving

$$\begin{aligned} M = \max_{b,z} \quad & |z - \Phi(b)| && (13a)\\ s.t. \quad & \check{f}(b) \leq z \leq \hat{f}(b), && (13b)\\ & b \in \mathcal{B}, && (13c) \end{aligned}$$

which concludes the proof by convexity of $\Phi$. □

Theorem 1 provides a worst-case guarantee on the generalization performance of the ICNN, which only depends on the value of $f$ on the dataset $\mathcal{D}$. The next Theorem 2 provides an explicit worst-case guarantee when the ICNN *perfectly fits* the training data.

**Theorem 2.** *Let $f$ be an ICNN for learning $\Phi$ and assume that, for all $i \in \{1, \dots, N\}$,*

$$f(b_i) = z_i = \Phi(b_i) \text{ and } \nabla_b f(b_i) = y_i = \Phi(b_i).$$

*Then, $\forall b \in \mathcal{B}, |f(b) - \Phi(b)| \leq \max_i \|y_i\| \times \text{diam}(\mathcal{B})$.*

*Proof.* Let $b \in \mathcal{B}$, i.e., $b = \sum_i \lambda_i b_i$ where weights $\lambda_i$ are non-negative and sum to 1. By convexity of $f$ and $\Phi$,

$$\forall i, \quad z_i + (b - b_i)^\top y_i \leq \Phi(b), f(b) \leq \sum_j \lambda_j z_j.$$

Combining the above inequalities with weights $\lambda_i$ yields

$$\begin{aligned} |\Phi(b) - f(b)| &\leq \sum_j \lambda_j z_j - \sum_i \lambda_i [z_i - (b - b_i)^\top y_i] \\ &\leq \sum_i \lambda_i (b - b_i)^\top y_i \\ &\leq \max_i (b - b_i)^\top y_i \\ &\leq \max_i \|y_i\| \,\text{diam}(\mathcal{B}) \end{aligned}$$

which concludes the proof. □

Observe that Theorem 2 applies to any subset of data points, which offers tighter guarantees over smaller domains.

## V. Numerical Experiments

The section reports numerical results for ICNNs that are trained to learn the value functions of DC-OPF, SOC-OPF, and AC-OPF. While DC-OPF and SOC-OPF are convex with convex value functions, AC-OPF is not convex and its value function is not guaranteed to be convex. In this last case, the task is thus to approximate (possibly) the OPF value function with a convex function.

### A. Experimental Setup

The proposed approach is evaluated on four test cases from PGLib [32] with up to 6468 buses. These test cases are 50 times larger in size compared to those in [21]. Moreover, these prior results only considered DC-OPF.

Table I reports, for each system, the number of buses $|\mathcal{N}|$, branches $|\mathcal{E}|$ and generators $|\mathcal{G}|$, as well as the nominal total demand ($\text{P}^d_{\text{ref}}$) and its range across the dataset ($[\underline{\text{P}}^d, \bar{\text{P}}^d]$).

TABLE I
STATISTICS OF THE PGLIB TEST CASES.

| System | $\lvert\mathcal{N}\rvert$ | $\lvert\mathcal{E}\rvert$ | $\lvert\mathcal{G}\rvert$ | $\mathrm{P}^d_{\mathrm{ref}}$ | $[\underline{\mathrm{P}}^d, \bar{\mathrm{P}}^d]$ |
|---|---|---|---|---|---|
| `ieee300` | 300 | 411 | 69 | 263 | [ 210, 280] |
| `pegase1k` | 1354 | 1991 | 260 | 781 | [ 625, 820] |
| `pegase2k` | 2869 | 4582 | 510 | 1522 | [1218, 1599] |
| `rte6k` | 6468 | 9000 | 399 | 1109 | [ 887, 1164] |

For each system, a dataset of $50,000$ OPF instances is generated by perturbing the load vectors as follows

$$\mathrm{p}^{\mathrm{d}} = \alpha \times \eta \times \mathrm{p}^{\mathrm{d}}_{\mathrm{ref}}, \quad \mathrm{q}^{\mathrm{d}} = \alpha \times \eta \times \mathrm{q}^{\mathrm{d}}_{\mathrm{ref}},$$

where $\alpha$ is the system-wide scaling factor, sampled from a uniform distribution, $\eta \in \mathbb{R}^{|\mathcal{N}|}$ is the bus-level uncorrelated noise vector, sampled from a log-normal distribution with mean $1$ and standard deviation $5\%$, and $\mathrm{p}^{\mathrm{d}}_{\mathrm{ref}}, \mathrm{q}^{\mathrm{d}}_{\mathrm{ref}}$ are the nominal active and reactive load vectors. The range of $\alpha$ is selected to avoid regions where AC-OPF becomes infeasible.

The OPF problems are formulated with *PowerModels.jl* and solved with Mosek (DC and SOC) or Ipopt (AC). Infeasible instances are excluded, and the remaining dataset is partitioned into training (40%), validation (30%), and testing (30%) sets for appropriate training and model assessment procedures. For each system and OPF formulation, this paper trains a DNN and an ICNN to predict the value function of the considered OPF. Both models use the very same architecture, the only difference being that DNN weights are unrestricted. All ML models are implemented in Julia using *Flux.jl* [33]. Experiments are carried out on Intel(R) Xeon(R) Gold 6226 CPU @ 2.70GHz machines with NVIDIA Tesla A100 GPUs on the Phoenix cluster [34].

## B. ICNN Performance

The paper uses relative absolute optimality gap to measure the performance of the DNN and ICNN models. Let $z^*$ and $\tilde{z}$ denote the ground truth (obtained by the optimization solver) and the predicted optimal value by a ML model, respectively. The relative absolute optimality gap, henceforth referred to as *optimality gap* for simplicity, is defined as

$$\text{gap} = \frac{|\tilde{z} - z^*|}{|z^*|}. \tag{14}$$

Note that the prediction $\tilde{z}$ may *over-estimate or under-estimate* the ground truth $z^*$, which is why the absolute value is needed. Unless specified otherwise, reported averages use the geometric mean $\mu(x_1, ..., x_n) = \sqrt[n]{x_1 \times ... \times x_n}$.

Table II reports, for each system and model architecture (DNN or ICNN), the geometric mean optimality gap, and the worst-case optimality gap across the testing set. First, observe that, except for SOC on the `pegase2k` system (for which ICNN training did not converge), *ICNN always yields a mean optimality gap below 0.5%*, and DNN yields mean gaps below 1%. Overall, predictions for DC-OPF tend to be slightly more accurate than those for SOC- and AC-OPF, which may be explained by the fact that DC-OPF is a

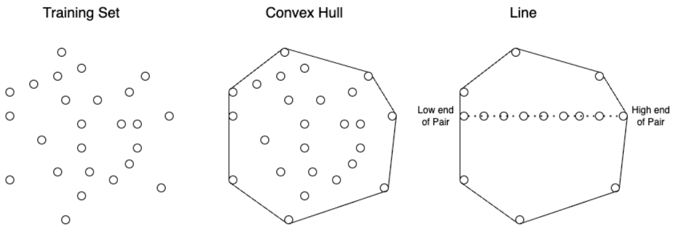


Fig. 3. Illustrating how instances are selected to analyze the behavior of AC-OPF operational cost. The idea is to select two extreme points of the convex hull of load instances and generating instances between them.

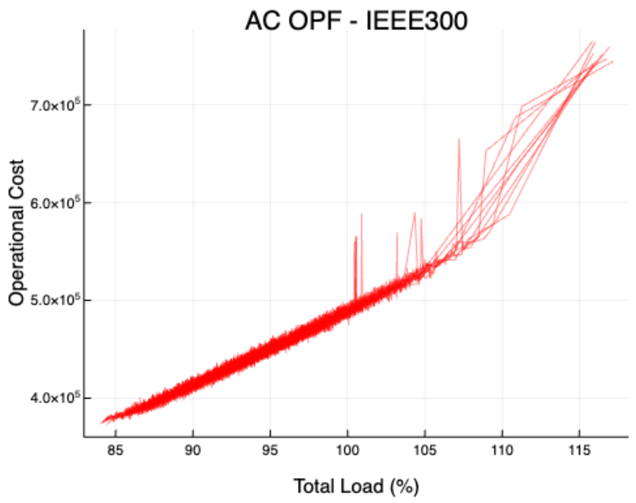


Fig. 4. AC-OPF Operational Cost for 10 sets of loads for `ieee300` obtained by the procedure sketched in Figure 3.

linear programming problem, whereas SOC- and AC-OPF are nonlinear. Second, *ICNN almost always outperforms DNN in terms of mean optimality gap.* This holds even for AC-OPF, whose value function is non-convex. *These results suggest that ICNN models are accurate enough to be used instead of general, non-convex DNN models to accurately represent the value functions of OPF problems.*

The ICNN effectiveness in approximating AC-OPF seems to stem from the nature of AC-OPF value functions that are approximately convex. Indeed, when analyzing a few random sets of instances of AC-OPF problems, each set consisting of instances between a pair of extreme points in the convex hull of load instances, it appears that the value function is mostly a convex function of load change. This is illustrated in Figure 4 where each line is a set of instances between two points in the convex hull (a visual explanation is given by Figure 3). This is consistent with the low optimality gaps between conic relaxations reported in the literature [35], and in Table II.

It is also important to study the worst-case optimality gaps, especially if ML models are to be embedded in larger optimization problems. Interestingly, the worst-case gaps are highest for the `ieee300` system, with both ICNN and DNN exhibiting worst-case optimality gaps of about 15%, i.e., roughly 40 times larger than their mean gap. This demon-

TABLE II
ICNN Performance Results.

| System | OPF | Mean gap (%) ICNN | Mean gap (%) DNN | Mean gap (%) DC | Mean gap (%) SOC | Worst gap (%) ICNN | Worst gap (%) DNN | Worst gap (%) DC | Worst gap (%) SOC |
|---|---|---|---|---|---|---|---|---|---|
| `ieee300` | DC | 0.15 | 0.19 | 0.00 | - | 1.58 | 1.90 | 0.00 | - |
| | SOC | 0.31 | 0.37 | 5.43 | 0.00 | 5.77 | 4.96 | 10.79 | 0.00 |
| | AC | 0.39 | 0.39 | 7.35 | 1.98 | 15.81 | 14.56 | 21.95 | 16.47 |
| `pegase1k` | DC | 0.28 | 0.33 | 0.00 | - | 2.35 | 1.83 | 0.00 | - |
| | SOC | 0.33 | 0.82 | 1.60 | 0.00 | 2.22 | 2.15 | 1.88 | 0.00 |
| | AC | 0.33 | 0.68 | 2.91 | 1.32 | 2.37 | 1.95 | 3.49 | 1.89 |
| `pegase2k` | DC | 0.22 | 0.30 | 0.00 | - | 3.45 | 3.21 | 0.00 | - |
| | SOC | 1.03 | 0.32 | 2.15 | 0.00 | 3.15 | 2.36 | 2.35 | 0.00 |
| | AC | 0.24 | 0.27 | 3.02 | 0.80 | 8.59 | 8.89 | 8.88 | 9.13 |
| `rte6k` | DC | 0.27 | 0.38 | 0.00 | - | 1.76 | 1.15 | 0.00 | - |
| | SOC | 0.29 | 0.57 | 2.67 | 0.00 | 1.69 | 5.52 | 3.17 | 0.00 |
| | AC | 0.25 | 0.33 | 3.05 | 0.33 | 2.71 | 3.08 | 3.67 | 0.36 |

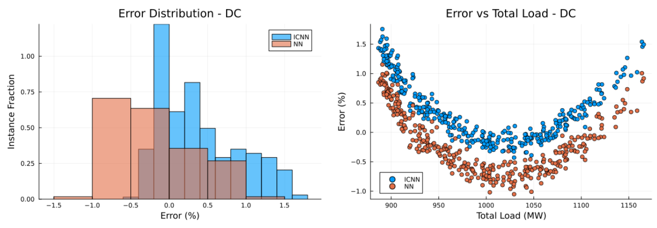
strates that low mean optimality gaps are not a guarantee of uniformly good performance. For the larger systems, both architectures exhibit lower worst-case gaps, ranging around 2–5%, or roughly 20 times larger than the mean optimality gap. Interestingly, ICNN tends to produce better worst-case optimality gaps than DNN for SOC and AC-OPF on the two larger test cases. Designing training procedures that reduce worst-case performance is an active area of research.

Figure 5 provides a more granular picture of the DNN and ICNN performance on the `rte6k` system. The figure reports, for each OPF formulation, the distribution of relative optimality gaps over the dataset (left panel) and as a function of total load (right panel). Note that these results are *relative optimality gaps* $(\tilde{z} - z^*)/|z^*|$: a positive (resp. negative) gap indicates that the ML model over-estimates (resp. under-estimates) the ground truth value. Overall, echoing the results of Table II, *the ICNN gaps exhibit lower variance than the DNN gaps, especially for SOC-OPF.* Furthermore, both formulations exhibit similar behavior with respect to the system total load, with overall higher gaps towards the ends of the range. The latter is expected as there are fewer training samples in those regions. This observation aligns with findings in [36] advocating for training on a broader input domain, thus underscoring the importance of comprehensive data coverage. For SOC-OPF and AC-OPF, the plots on the right panels highlight the benefits of ICNNs.

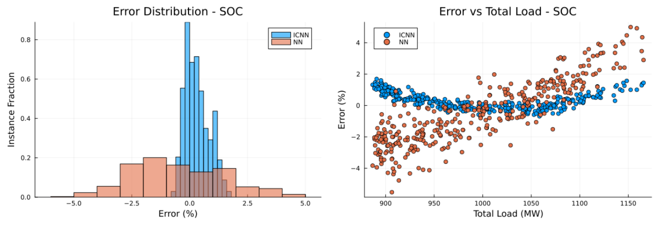
While ICNNs enjoy strong theoretical guarantees, it is important to shed more light on the generalization performance of ICNNs and DNNs. Additional experiments outlined in Table III demonstrate that DNNs and ICNNs exhibit very similar performance across instances both inside and outside the convex hull of the training set, even when the training set size is significantly reduced, suggesting that DNNs do not overfit in this setting.

## VI. Conclusion

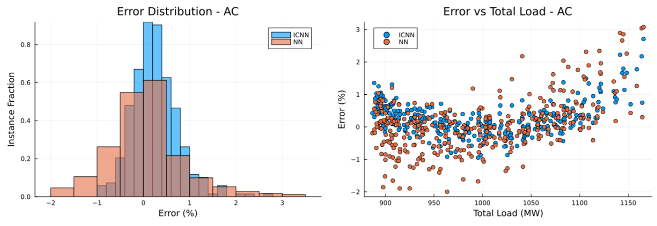
This paper studies whether ICNNs can be accurate enough to approximate the value function of large-scale OPF problems. ICNNs are an attractive avenue for approximating OPF problems, since their convexity enables them to be embedded in optimization models without resorting to a MIP reformulation. The results in this paper indicate that ICNNs can consistently approximate system operating costs under different grid conditions, often with optimization gaps below 0.5. This suggests that convexity constraints do not entail a substantial sacrifice in accuracy, even in inherently non-convex setups like AC-OPF. Notably, in most cases, ICNNs exhibit slightly lower geometric errors than DNNs with similar architectures,



(a) Comparison of errors on DC-OPF



(b) Comparison of errors on SOC-OPF



(c) Comparison of errors on AC-OPF

Fig. 5. Comparison of relative gap $(\tilde{z} - z^*)/|z^*|$ on `rte6k`. Positive (resp. negative) values mean the prediction over- (resp. under-) estimates the ground truth optimal value.

TABLE III
ICNN PERFORMANCE RESULTS FOR PREDICTING AC-OPF OPERATIONAL COSTS IN `ieee300`
(INSIDE CONVEX-HULL — OUTSIDE CONVEX-HULL)

| | | ICNN | | DNN | |
|---|---|---|---|---|---|
| Training Set size | Convex Hull Coverage of Test Set | Mean Gap | Worst Gap | Mean Gap | Worst Gap |
| 400 | 22% | (6.75 — 6.68) | (26.78 — 27.53) | (6.79 — 6.72) | (26.95 — 27.66) |
| 6500 | 22% | (1.47 — 1.46) | (18.73 — 19.85) | (1.76 — 1.76) | (18.22 — 17.90) |
| 20000 | 43% | (0.33 — 0.43) | (10.40 — 19.87) | (0.35 — 0.42) | (10.23 — 17.72) |

reinforcing the potential of this approach.

It is important however to acknowledge challenges and opportunities in ICNN research. As highlighted by the SOC-OPF formulation of the `pegase2k`system, ICNN training did not converge smoothly in some instances, indicating that fine tuning the optimization parameters may be needed for more efficient training. Additionally, convexity introduces a slight optimistic bias increasing the error in extreme cases. Although it may not result in large impacts, and might even be desirable depending on the application, it is important to mitigate its potential negative effects. Penalizing differently for under and over estimations might allow the training to be tailored for different applications. Moreover, a well-trained ICNN can offer, at its utmost capability, the optimal piece-wise convex approximation of a function. This resembles the objective pursued by numerous operators that fine-tune the DC approximation to approximate AC, which has proven beneficial in many cases. Nonetheless, in specific instances where the target function exhibits pronounced nonlinearities, ICNN performance may falter.

In summary, this paper identified a promising pathway for addressing complex and time-sensitive OPF couplings in power systems. The ability to achieve near-optimal cost estimations with reduced computational burden through machine learning methods has significant implications for the efficient operation and management of electrical grids. As research on ICNNs for OPF progresses, it is likely that they will play an increasingly vital role in ensuring the reliability and sustainability of power systems in the future.

## REFERENCES

[1] J. Carpentier, “Optimal power flows,” *International Journal of Electrical Power & Energy Systems*, vol. 1, no. 1, pp. 3–15, 1979.
[2] S. Granville, J. Mello, and A. Melo, “Application of interior point methods to power flow unsolvability,” *IEEE Transactions on Power Systems*, vol. 11, no. 2, pp. 1096–1103, 1996.
[3] J. Kotary, F. Fioretto, P. Van Hentenryck, and B. Wilder, “End-to-end constrained optimization learning: A survey,” *arXiv preprint arXiv:2103.16378*, 2021.
[4] P. Van Hentenryck, “Machine learning for optimal power flows,” *Tutorials in Operations Research: Emerging Optimization Methods and Modeling Techniques with Applications*, pp. 62–82, 2021.
[5] A. Velloso and P. Van Hentenryck, “Combining deep learning and optimization for preventive security-constrained dc optimal power flow,” *IEEE Transactions on Power Systems*, vol. 36, no. 4, pp. 3618–3628, 2021.
[6] A. Kody, S. Chevalier, S. Chatzivasileiadis, and D. Molzahn, “Modeling the AC power flow equations with optimally compact neural networks: Application to unit commitment,” *Electric Power Systems Research*, vol. 213, p. 108282, 2022.
[7] T. Wu and J. Wang, “Transient stability-constrained unit commitment using input convex neural network,” *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–13, 2023.
[8] D. R. Morrison, S. H. Jacobson, J. J. Sauppe, and E. C. Sewell, “Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning,” *Discrete Optimization*, vol. 19, pp. 79–102, 2016.
[9] A. W. Rosemberg, A. Street, J. D. Garcia, D. M. Valladão, T. Silva, and O. Dowson, “Assessing the cost of network simplifications in long-term hydrothermal dispatch planning models,” *IEEE Transactions on Sustainable Energy*, vol. 13, no. 1, pp. 196–206, 2021.
[10] W. B. Powell and S. Ghadimi, “The parametric cost function approximation: A new approach for multistage stochastic programming,” *arXiv preprint arXiv:2201.00258*, 2022.
[11] M. V. Pereira and L. M. Pinto, “Multi-stage stochastic optimization applied to energy planning,” *Mathematical programming*, vol. 52, no. 1-3, pp. 359–375, 1991.
[12] A. Shapiro, D. Dentcheva, and A. Ruszczyński, *Lectures on stochastic programming: modeling and theory*. SIAM, 2009.
[13] A. Ruszczynski, *Nonlinear optimization*. Princeton university press, 2011.
[14] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Springer Science & Business Media, 2003, vol. 87.
[15] T. Asamov and W. B. Powell, “Regularized decomposition of high-dimensional multistage stochastic programs with markov uncertainty,” *SIAM Journal on Optimization*, vol. 28, no. 1, pp. 575–595, 2018.
[16] W. Chen, S. Park, M. Tanneau, and P. Van Hentenryck, “Learning optimization proxies for large-scale security-constrained economic dispatch,” *Electric Power Systems Research*, vol. 213, p. 108566, 2022.
[17] F. Fioretto, T. W. Mak, and P. Van Hentenryck, “Predicting ac optimal power flows: Combining deep learning and lagrangian dual methods,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 01, 2020, pp. 630–637.
[18] T. W. Mak, F. Fioretto, and P. VanHentenryck, “Load embeddings for scalable ac-opf learning,” *arXiv preprint arXiv:2101.03973*, 2021.
[19] X. He, H. Wen, Y. Zhang, and Y. Chen, “Enabling fast unit commitment constraint screening via learning cost model,” 2022.
[20] Y. Chen, Y. Shi, and B. Zhang, “Data-driven optimal voltage regulation using input convex neural networks,” *Electric Power Systems Research*, vol. 189, p. 106741, 2020.
[21] L. Zhang, Y. Chen, and B. Zhang, “A convex neural network solver for dcopf with generalization guarantees,” *IEEE Transactions on Control of Network Systems*, vol. 9, no. 2, pp. 719–730, 2022.
[22] Y. Chen, L. Zhang, and B. Zhang, “Learning to solve dcopf: A duality approach,” *Electric Power Systems Research*, vol. 213, p. 108595, 2022.
[23] F. Cengil, H. Nagarajan, R. Bent, S. Eksioglu, and B. Eksioglu, “Learning to accelerate globally optimal solutions to the ac optimal power flow problem,” *Electric Power Systems Research*, vol. 212, p. 108275, 2022.
[24] L. Duchesne, Q. Louveaux, and L. Wehenkel, “Supervised learning of convex piecewise linear approximations of optimization problems,” in *29th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, 2021.
[25] J. Carpentier, “Contribution to the economic dispatch problem,” *Bulletin de la Societe Francoise des Electriciens*, vol. 3, no. 8, pp. 431–447, 1962.
[26] R. A. Jabr, “Radial distribution load flow using conic programming,” *IEEE Transactions on Power Systems*, vol. 21, no. 3, pp. 1458–1459, Aug 2006.
[27] B. Stott, J. Jardim, and O. Alsac, “DC power flow revisited,” *IEEE Transactions on Power Systems*, vol. 24, no. 3, pp. 1290–1300, Aug 2009.

[28] J. A. Taylor, *Convex optimization of power systems*. Cambridge University Press, 2015.

[29] B. Amos, L. Xu, and J. Z. Kolter, “Input convex neural networks,” in *International Conference on Machine Learning*. PMLR, 2017, pp. 146–155.

[30] N. Tripuraneni, N. Flammarion, F. Bach, and M. I. Jordan, “Averaging stochastic gradient descent on riemannian manifolds,” in *Conference on Learning Theory*. PMLR, 2018, pp. 650–687.

[31] J. Nocedal and S. J. Wright, *Numerical optimization*. Springer, 1999.

[32] S. Babaeinejadsarookolaee, A. Birchfield, R. D. Christie, C. Coffrin, C. DeMarco, R. Diao, M. Ferris, S. Fliscounakis, S. Greene, R. Huang *et al.*, “The power grid library for benchmarking ac optimal power flow algorithms,” *arXiv preprint arXiv:1908.02788*, 2019.

[33] M. Innes, E. Saba, K. Fischer, D. Gandhi, M. C. Rudilosso, N. M. Joy, T. Karmali, A. Pal, and V. Shah, “Fashionable modelling with flux,” *CoRR*, vol. abs/1811.01457, 2018. [Online]. Available: https://arxiv.org/abs/1811.01457

[34] PACE, *Partnership for an Advanced Computing Environment (PACE)*, 2017. [Online]. Available: http://www.pace.gatech.edu

[35] S. Gopinath and H. L. Hijazi, “Benchmarking large-scale acopf solutions and optimality bounds,” in *2022 IEEE Power & Energy Society General Meeting (PESGM)*. IEEE, 2022, pp. 1–5.

[36] A. Venzke, G. Qu, S. Low, and S. Chatzivasileiadis, “Learning optimal power flow: Worst-case guarantees for neural networks,” in *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, 2020, pp. 1–7.