

---

# Efficient Bisection Projection to Ensure Neural-Network Solution Feasibility for Optimization over General Set

---

Enming Liang<sup>1</sup> Minghua Chen<sup>1,2</sup>

## Abstract

Neural networks (NNs) have emerged as promising tools for solving constrained optimization problems in real-time. However, ensuring constraint satisfaction for NN-generated solutions remains challenging due to prediction errors. Existing methods to ensure NN feasibility either suffer from high computational complexity or are limited to specific constraint types. We present Bisection Projection, an efficient approach to ensure NN solution feasibility for optimization over general compact sets with non-empty interiors. Our method comprises two key components: (i) a dedicated NN (called IPNN) that predicts interior points (IPs) with low eccentricity, which naturally accounts for approximation errors; (ii) a bisection algorithm that leverages these IPs to recover solution feasibility when initial NN solutions violate constraints. We establish theoretical guarantees by providing sufficient conditions for IPNN feasibility and proving bounded optimality loss of the bisection operation under IP predictions. Extensive evaluations on real-world non-convex problems demonstrate that Bisection Projection achieves superior feasibility and computational efficiency compared to existing methods, while maintaining comparable optimality gaps.

## 1. Introduction

Constrained Optimization (CO) plays an essential role in various engineering fields, such as supply chain, transportation, and power systems. To solve CO problems, iterative algorithms, such as interior point methods, have been developed and embedded within commercial solvers like Gurobi. These tools are designed to tackle CO with high precision, providing high-quality solutions. However, they can be slow for real-time applications with tight time constraints.

---

<sup>1</sup>Department of Data Science, City University of Hong Kong, Hong Kong <sup>2</sup>School of Data Science, The Chinese University of Hong Kong (Shenzhen), China. Correspondence to: Minghua Chen <minghua.chen@cityu.edu.hk, minghua@cuhk.edu.cn>.

Recent advancements in machine learning (ML) have introduced innovative strategies for solving CO problems in real-time, including the end-to-end (E2E) solution mapping (Amo22), the learning-to-optimize (L2O) iterative scheme (CCC<sup>+</sup>21), the generative modeling framework (LC23), and hybrid approaches (KFVHW21). One powerful idea is to leverage the universal approximation ability of neural networks (NNs) (HSW89; LLPS93) to predict high-quality solutions given input parameters, significantly reducing computation time compared to iterative solvers. For instance, NNs have been trained to solve non-convex optimal power flow problems in grid operations, achieving 2-4 orders of magnitude speedup over iterative solvers (PZC19; GWW19; PZCZ20; FMVH20; ZB20; DRK20).

Despite the minor optimality loss and significant speedup of NN-based methods, guaranteeing the feasibility of NN solutions with respect to problem constraints, which is important to safe-critical scenarios, remains a challenge due to inherent NN prediction errors. Existing methods to ensure NN feasibility either suffer from high computational complexity or are limited to specific constraints. See Sec. 2 for discussions on related works.

In this paper, we develop **Bisection Projection** (BP) as a simple yet efficient scheme to recover NN solutions feasibility over a (fairly) general compact set with a non-empty interior, beyond previous works on linear (WZG<sup>+</sup>23; ZPCL23), convex (THH23), or ball-homeomorphic constraints (LCL23; LCL24). Our contributions are as follows:

▷ In Sec. 4, we introduce the BP framework for ensuring NN solution feasibility. It comprises two steps: (i) a dedicated NN (called IPNN) that predicts interior points (IPs) of low *eccentricity*; (ii) an efficient *bisection* algorithm that leverages these IPs to recover solution feasibility.

▷ In Sec. 5, we establish theoretical guarantees by providing sufficient conditions for IPNN feasibility and proving bounded *eccentricity*-related optimality loss of the bisection operation under IP predictions.

▷ In Sec. 6, we carry out extensive experiments over convex and non-convex problems to evaluate the performance of BP. The results show that it outperforms existing methods in feasibility and run-time complexity while achieving similar optimality losses.

Table 1. Existing work for ensuring NN solution feasibility for continuous constrained optimization problems.

Existing Work (refer to Sec. 2 for details)	Constraint Setting			Performance Guarantee		
	Input dependent	Non-linear equality	Non-convex inequality	Feasibility ensuring	Optimality bound	Low run-time
Penalty training	✓	✓	✓	✗	✗	✓
Orthogonal Proj.	✓	✓	✓	✓	✓	✗
Preventive learning	✓	✗ (linear)	✗ (linear)	✓	✗	✓
Gauge mapping	✓	✗ (linear)	✗ (linear)	✓	✗	✓
DC3	✓	✓	✓	✗	✗	✓
Gauge Proj.	✗	✗ (linear)	✗ (convex)	✓	✓	✓
Homeomorphic Proj.	✓	✓	✓ (BH <sup>1</sup> )	✓	✓	✓
<b>Bisection Proj.</b>	✓	✓	✓	✓	✓	✓

<sup>1</sup> BH indicates the constraint set is homeomorphic to a ball, which includes all compact convex sets and a part of non-convex sets.

## 2. Related Work

Machine Learning (ML)-driven optimization has emerged as an active research field (PZC19; KJVHW21; CCC<sup>+</sup>21; Amo22). A fundamental challenge is ensuring neural network (NN) solution feasibility under input-dependent constraints. For basic constraints like simplexes or boxes, feasibility can be enforced through activation functions (e.g., Softmax or Sigmoid). For complex constraints, various approaches have been developed, as summarized in Table 1.

**Equality constraints.** Linear equations and certain non-linear equations with constant ranks, can be embedded as neural network layers by predicting a subset of variables and solving for the remaining variables to satisfy the equality constraints (Aba69; PZC19; DRK20; LCL23; DWDS23).

**Warm-start approach.** The NN predictions can serve as warm-start points for iterative solvers, potentially reducing the number of iterations required to reach the optimal/feasible solution (Die19; Bak19; SHAS24; GXW<sup>+</sup>24).

**Penalty training.** To reduce constraint violations of predicted solutions, various penalty functions, such as quadratic penalties, have been incorporated into the NN loss function (COMB19; PZC20; ZB20; FMVH20). Additionally, integration of the Karush–Kuhn–Tucker (KKT) conditions as equality constraints has been explored to refine NN performance (NC21b; NC21a; ZCZ21). However, these methods do not ensure solution feasibility.

**Provable training/verification.** A Preventive Learning framework is proposed for ensuring linear constraint feasibility without post-processing in (ZPC<sup>+</sup>20; ZPCL23), which adjusts inequality constraints to account for NN prediction errors. A NN editing method applies parametric linear relaxation to find NN weights and ensure output feasibility over the polytope (TT24). Additionally, NN verification techniques can also be applied to assess the worst-case performance after training (VQLC20; uAYKJ22; LAL<sup>+</sup>21).

**Set representation/approximation.** To guarantee feasibility, an inner approximation of the original constraint set

can be constructed. The convex combination of vertices and rays can represent feasible points of linear constraints (FNC20; ZSRZ21). For general compact constraint sets, the *probabilistic transformer* utilizes collected feasible samples to ensure feasibility (KZLD21). However, scalability remains a challenge due to the exponential growth in required samples with increasing dimensionality.

**Projection approach.** To enforce solution feasibility, orthogonal/L2 projection is often employed. However, solving projection problems either by iterative solvers or equivalent *optimization layers* (AK17; AAB<sup>+</sup>19; CDB<sup>+</sup>21; WZG<sup>+</sup>23; ZYZ<sup>+</sup>24) over complex constraints is computationally intensive. Alternative strategies include gradient-based methods (e.g., DC3 (DRK20)) and L2O models (HWFGY21; HFL<sup>+</sup>22). However, those projection-analogous methods do not guarantee feasibility for general constraints.

**Homeomorphic Projection** ensures NN solution feasibility over *ball-homeomorphic* (BH) constraints by constructing a homeomorphism between the constraint set and a unit ball using invertible NN, allowing efficient projection via bisection (LCL23; LCL24).

**Gauge function.** These works utilize gauge/Minkowski functions (BM08) to conduct projection or constrain NN output. For fixed convex sets, given an interior point, *gauge projection* can be applied to find feasible boundary solutions (Mha22). Similar approaches have been applied in the following works (LBGH23; THH23; KU23; LM23; TVH24). The gauge function can also be used to construct a bijection, known as gauge mapping, to constrain the NN output within a polytope (TZ22a; TZ22b; LKM23; LLC25).

In summary, existing approaches either have limited applicability or incur significant computational overhead. We propose **Bisection Projection** as a simple yet efficient framework to ensure NN solution feasibility over (fairly) general compact sets while maintaining bounded optimality loss. While algorithmically related to gauge-based methods, our method offers broader applicability and provides rigorous theoretical analysis.

### 3. Setting and Open Issue

We consider the following continuous optimization problem

$$\min_{x \in \mathbb{R}^n} f(x; \alpha) \quad \text{s.t. } x \in C; \quad (1)$$

where  $x \in \mathbb{R}^n$  denotes the decision variable and  $\alpha \in \mathbb{R}^d$  represents the input contextual parameter. Without loss of generality, the input domain and constraint set are assumed to be compact. The objective function  $f(x; \alpha)$  is Lipschitz continuous and potentially non-convex with optimal solution denoted as  $x^* \in \arg \min_{x \in C} f(x; \alpha)$ . The constraint set  $C$  is specified by a set of inequalities:  $g(x; \alpha) = [g_1(x; \alpha); \dots; g_{n_{\text{ineq}}}(x; \alpha)] \leq 0$ .

While equality constraints are not explicitly included in this formulation, certain classes of equality constraints of a constant rank can be embedded as neural network layers and satisfied exactly (Ababa9; Lee13; PZC19; DRK20; LCL23; DWDS23). We do carry out simulations for problems with linear/nonlinear equality constraints in Sec. 6 and provide detailed discussion in Appendix A.4.

We further specify the constraint set as follows, beyond those discussed in related works in Sec. 2.

Assumption 1.  $\mathbb{R}^2$ , (i) the constraint set  $C$  has positive measure; (ii) the set-valued mapping  $\mathcal{C} : \mathbb{R}^d \rightarrow \mathcal{C}$  is continuous in the Hausdorff metric.

We remark that assumption (i) guarantees the existence of interior points, which is fundamental to the bisection approach detailed in Sec. 4.1; and assumption (ii) ensures continuous deformation of the constraint set with respect to the input parameter, enabling the learning of continuous mappings from inputs to interior points as discussed in Sec. 4.3. These assumptions deliberately exclude unusual sets (e.g., vertices of a hypercube) from consideration in our continuous optimization problem, enabling rigorous algorithmic development and theoretical analysis. Despite these restrictions, Assumption 1 encompasses a broad class of constraint sets, including linear, convex, and ball-homeomorphic sets studied in existing works (ZPCL23; WZG23; LCL23; THH23; LCL24).

Open issue. As discussed in Sec. 2, various NN-based methods have demonstrated success in solving constrained optimization problems, offering low run-time complexity and small optimality gaps (PZC19; DRK20; PVH23; HCL24). We denote one such trained NN predictor  $\hat{f}(\alpha) : \mathbb{R}^d \rightarrow \mathbb{R}^n$  and define its prediction error as  $\epsilon_{\text{pre}} = \sup_{\alpha} \| \hat{f}(\alpha) - x^* \|$ . Due to the error  $\epsilon_{\text{pre}}$ , ensuring NN solution feasibility is non-trivial. As illustrated in Fig.

<sup>1</sup>i.e.,  $\exists x \in \text{int}(C); \epsilon > 0$  such that  $B(x; \epsilon) \subset C$ , where  $B(x; \epsilon)$  is the open ball centered at  $x$  with radius  $\epsilon$ .  
<sup>2</sup>i.e.,  $\exists \delta > 0; \epsilon > 0; \eta > 0$  such that  $d_H(C; C_0) < \delta$  whenever  $\| \alpha - \alpha_0 \| < \epsilon$ , where  $d_H$  denotes the Hausdorff distance.

Figure 1. NN predicting optimal solution (on boundary) may incur infeasibility; NN predicting interior points accommodates errors.

1, an optimal solution often lies on the constraint boundary with active constraints, making NN feasibility particularly challenging—Any positive prediction errors can push NN predictions outside of feasible regions

Current approaches, as summarized in Table 1, are either computationally intensive or fail to provide performance guarantees over general input-dependent constraint sets. To date, ensuring NN solutions feasibility for constraint optimization in (1) under Assumption 1, while maintaining bounded optimality loss and low computational complexity, remains an open and pressing challenge.

### 4. The Bisection Projection Framework

We propose Bisection Projection (BP) to “project” infeasible NN solutions onto the constraint set with low run-time complexity and minor optimality loss. As detailed in Sec. 4.1, this framework applies bisection to iteratively narrow the gap between infeasible points and one interior point (IP) to find feasible solutions. In Sec. 4.2, to bound the optimality loss induced by bisection, we introduce the concept of eccentricity for IP and establish its connection to projection distance. In Sec. 4.3, to reduce the inference time for finding IPs under varying inputs, we train another NN, denoted as IPNN, to predict IPs in real time.

#### 4.1. Bisection with Interior Points

Given an infeasible NN prediction  $x \notin C$  and an IP  $x_0 \in C$ , we can “project”  $x$  to  $C$  as:

$$\hat{x} = \text{BP}(x; x_0), \quad (\hat{x} - x) + x_0; \quad (2)$$

where  $\lambda \in [0, 1]$  leads to  $\hat{x} \in C$  and  $C$  is the boundary of  $C$ . As illustrated in Fig. 2, the “projected” solution  $\hat{x}$  is located on the straight line segment connecting the infeasible solution  $x$  and an IP  $x_0$ . We note that there could be multiple  $\lambda$  and corresponding  $\hat{x}$ , given a pair of  $x$  and  $x_0$ . To determine one such  $\lambda$ , we employ the bisection method, as elaborated in Alg. 1. We initiate by drawing a straight line connecting  $x$  with an IP  $x_0$ . This segment is guaranteed to intersect the boundary of the feasible region

Figure 2. The Bisection Projection framework: we apply one NN to predict a solution for the constraint optimization problem (near-optimal but may not be feasible), and another NN to predict interior points (robust and feasible); then we apply bisection (Alg. 1) to recover solution feasibility.

at least once. Subsequently, we apply the bisection algorithm to iteratively pinpoint one feasible solution along this segment toward the constraint boundary. Such a bisection operation is applicable to general compact sets with non-empty interiors (Assumption 1), and is also efficient with a linear convergence rate and maintains low per-iteration computational cost involving feasibility checks.

Remark 1. We note that the concept of projection-analogous operation with IPs is non-sophisticated, and some works also leverage similar ideas to “project” infeasible prediction (Mha22; THH23; KU23; LM23; TVH24; LCL24), see Appendix A.1 for details of these methods. However, these works primarily focus on fixed convex constraint sets where a single IP can be computed of ine for online deployment.

Two critical gaps remain in the literature: (i) theoretical analysis of how IP selection influences the quality of injected solutions; (ii) developing efficient schemes to obtain interior point’s centrality within the feasible region. For instance, the center of a unit ball achieves zero eccentricity.

Our work advances these fundamental issues by: (i) we first introduce the eccentricity of IP and establish its connection to the bisection-induced projection distance in Sec. 4.2. (ii)

We then employ another NN, called IPNN, to efficiently predict IPs under varying inputs for input-dependent constraints in Sec. 4.3. We also present a comprehensive performance analysis in Sec. 5.

#### 4.2. Minimum-Eccentricity IP for Bisection

We first define the eccentricity of IP, which is crucial for bounding the bisection-induced projection distance.

Definition 4.1 (Eccentricity of IP) For a compact set  $\mathcal{C}$  satisfying Assumption 1 with a non-empty interior, the eccentricity for one IP  $x \in \mathcal{C}$  with respect to a compact subset of boundary  $\mathcal{C}$  is defined as:

$$E(x; \mathcal{C}) = \max_{y \in \mathcal{C}} \|x - y\|_2 \min_{y \in \mathcal{C}} \|x - y\|_2 \quad (3)$$

#### Algorithm 1 Bisection for Feasibility.

Input: an NN prediction  $x \notin \mathcal{C}$  and an IP  $x \in \mathcal{C}$

- 1: set total iteration  $K$ ,  $l = 0$ , and  $u = 1$
- 2: for  $n = 1 : K$  do
- 3:   bisection:  $m = (l + u)/2$
- 4:   if  $x + m(x - x) \in \mathcal{C}$  then
- 5:     increase lower bound:  $l = m$
- 6:   else
- 7:     decrease upper bound:  $u = m$
- 8:   end if
- 9: end for

Output: feasible solution  $x = l(x - x) + x \in \mathcal{C}$

When  $\mathcal{C} = \mathbb{C}$ , eccentricity quantifies the variation in point-to-boundary distances, effectively measuring the interior point’s centrality within the feasible region. For instance, the center of a unit ball achieves zero eccentricity.

This measure directly relates to the optimality loss in bisection projection — lower eccentricity corresponds to smaller bisection-induced optimality gaps.

When  $\mathcal{C}$  is a subset of the boundary, eccentricity provides a localized characterization focused on specific boundary regions of interest. Particularly, when  $\mathcal{C}$  encompasses the projected solutions of neural network predictions with bounded errors, this local eccentricity measure yields tighter bounds on the optimality loss compared to the global eccentricity measure (i.e.  $E(x; \mathcal{C}_1) \leq E(x; \mathcal{C}_2)$  if  $\mathcal{C}_1 \subseteq \mathcal{C}_2$ ).

Next, we establish the connection between eccentricity measure and the bisection-induced projection distance.

Proposition 4.1. Let  $x = F(\cdot)$  be an infeasible NN prediction with bounded prediction error  $\|F(\cdot) - x\|_2 \leq k_{pre}$ ;  $x^* = BP(x; \mathcal{C})$  be the projected solution with  $x^* \in \mathcal{C}$ ; Then, the worst-case projection distance is bounded as:

$$\max_{x \in \mathcal{B}(x; k_{pre})} \|x - x^*\|_2 \leq BP(x; \mathcal{C}) + E(x; \mathcal{C});$$

where  $B(x; \text{pre})$  represents the NN prediction region, containing all possible NN predictions with an error  $\text{pre}$ ; and  $\mathcal{C} = \{x \in \mathbb{R}^d \mid f_{\text{BP}}(x; x) \leq \delta\}$  denotes a subset of the constraint boundary, containing all “projected” infeasible NN predictions.

The complete proof and a geometric illustration are included in Appendix B.1. We remark that the applicability of this bound extends to general compact sets under Assumption 1. Further, informed by Prop. 4.1, we seek to find an IP with minimized eccentricity (MEIP) with respect to constraint boundary  $\mathcal{C}$  to reduce the worst-case<sup>3</sup> bisection-induced projection distance for any infeasible NN solutions.

However, computing MEIP exactly presents significant challenges even for convex constraints, due to the non-convex boundary constraints (26). To address this, we can bound the eccentricity by finding a surrogate central point<sup>4</sup> such as the Chebyshev center (BBV04), defined as the IP maximizing the minimum distance to the boundary:  $\max_{x \in \mathcal{C}} \min_{y \in \mathcal{C}^c} \|x - y\|$ , which is also equivalent to minimizing the second term of the eccentricity in Def. 4.1.

Thus, this Chebyshev center serves as a relaxation for MEIP and provides an upper bound on the eccentricity. Moreover, it admits a tractable reformulation as the center of the largest Euclidean ball that can be inscribed on  $\mathcal{C}$ , which eliminates the non-convex boundary constraints in (26):

$$\max_{x \in \mathbb{R}^d} \min_{y \in \mathcal{C}^c} \|x - y\| \quad \text{s.t. } B(x; r) \subseteq \mathcal{C} \quad (4)$$

However, computing the Chebyshev center for constraint set  $\mathcal{C}$  under varying input poses significant computational challenges, particularly when rapid response times are essential for online applications. To overcome this limitation, we develop a learning-based approach in the following section.

We train another neural network (denoted as IPNN) of input dimension  $\mathbb{R}^d \rightarrow \mathbb{R}^n$ , to predict the IPs. We remark that such a continuous mapping from input to some IPs of interest exists under Hausdorff continuity specified in the Assumption 1.

### 4.3. Interior Points Neural Network (IPNN) Training

We utilize another neural network, denoted as IPNN:  $\mathbb{R}^d \rightarrow \mathbb{R}^n$ , to predict the IPs. We remark that such a continuous mapping from input to some IPs of interest exists under Hausdorff continuity specified in the Assumption 1.

<sup>3</sup>We consider minimize eccentricity  $E(x; \mathcal{C})$  with respect to the entire constraint boundary, which is prediction-agnostic and provides robust performance across different NN predictors over the same constraint set. In the appendix C, we also discuss prediction-aware eccentricity minimization when NN predictions or optimal solution data are given in advance.

<sup>4</sup>We provide a comprehensive review of different center definitions in Appendix A.3 to justify the design of MEIP.

### Algorithm 2 IPNN Training.

Input: Input dataset  $\{g_{i=1}^N\}$ , unit ball samples  $\{u_i\}_{i=1}^M$   
 1: Training epoch  $E$ , batch size  $B$ , IPNN  
 2: Initialize robust margins  $\delta = 10^{-2}$   
 3: for  $e = 1 : E$  do  
 4: Sampling batched data:  $\{g_{i=1}^B, f, u_i\}_{i=1}^B$   
 5: Loss:  $L(\theta; \mathcal{D}) = \frac{1}{B} \sum_{i=1}^B P(x_i + u_i; \theta) \log(\delta)$   
 6: Model update:  $\theta \leftarrow \text{Adam}(L(\theta; \mathcal{D}))$   
 7: end for  
 Output: Trained IPNN  $\theta$ .

We design the following loss function for IPNN training:

$$L(\theta; \mathcal{D}) = E_u [P(x + u; \theta)] \log(\delta) \quad (5)$$

where the IP prediction is denoted as  $x = \theta(x)$ . The first loss term,  $P(x + u; \theta) = k g(x + u; \theta)^+ k$ , denotes the constraint violation under the perturbed prediction with random samples  $u$  from a unit ball, and its expectation represents the penalty for the inscribed ball constraint  $B(x; \delta) \subseteq \mathcal{C}$  in (4). The regularization term  $\log(\delta)$ , represents the radius maximization objective (4) and is adjusted by a positive coefficient.

We also note that the loss function is analogous to the adversarial learning techniques like randomized smoothing (CRK19) with Gaussian perturbations and ensuring the predicted IPs maintain a robust margin from the constraint boundary. Finally, to optimize the average performance across different input parameters, we uniformly sample  $\mathcal{D}$  and minimize the total loss as  $L(\theta; \mathcal{D}) = E [L(\theta; \mathcal{D})]$ . The IPNN training procedure outlined in Alg. 2 involves sampling input parameters and unit vectors  $u$  and follows regular NN training procedures.

## 5. Performance Analysis

In this section, we present a comprehensive analysis of the BP framework: (i) the sufficient conditions for IPNN training for producing feasible IP under any input parameter in Sec. 5.1; (ii) the optimality loss and run-time complexity for the bisection operation in Sec. 5.2. We also discuss the connection to existing approaches (ZZ22b; LCL23; THH23) and IPNN training guarantees in Sec. 5.3.

### 5.1. Sufficient Conditions for IPNN Feasibility

Proposition 5.1. Let  $\mathcal{D}$  be an  $\epsilon$ -covering dataset for input domain as  $\bigcup_{i=1}^N B(x_i; r)$ ; the constraint violation function  $G(x; \theta) = \max_{1 \leq j \leq n_{\text{ineq}}} f_j g_j(x; \theta)$  with Lipschitz  $L_{G,x}$  and  $L_{G,g}$  for  $x$  and  $g$ , respectively; and IPNN is  $L$ -Lipschitz continuity for  $\theta$ . If  $G(\theta(x_i); \theta) + L_{G,x} L_{G,g} r \leq \delta$  for  $i = 1; \dots; N$ , then  $B(x_i; \delta) \subseteq \mathcal{C}$ .

The complete proof is included in Appendix B.2. Prop. 5.1 which is mitigated by the Chebyshev center-informed loss establishes sufficient conditions for the trained IPNN to generalize to unseen input parameters. These conditions require two key components: (i) the IPNN must achieve feasibility over finite training samples (i.e.,  $\mathbb{G}(\mathcal{C}; \mathcal{I}) \neq \emptyset$ ) — a condition readily satisfied in practice, as demonstrated by our empirical results in Sec. 6. The robust penalty loss in (5) effectively minimizes constraint violations for perturbed IP predictions. (ii) to extend feasibility guarantees to the entire input space  $\mathbb{R}^d$ , the IPNN must maintain a robust margin for IP prediction that accounts for generalization errors, which can be bounded by Lipschitz conditions (i.e.,  $L_{G;x} \leq L_r + L_G; r$ ).

While direct verification of these Lipschitz constants remains computationally intractable, they provide valuable insights into the relative difficulty of guaranteeing feasibility across different constraint types. Specifically, a smaller covering radius  $r_c$  is required for: “thin” constraint sets (i.e., small robust margin) or highly variable constraint geometries (i.e., large  $L_G$ ). For such challenging constraints, a smaller covering radius necessitates a larger number of training samples  $N$ , scaling as  $\mathcal{O}((\text{diam}(\mathcal{C})/r_c)^d)$  to adequately cover the input space.

We also remark that this condition is based on the Lipschitz-based worst-case analysis, while the empirical experiment shows that IPNN trained over less than 10,000 uniform-sample inputs already induces feasible IP prediction under unseen input parameters for high-dimensional problems.

5.2. Optimality and Run-time Complexity for Bisection

Theorem 1. Given constraint set  $\mathcal{C}$  under Assumption 1, an infeasible NN prediction  $x_{pre}$  with bounded error to the optimal solution  $x^*$  as  $\|x_{pre} - x^*\| \leq \epsilon$ , and valid IP prediction  $x \in \mathcal{C}$  produced by IPNN, after executing  $K$  steps of bisection shown in Alg. 1. We obtain a solution  $x$  satisfying the following:

(i) it is guaranteed to be feasible, i.e.,  $x \in \mathcal{C}$  ;  
 (ii) it has a bounded optimality gap  $\|x - x^*\| \leq 2\epsilon_{pre} + E(x; \mathcal{C}) + 2^{-K}(\epsilon_{pre} + \text{diam}(\mathcal{C}))$ ,  
 (iii) the run-time complexity is  $\mathcal{O}(KG)$ , where  $G$  is the complexity of checking the feasibility of a solution.

The complete proof is included in Appendix B.3. First, given an interior point, the bisection in Alg. 1 consistently returns a feasible solution. The optimality loss of the returned feasible solution is mainly bounded by three factors: the initial NN prediction error for the optimal solution, the eccentricity measure of the predicted IP, and the error due to finite-step bisection. (i) The initial prediction error is typically small, thanks to the NN’s universal approximation capabilities. (ii) The eccentricity represents the upper bound of the deviation caused by employing bisection with IPs,

5.3. Discussions

The algorithm’s run-time complexity, i.e., the number of arithmetic operations, is primarily affected by the number of bisection steps  $K$  and the complexity of checking feasibility at each step  $G$ . Such a feasibility checking procedure is also known as membership oracle which has been extensively analyzed in (Mha22; LBGH23). Further, for common convex sets, the bisection projection has a closed-form computation provided in (THH23).

Connection to existing works As discussed in Sec. 2, the BP framework is algorithmically related to the homeomorphic projection (LCL23; LCL24) and gauge function-based methods (TZ22a; Mha22; THH23). The following proposition reveals the connection between the BP framework and some existing schemes.

Proposition 5.2. The homeomorphic projection (LCL23) with gauge mapping (TZ22a) is equivalent to bisection projection over a convex set. The Gauge projection (Mha22; THH23) is equivalent to bisection projection with a fixed IP.

The complete proof is included in Appendix A.2. Thus, the bisection projection framework provides a unified view for some existing projection-analogous approaches over convex sets. Meanwhile, we highlight the theoretical analysis and application scenario for BP works on general compact sets under Assumption 1. It also achieves better performance in feasibility and speedup as shown in Sec. 6.

Availability and guarantees of IPs. Several existing NN feasibility approaches rely on IPs, including gauge mapping (TZ22a; TZ22b; LKM23), gauge projections (THH23; KU23; LM23; TVH24), and homeomorphic projections (LCL23; LCL24). While our BP framework similarly utilizes IPs, it advances the state-of-the-art through: general constraints (Sec. 3), eccentricity-based optimality bound (Sec. 4), and IPNN loss design (5). Despite these advances, the guarantees for NN-based IP findings depend on NN training by minimizing penalty-based loss functions. While exact convergence has been established for over-parameterized NNs (L18; LZB22), The general convergence analysis remains challenging for practical scenarios involving finite-size networks and non-convex penalties and warrants future exploration.

Extension to Multiple Interior Points . The BP framework naturally extends to multiple IPs. Specifically, we can perform bisection from multiple IPs and select the projected point with the minimal projection distance. A detailed discussion of this extension is provided in Appendix A.5.

## 6. Numerical Experiments

We first consider a toy example to visualize the training and testing performance of our framework in Sec. 6.1. We then carry out comprehensive simulations to validate the efficiency of BP against existing methods on various constrained optimization problems in Sec. 6.2. We also demonstrate the efficacy of key design and parameters in the BP framework through sensitivity analysis in Sec. 6.3. The detailed experimental setting and problem formulations are provided in D.

### 6.1. Illustrative Toy Examples

Figure 3. Perturbed penalty loss (left) and IPNN prediction feasibility rate (right) during training.

Figure 4. Robust margin (left) and estimated eccentricity (right), calculated by the gap between maximum and minimum IP-to-boundary distances.

Figure 5. BP with IPNN prediction given test input parameters.

To validate that the proposed training algorithm for IPNN can indeed produce low-eccentricity points or approximated Chebyshev centers, we consider a non-convex quadratic constraint set  $C = \{x \mid x^T Q_i x + q_i^T x + b_i \leq 0; i = 1, \dots, 6\}$ , where the input parameter is defined as  $\theta = [q; q_0]$ . We train IPNN over such constraint sets and visualize its training and testing performance. Our experiments yield the following observations: (i) Minimizing the penalty loss successfully improves feasibility over test input parameters, as shown in Fig. 3; (ii) Maximizing the robust margin indeed reduces the eccentricity of IPNN predictions, as shown in Fig. 4; (iii) After training, bisection projection with IPNN-predicted “central” IPs incurs low projection distance, as

shown in Fig. 5. We provide comprehensive visualizations from Fig. 12 to 17 in Appendix E.

### 6.2. NN feasibility for Constrained Optimization

**Dataset** We apply the BP framework to four benchmark convex problems (QP, convex QCQP, SOCP, and SDP) and two non-convex real-world scenarios, including optimal power flow problems in grid operation (AC-OPF) and joint chance-constrained problems in inventory management (JC-CIM). We follow the established parameter configuration and sampling strategy from available codes in previous works (DRK20; LCL23). We first train an NN predictor to learn the mapping from input parameters to the optimal solutions in existing works (DRK20), where the training and testing data are generated by randomly sampling the input parameter and solve the corresponding optimal solutions through iterative solvers as ground truth (DRK20; LCL23).

**Baselines** (i) **Optimizer**: for convex optimization, we use MOSEK to solve the optimal solution. For JC-CIM, we adopt its scenario-based approximation and solve it by MOSEK (PAS09; For AC-OPF problems, we adopt PY-POWER as the specialized solver (MS11)); (ii) **NN**: it directly maps the input parameter to the solution without post-processing; (iii) **WS**: The infeasible prediction of NN is regarded as the warm-start initialization for the iterative solver; (iv) **Proj**: the infeasible predicted solution by NN is processed by orthogonal projection and solved with the iterative solver; (v) **D-Proj**: this is proposed in DC3 (DRK20), which applies gradient descent with equality completion to minimize the constraint violation; (vi) **H-Proj**: the homeomorphic projection are applied to the infeasible predictions (LCL23); (vii) **B-Proj**: we apply bisection in Alg. 1 with predicted IPs to recover the feasibility. To ensure the feasibility of equality constraints, we utilize predict-then-reconstruct technique (PZCZ20; DRK20), as detailed in Appendix A.4. Note that some baselines shown in Table 1 are not included due to their limited applicability.

Table 2 summarizes our experimental results across six constraint optimization problems, revealing several key insights. The BP framework consistently achieves 100% feasibility for initially infeasible NN predictions while offering up to four orders of magnitude speedup compared to standard projection approaches, all while maintaining competitive optimality loss. Direct NN outputs cannot guarantee complete feasibility due to prediction errors that may push solutions outside the constraint set. Iterative solver-based methods such as warm-start and orthogonal projection ensure feasibility with minimal optimality loss but incur substantial computational overhead (exceeding 400 seconds for QCQP problems). The gradient-based D-Proj method, though applicable across different constraint sets, fails to guarantee feasibility and exhibits high sensitivity to step size selection. While

Table 2. Performance comparison for constrained optimization problems.

Method	Feasibility rate (%)	Solution opt. (%)	Objective opt. (%)	Pred. cost(s)	Post. cost(s)	Feasibility rate (%)	Solution opt. (%)	Objective opt. (%)	Pred. cost(s)	Post. cost(s)
QP (n=400,d=100,n <sub>eq</sub> =100,n <sub>ineq</sub> =100)						QCQP (n=400,d=100,n <sub>eq</sub> =100,n <sub>ineq</sub> =100)				
NN	80.8	2.26	0.97			92.0	3.91	3.26		
NN+WS	100	1.81	0.79		5.37	100	3.58	3.01		434
NN+Proj	100	2.26	0.97	0.0014	5.23	100	3.91	3.26	0.0017	401
NN+D-Proj	80.9	2.26	0.97		11.1	92.3	3.91	3.26		7.23
NN+H-Proj	100	16.49	15.42		0.763	100	3.95	3.31		0.413
NN+B-Proj	100	2.31	1.00		0.0160	100	3.94	3.30		0.0162
SOCP (n=400,d=100,n <sub>eq</sub> =100,n <sub>ineq</sub> =100)						SDP (n=40x40,d=40,n <sub>eq</sub> =40,n <sub>ineq</sub> =1)				
NN	89.6	1.24	0.55			58.0	2.10	2.74		
NN+WS	100	1.10	0.50		136	100	1.20	1.57		105
NN+Proj	100	1.24	0.55	0.0015	128	100	2.10	2.74	0.0017	158
NN+D-Proj	95.5	1.24	0.55		5.03	58.0	2.10	2.74		236
NN+H-Proj	100	1.25	0.56		0.571	100	31.51	33.67		0.958
NN+B-Proj	100	1.25	0.56		0.0272	100	2.26	2.96		0.0163
AC-OPF (n=476,d=400,n <sub>eq</sub> =400,n <sub>ineq</sub> =1042)						JCC-IM (n=400,d=100,n <sub>eq</sub> =0,n <sub>ineq</sub> =10,100)				
NN	94.0	0.15	0.001			84.5	1.94	1.40		
NN+WS	100	0.14	0.001		4.73	100	1.64	1.21		48.3
NN+Proj	100	0.25	0.002	0.251	14.1	100	1.94	1.40	0.0013	128
NN+D-Proj	96.5	0.15	0.001		14.9	84.5	1.94	1.40		81.1
NN+H-Proj	100	0.63	0.042		1.32	100	10.24	10.39		0.876
NN+B-Proj	100	0.15	0.001		1.13	100	1.98	1.45		0.193

<sup>1</sup> Evaluation metrics: (i) Feasibility: Percentage of feasible solutions among 1,024 test instances, where feasibility requires satisfying equality and inequality constraints within tolerance  $10^{-5}$ ; (ii) Optimality: Mean absolute percentage error (MAPE) between output and optimal solutions for both decision variables and objective values; (iii) Running-time: Total inference time comprising NN predictions and post-processing for constraint satisfaction. Iterative solvers are parallelized when computing projections or warm-start solutions.

<sup>2</sup> n and d represent the dimensions for input parameter and output decision variables, respectively. n<sub>eq</sub> and n<sub>ineq</sub> denote the number of equality and inequality constraints, respectively.

H-Proj also achieves 100% feasibility, it introduces larger Prop. 5.1 establishes that increasing both training sample optimality gaps and is less efficient than B-Proj, particularly for high-dimensional constraint sets (e.g., SDP), due to its guarantees on unseen inputs. We empirically validate complex invertible NN (INN) training and computationally intensive INN calculations during inference.

In summary, across both convex and non-convex constraint sets, BP demonstrates superior performance by achieving perfect feasibility or significantly reduced computational complexity while maintaining comparable optimality loss.

### 6.3. Sensitivity Analysis for BP Framework

We investigate the impact of key components in the BP framework to validate their effectiveness.

Impacts of  $\epsilon$  on out-of-sample feasibility (Fig. 6)

with maximizing  $\epsilon$ , trained with fixed  $\epsilon$ , and trained without  $\epsilon$ , under varying training sample sizes. Fig. 6 demonstrates that maximizing the robust margin improves out-of-sample feasibility (over 1024 test inputs) compared to fixed or zero-margin approaches. These improvements are particularly pronounced under limited training data scenarios, confirming the effectiveness of our proposed Chebyshev center-informed loss function (5).

Impacts of  $\epsilon$  on optimality of projected solution (Fig. 7):

Figure 7. Distribution of projection distances for IPNN trained with and without  $\epsilon$  over QP (left) and JCC-IM (right).

Figure 6. Feasibility rates over unseen test instances for QP (left) and JCC-IM (right) under varying training sample sizes. We conducted an ablation study to evaluate the effectiveness of incorporating eccentricity minimization (approximated

by maximizing  $\lambda$  on the projection loss. We generated bisection stepsize (e.g.,  $\lambda = 0:1$  instead of the standard 10,000 infeasible test instances by random Gaussian sampling with varying variance magnitudes (noise levels shown in Fig. 7). For each infeasible point, we applied bisection approach guides the solution trajectory incrementally from the projection using three IPNN variants: trained with maximizing infeasible prediction toward the feasible boundary, typically resulting in an intersection point closer to the original neural network output.

Fig.7 shows the distributions of projection distances (measured between infeasible points and their respective projections). While this strategy better preserves the quality of initial predictions, it necessitates additional iterations due to the IPNN (trained with  $\lambda$ ) produces smaller projection distances, smaller stepsize. This trade-off between solution quality and confirming the theoretical bounds established in Prop. 4.1 computational efficiency should be considered based on the. This improved projection quality directly translates to better specific application requirements and constraint geometry. better optimality preservation when correcting infeasible NN predictions.

Impacts of number of bisection steps (Fig. 8)

Figure 8. Effect of the number of bisection steps ( $k$ ) on optimality gap and running time for QP (left) and JCCIM (right).

We investigated how the number of bisection steps during inference affects both solution quality and computational efficiency. As shown in Fig. 8, and in accordance with Theorem 1, the optimality gap decreases exponentially as the number of bisection steps increases. Concurrently, the running time exhibits linear growth due to the sequential nature of the computation, continuing until reaching the convergence threshold.

Notably, our experiments demonstrate that for real-time applications, only a small number of bisection steps (typically 5-10) are sufficient to achieve well-converged feasible solutions, offering an excellent trade-off between solution quality and computational efficiency. This confirms the practical utility of our approach in time-sensitive decision-making contexts.

Impacts of bisection stepsize on optimality (Fig. 19)

As illustrated in Fig. 2, when multiple intersections exist between the projection ray and the feasible set boundary, the bisection algorithm may converge to any of these intersection points. While this phenomenon did not occur in our benchmark optimization problems, it remains a theoretical concern for constraint sets with complex geometries.

To address potential convergence to boundary points distant from the initial prediction, we may employ a reduced

Scalability on constraint dim. and decision dim: We further remark on the BP framework's scalability based on large-scale problems in Table 2. For joint chance constraints, where constraint dimension grows linearly with sampled scenarios, iterative solver-based approaches face memory limitations (NS06). For AC-OPF in large-scale power grids, non-linear power balance and branch flow constraints incur high computational complexity for existing solvers (MS11). Our bisection methods require only constraint checking per iteration, with GPU-based batch processing further accelerating these calculations.

## 7. Conclusion and Limitation

We introduce Bisection Projection, an efficient scheme to project infeasible NN predictions onto general compact constraint sets through bisection. We establish the connection between the eccentricity of interior points (IPs) and projection distance, then employ IPNN to efficiently predict IPs. Our theoretical analysis provides sufficient conditions for IPNN feasibility and proves bounded optimality loss under IP predictions. Extensive simulations demonstrate that bisection projection outperforms existing methods in feasibility and efficiency with comparable optimality.

Our framework has several limitations, suggesting future research directions: (i) extending BP to discrete constraints such as mixed-integer problems for broader applicability, (ii) jointly optimizing interior point selection and bisection trajectory to further reduce optimality gaps, (iii) and exploiting problem-specific structures like symmetry and sparsity to design NN/IPNN to improve training efficiency.

## Acknowledgements

This work is supported in part by a General Research Fund from Research Grants Council, Hong Kong (Project No. 11200223), a Collaborative Research Fund from Research Grants Council, Hong Kong (Project No. C1049-24G), an InnoHK initiative, The Government of the HKSAR, Laboratory for AI-Powered Financial Technologies, and a Shenzhen-Hong Kong-Macau Sci-

ence & Technology Project (Category C, Project No. SGDX20220530111203026). The authors would also like to thank the anonymous reviewers for their helpful comments.

## Code Availability

Code for data generation and model training is available at [Github](#).

## Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

## References

- [AAB<sup>+</sup>19] 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.
- [Aba69] Jean Abadie. Generalization of the wolfe reduced gradient method to the case of nonlinear constraints. *Optimization*, pages 37–47, 1969.
- [AK17] Brandon Amos and J Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning* pages 136–145. PMLR, 2017.
- [Amo22] Brandon Amos. Tutorial on amortized optimization for learning to optimize over continuous domains. *arXiv preprint arXiv:2202.00665*, 2022.
- [Bak19] Kyri Baker. Learning warm-start points for an optimal power flow. In *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP) 2019*.
- [BBV04] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.
- [BM08] Franco Blanchini and Stefano Miani. Set-theoretic methods in control volume 78. Springer, 2008.
- [CCC<sup>+</sup>21] Tianlong Chen, Xiaohan Chen, Wuyang Chen, Howard Heaton, Jialin Liu, Zhangyang Wang, and Wotao Yin. Learning to optimize: A primer and a benchmark. *arXiv preprint arXiv:2103.12828* 2021.
- [CDB<sup>+</sup>21] Bingqing Chen, Priya L Donti, Kyri Baker, J Zico Kolter, and Mario Berghs. Enforcing policy feasibility constraints through differentiable projection for energy optimization. In *Proceedings of the Twelfth ACM International Conference on Future Energy Systems*, pages 199–210, 2021.
- [COMB19] Richard Cheng, Gabor Orosz, Richard M Murray, and Joel W Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. *Proceedings of the AAAI Conference on Artificial Intelligence* volume 33, pages 3387–3395, 2019.
- [CRK19] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International conference on machine learning* pages 1310–1320. PMLR, 2019.
- [Die19] Frederik Diehl. Warm-starting an optimal power flow with graph neural networks. In *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*, pages 1–6, 2019.
- [DRK20] Priya L Donti, David Rolnick, and J Zico Kolter. Dc3: A learning method for optimization with hard constraints. In *International Conference on Learning Representation*, 2020.
- [DWDS23] Shutong Ding, Jingya Wang, Yali Du, and Ye Shi. Reduced policy optimization for continuous control with hard constraints. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.
- [FMVH20] Ferdinando Fioretto, Terrence WK Mak, and Pascal Van Hentenryck. Predicting an optimal power flows: Combining deep learning and lagrangian dual methods. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 630–637, 2020.
- [FNC20] Thomas Frerix, Matthias Nießner, and Daniel Cremers. Homogeneous linear inequality constraints for neural network activations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 748–749, 2020.
- [GWWM19] Neel Guha, Zhecheng Wang, Matt Wytock, and Arun Majumdar. Machine learning for an optimal power flow. *arXiv preprint arXiv:1910.08842* 2019.
- [GXW<sup>+</sup>24] Xi Gao, Jinxin Xiong, Akang Wang, Jiang Xue, Qingjiang Shi, et al. Im-1stm: A learning-based interior point method for solving nonlinear programs. *Advances in Neural Information Processing Systems* 37:122891–122916, 2024.
- [HCL24] Wanjun Huang, Minghua Chen, and Steven H Low. Unsupervised learning for solving an optimal power flows: Design, analysis, and experiment. *IEEE Transactions on Power Systems*, 2024.
- [HFL<sup>+</sup>22] Howard Heaton, Samy Wu Fung, Alex Tong Lin, Stanley Osher, and Wotao Yin. Wasserstein-based projections with applications to inverse problems. *SIAM Journal on Mathematics of Data Science* 4(2):581–603, 2022.
- [HSW89] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks* 2(5):359–366, 1989.

- [HWFGY21] Howard Heaton, Samy Wu Fung, Aviv Gibali, and Wotao Yin. Feasibility-based fixed point networks. *Fixed Point Theory and Algorithms for Sciences and Engineering* 2021(1):1–19, 2021.
- [HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [KFBVHW21] James Kotary, Ferdinando Fioretto, Pascal Van Hentenryck, and Bryan Wilder. End-to-end constrained optimization learning: A survey. *arXiv preprint arXiv:2103.16378* 2021.
- [KU23] Andrei V Konstantinov and Lev V Utkin. A new computationally simple approach for implementing neural networks with output hard constraints. In *Doklady Mathematics*, volume 108, pages S233–S241. Springer, 2023.
- [KZLD21] Anastasis Kratsios, Behnoosh Zamanlooy, Tianlin Liu, and Ivan Dokmanić. Universal approximation under constraints is possible with transformers. In *International Conference on Learning Representations*, 2021.
- [LAL<sup>+</sup>21] Changliu Liu, Tomer Arnon, Christopher Lazarus, Christopher Strong, Clark Barrett, Mykel J Kochenderfer, et al. Algorithms for verifying deep neural networks. *Foundations and Trends in Optimization*, 4(3-4):244–404, 2021.
- [LBGH23] Zhou Lu, Nataly Brukhim, Paula Gradu, and Elad Hazan. Projection-free adaptive regret with membership oracles. *International Conference on Algorithmic Learning Theory*, pages 1055–1073. PMLR, 2023.
- [LC23] Enming Liang and Minghua Chen. Generative learning for solving non-convex problem with multi-valued input-solution mapping. *The Twelfth International Conference on Learning Representations* 2023.
- [LCL23] Enming Liang, Minghua Chen, and Steven H. Low. Low complexity homeomorphic projection to ensure neural-network solution feasibility for optimization over (non-)convex set. *International Conference on Machine Learning*. PMLR, 2023.
- [LCL24] Enming Liang, Minghua Chen, and Steven H. Low. Homeomorphic projection to ensure neural-network solution feasibility for constrained optimization. *Journal of Machine Learning Research*, 2024.
- [Lee13] John M Lee. *Smooth manifolds. Introduction to smooth manifolds*, pages 1–31. Springer, 2013.
- [LKM23] Meiyi Li, Soheil Kolouri, and Javad Mohammadi. Learning to solve optimization problems with hard linear constraints. *IEEE Access*, 2023.
- [LLC24] Chenghao Liu, Enming Liang, and Minghua Chen. Characterizing resnet’s universal approximation capability. In *Forty-first International Conference on Machine Learning* 2024.
- [LLC25] Xinpeng Li, Enming Liang, and Minghua Chen. Gauge flow matching for efficient constrained generative modeling over general convex set. *ICLR 2025 Workshop on Deep Generative Model in Machine Learning: Theory, Principle and Efficiency* 2025.
- [LLPS93] Moshe Leshno, Vladimir Ya Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural networks* 6(6):861–867, 1993.
- [LM23] Meiyi Li and Javad Mohammadi. Toward rapid, optimal, and feasible power dispatch through generalized neural mapping. *arXiv preprint arXiv:2311.04838* 2023.
- [LZB22] Chaoyue Liu, Libin Zhu, and Mikhail Belkin. Loss landscapes and optimization in over-parameterized non-linear systems and neural networks. *Applied and Computational Harmonic Analysis*, 59:85–116, 2022.
- [MH<sup>+</sup>19] Daniel K Molzahn, Ian A Hiskens, et al. *A survey of relaxations and approximations of the power flow equations*. 2019.
- [Mha22] Zakaria Mhammedi. Efficient projection-free online convex optimization with membership oracle. In *Conference on Learning Theory*, pages 5314–5390. PMLR, 2022.
- [NC21a] Rahul Nellikkath and Spyros Chatzivasileiadis. Physics-informed neural networks for ac optimal power flow. *arXiv preprint arXiv:2110.02672* 2021.
- [NC21b] Rahul Nellikkath and Spyros Chatzivasileiadis. Physics-informed neural networks for minimising worst-case violations in dc optimal power flow. In *2021 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, pages 419–424. IEEE, 2021.
- [NS06] Arkadi Nemirovski and Alexander Shapiro. Scenario approximations of chance constrained probabilistic and randomized methods for design under uncertainty. *pages 3–47*, 2006.
- [NSW22] Deanna Needell, William Swartworth, and David P Woodruff. Testing positive semidefiniteness using linear measurements. *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)* pages 87–97. IEEE, 2022.
- [NW99] Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 1999.
- [PAS09] Bernardo K Pagnoncelli, Shabbir Ahmed, and Alexander Shapiro. Sample average approximation method for chance constrained programming: theory and applications. *Journal of optimization theory and applications*, 142(2):399–416, 2009.

- [PCZL22] Xiang Pan, Minghua Chen, Tianyu Zhao, and Steven H Low. Deepopf: A feasibility-optimized deep neural network approach for ac optimal power flow problems. *IEEE Systems Journal*, pages 42–47, 2022.
- [PVH23] Seonho Park and Pascal Van Hentenryck. Self-supervised primal-dual learning for constrained optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 4052–4060, 2023.
- [PZC19] Xiang Pan, Tianyu Zhao, and Minghua Chen. Deepopf: A deep neural network approach for security-constrained dc optimal power flow. *2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)* 2019.
- [PZCZ20] Xiang Pan, Tianyu Zhao, Minghua Chen, and Shengyu Zhang. Deepopf: A deep neural network approach for security-constrained dc optimal power flow. *IEEE Transactions on Power Systems* 36(3):1725–1735, 2020.
- [SHAS24] Rajiv Sambharya, Georgina Hall, Brandon Amos, and Bartolomeo Stellato. Learning to warm-start xed-point optimization algorithms. *Journal of Machine Learning Research* 25(166):1–46, 2024.
- [SJL18] Mahdi Soltanolkotabi, Adel Javanmard, and Jason D Lee. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *IEEE Transactions on Information Theory* 65(2):742–769, 2018.
- [THH23] Jesus Tordesillas, Jonathan P How, and Marco Hutter. Rayen: Imposition of hard convex constraints on neural networks. *arXiv preprint arXiv:2307.08336* 2023.
- [TT24] Zhe Tao and Aditya V Thakur. Provable editing of deep neural networks using parametric linear relaxation. *Advances in Neural Information Processing Systems* 37:113846–113883, 2024.
- [TVH24] Mathieu Tanneau and Pascal Van Hentenryck. Dual lagrangian learning for conic optimization. *The Thirty-eighth Annual Conference on Neural Information Processing Systems* 2024.
- [TZ22a] Daniel Tabas and Baosen Zhang. Computationally efficient safe reinforcement learning for power systems. In *2022 American Control Conference (ACC)* pages 3303–3310. IEEE, 2022.
- [TZ22b] Daniel Tabas and Baosen Zhang. Safe and efficient model predictive control using neural networks: An interior point approach. *2022 IEEE 61st Conference on Decision and Control (CDC)* pages 1142–1147. IEEE, 2022.
- [uAYKJ22] Zain ul Abdeen, He Yin, Vassilis Kekatos, and Ming Jin. Learning neural networks under input-output specifications. In *2022 American Control Conference (ACC)* pages 1515–1520. IEEE, 2022.
- [VQLC20] Andreas Venzke, Guannan Qu, Steven Low, and Spyros 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)* pages 1–7. IEEE, 2020.
- [WZG<sup>+</sup> 23] Runzhong Wang, Yunhao Zhang, Ziao Guo, Tianyi Chen, Xiaokang Yang, and Junchi Yan. Linsat-net: the positive linear satisfiability neural networks. In *International Conference on Machine Learning* pages 36605–36625. PMLR, 2023.
- [ZB20] Ahmed S Zamzam and Kyri 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)* pages 1–6. IEEE, 2020.
- [ZCZ21] Ling Zhang, Yize Chen, and Baosen Zhang. A convex neural network solver for dcoptf with generalization guarantees. *IEEE Transactions on Control of Network Systems* 2021.
- [ZMS11] Ray D Zimmerman and Carlos E Murillo-Sanchez. *Matpower 4.1 user's manual*. Power Systems Engineering Research Center (PSERC), 2011.
- [ZMSG97] Ray D Zimmerman, Carlos E Murillo-Sanchez, and Deqiang Gan. *Matpower*. PSERC. [Online]. Software Available at: <http://www.pserc.cornell.edu/matpower/> 1997.
- [ZPC<sup>+</sup> 20] Tianyu Zhao, Xiang Pan, Minghua Chen, Andreas Venzke, and Steven H Low. Deepopf+: A deep neural network approach for dc optimal power flow for ensuring feasibility. In *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)* pages 1–6. IEEE, 2020.
- [ZPCL23] Tianyu Zhao, Xiang Pan, Minghua Chen, and Steven H Low. Ensuring dnn solution feasibility for optimization problems with convex constraints and its application to dc optimal power flow problems. In *International Conference on Learning Representations* 2023.
- [ZSRZ21] Liyuan Zheng, Yuanyuan Shi, Lillian J Ratliff, and Baosen Zhang. Safe reinforcement learning of control-affine systems with vertex networks. In *Learning for Dynamics and Control* pages 336–347. PMLR, 2021.
- [ZYZ<sup>+</sup> 24] Hongtai Zeng, Chao Yang, Yanzhen Zhou, Cheng Yang, and Qinglai Guo. Glinsat: The general linear satisfiability neural network layer by accelerated gradient descent. *The Thirty-eighth Annual Conference on Neural Information Processing Systems* 2024.

# Contents

A	Discussion and Connection of Related Work	13
A.1	Availability of Interior Points	13
A.2	Connections to Related Works (Prop. 5.2)	14
A.3	Comparison of Different Centers	15
A.4	Tackling Equality Constraint	16
A.5	Extension BP to Multiple Interior Points	17
B	Proof for Main Results	18
B.1	Proof for Bisection-induced Projection Distance (Prop. 4.1)	18
B.2	Proof for Feasibility Guarantee in (Prop. 5.1)	19
B.3	Proof for Optimality Gap and Run-time Complexity (Theorem 1)	19
C	Training for IPNN	20
C.1	Prediction-agnostic training	20
C.2	Prediction-aware training	21
D	Data and Experiment Setting	21
D.1	Formulation for Optimization Problems	21
D.2	Experiment Setting	24
D.3	Hyper-parameters of NN and IPNN	25
E	Supplementary Experiment Results	26
E.1	IPNN Training and Bisection Projection over Various Constraint Sets	26
E.2	Sensitivity Analysis on for Out-of-Sample Feasibility and Projection Distance	27
E.3	Sensitivity Analysis on Bisection Steps on Optimality Gap and Iteration Complexity	27

## A. Discussion and Connection of Related Work

### A.1. Availability of Interior Points

We note that there is a line of research on NN feasibility operating under assumptions regarding the availability of IPs.

- **Gauge Mapping (for Linear Sets):** This approach was proposed in [TZ22a, TZ22b], establishing a bijective mapping between a unit cube and a polytope, which can be used as the output layer of neural networks to ensure feasibility over polytopes. The method relies on input-dependent IPs (IPs) and assumes the existence of an affine mapping from input to IP, solving this mapping through semidefinite programming (SDP). However, it does not guarantee the existence of such an affine policy. LOOP-LC [KM23] directly assumes the existence of an input-invariant IP and solves it by linear constraint residual minimization, but it also does not provide a guarantee or sufficient conditions for such an invariant IP.
- **Gauge Projection (for Convex Sets):** Gauge projection [Ma22] restores solution feasibility by scaling infeasible solutions along rays from a (fixed) of infeasible IP to find feasible solutions at the constraint boundary. This projection technique is differentiable and computationally efficient, which has been successfully integrated into neural network architectures to enforce output feasibility with respect to input-invariant convex sets. Notable implementations include RAYEN [THH23] (supporting linear, quadratic, second-order cone, and linear matrix inequality constraints), ConstraintNet [KU23] (handling linear and quadratic constraints), LOOP-LC [LM23] (for linear sets), and the radial projection approach [TVH24] (applicable to several convex cones).
- **Homeomorphic Projection (for Ball-homeomorphic Sets):** It ensures NN solution feasibility over ball-homeomorphic constraints by constructing a homeomorphism between the constraint set and a unit ball using invertible NN (INN), allowing efficient projection via bisection [LCL23; LCL24]. This approach also relies on a valid INN to map the center of a unit ball to an IP of the constraint set. It provides a sufficient condition for feasibility guarantee [LCL23; LCL24]. However, INNs have higher training and inference complexity than regular fully connected neural networks due to their sophisticated architectural design requirements.

- Bisection Projection (for General sets): In this work, we consider a more general input-dependent constraint set and propose an efficient bisection projection framework. We first characterize the eccentricity of the IP, which is directly related to the bisection-induced optimality loss. We then relax the minimum-eccentricity IP problem into a tractable Chebyshev center-based formulation and employ another neural network, called IPNN, to efficiently predict IPs under varying inputs for input-dependent constraints. We also provide a sufficient condition for IPNN feasibility similar to (LCL23; LCL24), but applicable to more general sets beyond the ball-homeomorphic ones.

A.2. Connections to Related Works (Prop. 5.2)

As discussed in Sec. 2, the proposed framework is conceptually related to the homeomorphic projection (LCL24) and gauge function based methods (TZ22b; THH23).

Definition A.1 (Gauge/Minkowski function (M08)). Let  $C \subseteq \mathbb{R}^n$  be a compact convex set with a non-empty interior. The Gauge/Minkowski function  $c_C : \mathbb{R}^n \setminus \text{int}(C) \rightarrow \mathbb{R}_+$  is defined as

$$c_C(x; x) = \inf \{ \alpha > 0 \mid x \in \alpha C \}; \tag{6}$$

where  $x \in \text{int}(C)$  is an IP of  $C$ .

The Gauge function generalizes the concept of a norm. For a set  $C$  that is symmetric about the origin, the gauge function  $c_C(x; 0)$  defines a norm. In particular, when  $C = B_p = \{x \in \mathbb{R}^n \mid \|x\|_p \leq 1\}$  is the unit ball of the  $p$ -norm, we have  $c_{B_p}(x; 0) = \|x\|_p$ .

Based on the gauge function, we can construct the following bijection between two compact convex sets:

Definition A.2 (Gauge Mapping (TZ22a)). Let  $Z, X \subseteq \mathbb{R}^n$  be compact convex sets with  $z \in \text{int}(Z)$  and  $x \in \text{int}(X)$ , respectively.

The gauge mapping  $\gamma : Z \rightarrow X$  is defined as:

$$\gamma(z) = \frac{z(z - z; z)}{x(z - z; x)}(z - z) + x; \quad z \in Z \tag{7}$$

The inverse mapping  $\gamma^{-1} : X \rightarrow Z$  is given by:

$$\gamma^{-1}(x) = \frac{x(x - x; x)}{z(x - x; z)}(x - x) + z; \quad x \in X \tag{8}$$

- In essence, the gauge mapping scales the boundary of a convex set from an IP to another convex set and with translation to its IP.
- When  $Z$  is a unit  $p$ -norm ball, the gauge mapping is simplified as:

$$\gamma(z) = \frac{\|z\|_p}{\|z - z\|_p} z + x; \quad z \in B; \quad \gamma^{-1}(x) = \frac{c_B(x - x; x)}{\|x - x\|_p} (x - x); \quad x \in C; \tag{9}$$

Definition A.3 (Gauge Projection (Mha22; THH23)). Let  $C \subseteq \mathbb{R}^n$  be a compact convex set with  $x \in \text{int}(C)$ . For any  $x \in \mathbb{R}^n \setminus C$ , the Gauge Projection  $\mathbb{G}_C : \mathbb{R}^n \setminus C \rightarrow \partial C$  is defined as

$$\mathbb{G}_C(x) := x + \frac{x - x}{c_C(x - x; x)} \in \partial C; \tag{10}$$

where  $\partial C$  denotes the boundary of  $C$ .

Definition A.4 (Homeomorphic Projection (LCL23)). Let  $K \subseteq \mathbb{R}^n$  be a compact set that is homeomorphic to the unit ball. Let  $\mathbb{H} : B \rightarrow K$  be a homeomorphism with inverse mapping  $\mathbb{H}^{-1} : K \rightarrow B$ . For any point  $x \in \mathbb{R}^n \setminus K$ , the Homeomorphic Projection  $\mathbb{H}_K : \mathbb{R}^n \setminus K \rightarrow \partial K$  is defined as:

$$\mathbb{H}_K(x) := \mathbb{H}(\mathbb{B}(\mathbb{H}^{-1}(x))); \tag{11}$$

where  $\mathbb{B}$  is the Euclidean projection operator.

The following observation reveals the connection between the bisection projection framework and some existing schemes.

Proposition A.1. The homeomorphic projection (L23) with gauge mapping (Z22) is equivalent to bisection projection over a convex set. The gauge projection (Mha22) is a special case of bisection projection with one fixed IP.

Proof. Let's consider applying the gauge mapping to the homeomorphic projection for a compact convex set. Then we can simplify the homeomorphic projection operator as:

$$\mathcal{H} = (\text{Proj}_B(\frac{1}{c(x; X)}(x - x))) = (\text{Proj}_B(\frac{c(x; X)}{kx - xk}(x - x))) \tag{12}$$

$$= (\frac{\frac{c(x; X)}{kx - xk}(x - x)}{k\frac{c(x; X)}{kx - xk}(x - x)k}) = (\frac{x - x}{kx - xk}) \tag{13}$$

$$= \frac{k\frac{x - x}{kx - xk}k}{c(\frac{x - x}{kx - xk}; X)}(\frac{x - x}{kx - xk}) + x \tag{14}$$

$$= \frac{x - x}{c(x; X)} + x \tag{15}$$

When considering  $x$  as an infeasible point, we need scale down  $\frac{1}{c(x; X)}$  such that the  $\frac{1}{c(x; X)}(x - x) + x$  will be located in the boundary. Therefore, we take  $\frac{1}{c(x; X)}$ , the homeomorphic projection operator is indeed the bisection projection operator (2). It is also equivalent to the gauge projection or RAYEN methods by its definition in Def. A.3.

□

Thus, the bisection projection framework provides a unified view for some existing projection-analogous approaches over convex sets. Meanwhile, we highlight the theoretical analysis and application scenario for BP works on general compact sets under Assumption 1 beyond those in the existing studies, further exploring the projection-based design and achieving substantially better performance in feasibility, optimality loss, and speedup as shown in Sec. 6.

### A.3. Comparison of Different Centers

Defining the centers of a set is a classic problem in mathematics, which involves various definitions tailored to serve specific purposes. Each definition captures a unique aspect of “centrality” depending on the application or theoretical requirements. Here, as shown in Table 3, we discuss several classic definitions including the proposed minimum-eccentricity interior point (MEIP) in our work.

Table 3. Comparison of different definitions of center for a set

Name	Definition	Description
MEIP	$x = \arg \min_{x \in X} \max_{y \in @X} kx - yk \quad \min_{y \in @X} kx - yk$	Minimizes the discrepancy between the maximum and minimum IP-to-boundary distances.
Chebyshev Center	$x = \arg \max_{x \in X} \min_{y \in @X} kx - yk$	Maximizes the minimum IP-to-boundary.
Circumcircle Center	$x = \arg \min_{x \in X} \max_{y \in @X} kx - yk$	Minimizes the maximum IP-to-boundary.
Analytical Center	$x = \arg \max_{x \in X} \prod_{i=1}^{n_{\text{ineq}}} \log(g_i(x))$	Maximizes the logarithmic barrier of the inequality residuals $g_i(x) > 0$ .
Max-residual Center	$\max_{x, t} t \quad \text{s.t.} \quad g_i(x) + t > 0; \quad i = 1, \dots, n_{\text{ineq}}$	Maximize the constraint residual to find a “central” IP.
Centroid	$x = \frac{1}{P} \sum_{i=1}^n x_i$	Calculates the average position of all points in the set.
Barycenter	$x = \frac{1}{P} \sum_{i=1}^n w_i x_i = \sum_{i=1}^n w_i$	Calculates the weighted average position of all points in the set, where each point has an associated weight.

- Geometric: The MEIP, Chebyshev Center, and Circumcircle Center focus on geometric properties of sets, specifically distances to the boundary. We propose the MEIP for bisection operation, justified by the performance guarantee in Prop. 4.1.

- Barrier : The Analytical Center and Max-residual Center use optimization techniques (Lee14; THH23) to find a “central” IP within a feasible region. This approach is crucial for solving linear and nonlinear programming problems. However, maximizing the log-residual or residual directly of the inequality function does not directly reflect the point-to-boundary distance for general constraint sets, which may result in a large deviation for the bisection operation.
- Statistical: The Centroid and Barycenter represent statistical approaches to defining centrality by calculating averages of point sets. The Centroid computes a simple arithmetic mean, suitable for applications in statistics and machine learning. The Barycenter incorporates weights, allowing for differentiated influence among points. Despite their simplicity, these centers are not necessarily interior points for general non-convex sets, making them unsuitable for our bisection operation.

#### A.4. Tackling Equality Constraint

Consider the following constraint set defined by both inequality and equality constraints:

$$C = \{x \in \mathbb{R}^n \mid h(x) = 0; g(x) \leq 0\} \tag{16}$$

where the functions  $h(\cdot) : \mathbb{R}^{n+d} \rightarrow \mathbb{R}^{n_{eq}}$  and  $g(\cdot) : \mathbb{R}^{n+d} \rightarrow \mathbb{R}^{n_{ineq}}$  are continuous with respect to  $x$  and  $\cdot$ . For simplicity, we use  $h(\cdot) = h(\cdot; \cdot)$ .

Assuming the equality constraint maintains a constant rank:

$$\text{rank}(J_h(x)) = r; \quad \forall x \in C \tag{17}$$

This condition implies that  $C$  has a Euclidean dimension of  $n - r$ , as per the Constant-Rank Level Set Theorem (Lee13).

In simpler terms, we can utilize a subset of decision variables  $x_1 \in \mathbb{R}^{n-r}$  and reconstruct the complete set of decision variables  $[x_1; x_2] \in \mathbb{R}^n$  by solving  $x_2 = f(x_1)$ , such that  $([x_1; f(x_1)]) \in C$ . Note that such a parametrization are not necessarily held globally for non-linear equality constraints. This method of reconstruction, which ensures the feasibility of the equality constraint, is extensively used in optimization literature (Ab69; PZC19; ZB20; DRK20; LCL23; THH23; LM23; DWDS23).

We then denote the reduced constraint set as

$$C^s = \{x \in \mathbb{R}^{n-r} \mid g([x_1; f(x_1)]; \cdot) \leq 0\} \tag{18}$$

This set  $C^s$  is not only equivalent to the original constraint set but also homeomorphic to it, implying a one-to-one, continuous, and bicontinuous correspondence between the two sets. The forward and inverse mappings of this homeomorphism are described by the following transformations:

$$[x_1; x_2] \in C \iff x_1 \in C^s; \tag{19}$$

$$x_1 \in C^s \iff [x_1; f(x_1)] \in C; \tag{20}$$

Let's consider two examples to illustrate this equality completion/reconstruction process:

##### Linear equality constraint

Let's consider an equality constraint defined as  $\mathbb{R}^n \mid Ax = b; A \in \mathbb{R}^{r \times n}; b \in \mathbb{R}^r, g$ , where  $x$  is the decision variable and  $b$  is the input parameter. We can assume, without loss of generality, that the rank of matrix  $A$  is  $\text{rank}(A) = r$ .

To facilitate the reconstruction process, we partition the decision variable into two groups:  $x_1 \in \mathbb{R}^{n-r}$  and  $x_2 \in \mathbb{R}^r$ . Accordingly, we also partition matrix  $A$  into  $A = [A_1; A_2]$ , where  $A_1 \in \mathbb{R}^{r \times (n-r)}$  and  $A_2 \in \mathbb{R}^{r \times r}$ . Hence, the equality constraint can be represented as  $A_1 x_1 + A_2 x_2 = b$ . The reconstruction process indicates that we can determine  $x_2$  using only the subset of variables  $x_1$ , with the explicit relationship given by:

$$x_2 = b - A_1 x_1 = A_2^{-1} (b - A_1 x_1); \tag{21}$$

<sup>5</sup>If an open set  $X$  is Euclidean of dimension, then every point  $x \in X$  has a neighborhood that is homeomorphic to an open subset of  $\mathbb{R}^n$  (Lee13).

Here, we choose the partition of  $x_2$  such that  $A_2$  has the full rank of  $r$ .

The relevant Jacobian matrix for back-propagation in this context is:

$$J(x_1) = A_2^{-1} A_1 \quad (22)$$

### Non-linear equality constraint

For a non-linear equality constraint defined as  $h(x) = 0$ ;  $x \in \mathbb{R}^d$ ;  $h: \mathbb{R}^{n+d} \rightarrow \mathbb{R}^r$ , we partition the decision variable into  $x_1 \in \mathbb{R}^{n-r}$  and  $x_2 \in \mathbb{R}^r$  in a similar fashion to the linear case. Under the assumption that the Jacobian matrix of  $h$  with respect to  $x_2$  has a constant rank, the completion function is well-defined and satisfies:

$$h([x_1; \phi(x_1)]) = 0 \quad (23)$$

To solve for  $\phi(x_1)$  when  $h$  is non-linear, we can employ an iterative technique such as Newton's method. The necessary Jacobian matrix for back-propagation can be computed using the Implicit Function Theorem, which provides the derivative of the implicitly defined function  $\phi$ . The Jacobian matrix is given by:

$$J(x_1) = J_h^{-1}(x_2) J_h(x_1) \quad (24)$$

Note that  $\phi$  for such a non-linear constraint may not be single-valued globally and depends on the initial value for the iterative algorithm, which may bring potential convergence issues.

In conclusion, reconstruction techniques utilizing equality constraints allow for a reduction in the dimensionality of the decision variable space. By modeling only a subset of the decision variables, we can focus on the inequality constraints and use the equality constraints to define the remaining variables implicitly. This process is differentiable, making it suitable for integration into the training of machine learning models, hence providing a powerful tool for incorporating equality constraints into such models [69, PZC19, PCZL22, DRK20, DWDS23]. For the implementation issues, we follow the established procedures in previous works [20, LCL23], where the partial variables for convex problems are randomly sampled, and the ones for AC-OPF problems are strategically selected.

### A.5. Extension BP to Multiple Interior Points

The bisection method can be executed in batch for multiple interior points  $x_{:,k} := f(x_{:,k}) \in \mathbb{C}$ , and we select the projected point as the one with minimum deviation, defined as:

$$\hat{x}_{:,k} = \text{BP}(x_{:,k}; X_{:,m}), \quad \arg \min_{x_{:,k}} \|x_{:,k} - \hat{x}_{:,k}\| \quad (25)$$

where  $\hat{x}_{:,k} = \text{BP}(x_{:,k}; X_{:,m})$  is the returned feasible point by bisection w.r.t. the IP  $x_{:,k} \in X_{:,m}$ .

Similarly, we can define the eccentricity of a set of IPs, crucial for bounding the bisection-induced projection distance.

**Definition A.5 (Eccentricity of IPs)** For a compact set  $X$  satisfying Assumption 1 with non-empty interior, the eccentricity of a set of IPs  $X_m := \{x_{:,k}\}_{k=1}^m \subset X$  with respect to a compact subset of boundary  $\partial X$  is defined as:

$$E(X_m; \partial X) = \max_{y \in \partial X} \frac{\|y - X_m\|}{\min_{x \in X_m} \|y - x\|} \quad (26)$$

where  $\|y - X_m\| = \min_{x \in X_m} \|y - x\|$  is the point-to-set distance.

Next, we establish the connection between eccentricity and the bisection-induced projection distance.

**Proposition A.2.** Let  $x = F(\cdot)$  be an infeasible NN prediction with bounded prediction error  $\|x - x_{pre}\| \leq \epsilon$ ;  $\hat{x} = \text{BP}(x; X_{:,m})$  be the projected solution with interior points  $X_{:,m} \subset \mathbb{C}$ ; Then, the worst-case projection distance is upper bounded as:

$$\max_{x \in B(x; \epsilon)} \|x - \hat{x}\| \leq \|x - \text{BP}(x; X_{:,m})\| + E(X_{:,m}; \partial X); \quad (27)$$

where  $B(x; \epsilon)$  represents the NN prediction region, enclosing all infeasible NN predictions with prediction error  $\|x - x_{pre}\| \leq \epsilon$  and  $\hat{x} = \text{BP}(x; X_{:,m})$ ;  $\partial X \cap B(x; \epsilon) \cap \mathbb{C}$  defines a subset of the constraint boundary containing all projected NN solutions from the NN infeasibility region.

The proof is similar to the single IP case and presented in the next section.

Figure 9.A geometric illustration of eccentricity and the proof.

## B. Proof for Main Results

### B.1. Proof for Bisection-induced Projection Distance (Prop. 4.1)

Proof. Without loss of generality, we assume that the optimal solution  $x^*$  lies on the boundary of the constraint set, implying the existence of active constraints at optimality. This is a standard assumption in constrained optimization theory (NW99).

For an infeasible solution  $x \notin B(x; r_{pre})$ , recall the definition of the projected solution as:

$$x^{\dagger}, BP(x; x) = (x - x^*) + x^* \tag{28}$$

where  $\alpha \in [0, 1]$  leads to  $x^{\dagger} \in \mathcal{C}$ .

Prediction-agnostic bound

Then, the projection distance for an infeasible prediction is bounded as:

$$\|x - x^{\dagger}\| \stackrel{(a)}{=} \|x - x^* + x^* - x^{\dagger}\| \leq \|x - x^*\| + \|x^* - x^{\dagger}\| \tag{29}$$

$$\|x^* - x^{\dagger}\| \stackrel{(b)}{\leq} \|x^* - x + x - x^{\dagger}\| \leq \|x^* - x\| + \|x - x^{\dagger}\| \tag{30}$$

$$\|x^* - x^{\dagger}\| \stackrel{(c)}{\leq} r_{pre} + \|x - x^{\dagger}\| \tag{31}$$

$$\|x^* - x^{\dagger}\| \stackrel{(d)}{\leq} r_{pre} + \max_{x \in \mathcal{C}} \|x - x^*\| - \min_{x \in \mathcal{C}} \|x - x^*\| \tag{32}$$

$$\stackrel{(e)}{=} r_{pre} + E(x; \mathcal{C}) \tag{33}$$

$$r_{pre} + E(x; \mathcal{C}) \tag{34}$$

Equality (a) is by three points  $x, x^*$ , and  $x^{\dagger}$ , exist in the same straight line. Inequality (b) is by the triangle inequality with auxiliary point  $x$ . Inequality (c) is by  $x^{\dagger} \in B(x; r_{pre})$ . Inequality (d) is by taking the maximum and minimum point over local boundary  $\mathcal{C} = f BP(x; x); \delta x \in B(x; r_{pre}) \cap \mathcal{C}$ . Equality (e) is by the definition of eccentricity in Def. 4.1.

Prediction-aware bound Based on (31), the projection distance for an infeasible prediction is bounded as:

$$\|x - x^{\dagger}\| \leq r_{pre} + \|x - x^{\dagger}\| \tag{35}$$

$$\|x - x^{\dagger}\| \leq r_{pre} + \min_{x \in \mathcal{C}} \|x - x^*\| \tag{36}$$

where  $\|x - x^{\dagger}\| = \|x - x^*\|$  denotes the distance between the IP prediction and the optimal solution,  $\min_{x \in \mathcal{C}} \|x - x^*\|$  denotes minimum point-to-boundary distance.

Proof for Proposition A.2:

For the multiple IPs setting  $x_{;m} = f_{x_{;k}} g_{k=1}^m C$ , we derive the upper bound as follows:

$$\min_{1 \leq k \leq m} \|x_{;k} - x_{;k}^*\| \stackrel{(a)}{\leq} \min_{1 \leq k \leq m} f_{pre} + \|x_{;k} - x_{;k}^*\| \tag{37}$$

$$\stackrel{(b)}{\leq} f_{pre} + \min_{1 \leq k \leq m} \|x_{;k} - x_{;k}^*\| \tag{38}$$

$$\stackrel{(c)}{\leq} f_{pre} + \max_{x^2} \min_{1 \leq k \leq m} \|x_{;k} - x_{;k}^*\| \tag{39}$$

$$\stackrel{(d)}{=} f_{pre} + E(X_{;m}; \cdot) \tag{40}$$

Inequality(a) is by the bound for the single-IP setting above. Inequality(b) is by the minimization of the joint term, which is smaller than the minimization separately. Inequality(c) is by taking the maximum and minimum point over the local boundary  $\mathcal{C} \setminus f_{BP}(x; X_{;m}); \delta \times 2 B(x; pre) \cap \mathcal{C} \cap g$

Thus, we complete the proof as follows:

$$\max_{x \in 2B(x; pre)} \min_{1 \leq k \leq m} \|x_{;k} - x_{;k}^*\| \leq f_{pre} + E(X_{;m}; \cdot) \tag{41}$$

□

### B.2. Proof for Feasibility Guarantee in (Prop. 5.1)

Proof. Since  $D$  is an  $r$ -covering dataset for  $\mathcal{S} = \bigcup_{i=1}^N B(x_i; r)$ , for any  $x \in \mathcal{S}$ , there exists at least one  $x_i \in D$  such that:

$$\|x - x_i\| \leq r \tag{42}$$

Given that  $\mathcal{C}$  is  $L$ -Lipschitz continuous over  $\mathcal{S}$ , we have:

$$\| \mathcal{C}(x) - \mathcal{C}(x_i) \| \leq L \|x - x_i\| \leq L r \tag{43}$$

Since constraint violation function  $G$  is  $L_{G,x}$ -Lipschitz in  $x$  and  $L_{G;}$ -Lipschitz in  $\cdot$ , we can bound the change  $\Delta G$  due to perturbations in both arguments:

$$G(x; \cdot) - G(x_i; x_i) \leq L_{G,x} \|x - x_i\| + L_{G;} \| \cdot - x_i \|$$

Suppose for each  $x_i \in D$ , the training requirement is satisfied as  $G(x_i; x_i) + L_{G,x} L r + L_{G;} r \leq 0$ , then we have for any  $x \in \mathcal{S}$ , the constraint violation can be bounded as  $G(x; \cdot) \leq 0$ .

□

### B.3. Proof for Optimality Gap and Run-time Complexity (Theorem 1)

Proof. First, the feasibility of the solution returned through bisection is guaranteed due to the bisection trajectory connecting an infeasible point and an interior point, which must intersect the constraint boundary. Thus, the bisection algorithm can always find a feasible solution by scaling down the infeasible solution along the line segment. We remark that for general non-convex sets, the line segment between an infeasible point and an interior point may intersect the constraint boundary multiple times, causing our bisection algorithm to converge to one of the multiple feasible solutions.

Let  $x^* \in \mathcal{C}$  be the converged boundary feasible solution given in finite bisection with an interior point  $x^K$

We divide the optimality gap by the following three terms:

$$\|x^K - x^*\| \leq \underbrace{\|x^K - x^*\|}_{\text{prediction error}} + \underbrace{\|x^* - x^*\|}_{\text{projection error}} + \underbrace{\|x^* - x^K\|}_{\text{bisection error}} \tag{44}$$

The prediction error is determined by the provided NN predictor, and we denote it as  $\phi_k(x; \hat{x})$ , where  $\hat{x}$  is the NN predictor to predict the optimal solution.

The projection error from bisection-projection, as proved in Proposition 4.1, can be bounded by the eccentricity-related term as:

$$\|x^* - \hat{x}\| \leq \max_{y \in B(x; r_{pre})} \|y - x^*\| \quad (45)$$

$$r_{pre} + E(x; \mathcal{C}) \quad (46)$$

Since  $x^*$  is the converged boundary feasible solution under the bisection algorithm, the bisection error induced by  $K$  iteration can be derived as:

$$\|x^* - \hat{x}^K\| \leq \|x^* - x\| + \|x - \hat{x}^K\| \quad (47)$$

$$= \|x^* - x\| + \|x - \hat{x}^K\| \quad (48)$$

$$\leq 2^{-K} \|x^* - x\| + \|x - \hat{x}^K\| \quad (49)$$

$$\leq 2^{-K} (\|x^* - x\| + \|x - \hat{x}^K\|) \quad (50)$$

$$\leq 2^{-K} (r_{pre} + D) \quad (51)$$

where  $D = \text{diam}(\mathcal{C})$  denote the diameter of a compact set.

Combining the three terms together, we have:

$$\|x^* - \hat{x}^K\| \leq 2^{-K} (r_{pre} + E(x; \mathcal{C})) + 2^{-K} (r_{pre} + D) \quad (52)$$

The complexity of executing the bisection algorithm involves the iteration steps, the number of IPs, and the complexity of verifying the inequality constraints at each iteration. For example, if the inequality constraint  $g_i(x)$  is a linear function for all  $i = 1, \dots, n_{ineq}$ , then  $G = n_{ineq}$ . In contrast, iterative algorithms such as interior point methods have a complexity of  $O((n + n_{ineq})^3)$  at each iteration due to the matrix inversion operation.

□

## C. Training for IPNN

### C.1. Prediction-agnostic training

In scenarios where prior information about the optimal solution or a trained neural network (NN) predictor is unavailable, our objective is to minimize the worst-case projection distance induced by bisection under any NN predictor. Guided by Proposition 4.1, we aim to minimize the eccentricity relative to the constraint boundary, denoted as  $\mathcal{C}$ .

**Definition C.1 (Prediction-agnostic MEIP)** For a compact set  $\mathcal{C}$  with a non-empty interior, the minimum eccentricity IP is defined as the solution of the following problem:

$$\min_x E(x; \mathcal{C}), \quad \max_{y \in \mathcal{C}} \|y - x\| \quad \min_{y \in \mathcal{C}} \|y - x\| \quad (53)$$

To facilitate the minimization of the MEIP, we adopt the Chebyshev Center as a relaxation. The Chebyshev Center provides a robust central point within the constraint set by maximizing the minimal distance from the center to the boundary.

**Definition C.2 (Chebyshev center)** For a compact set  $\mathcal{C}$  with a non-empty interior, the Chebyshev Center is defined as:

$$\max_x r; \quad \text{st: } B(x; r) \subset \mathcal{C} \quad (54)$$

Building upon this formulation, we derive the Prediction-Agnostic loss function, which aims to encourage the NN predictor  $\hat{x}$  to lie within the feasible set while maximizing the robust margin

$$L(\hat{x}; \mathcal{C}) = E_u [P(x + u; \mathcal{C})] - \log(r) \quad (55)$$

where  $P(x; \epsilon)$  represents the penalty function for constraint violations,  $u$  is sampled from unit ball to perturb the center  $x$ ,  $\epsilon$  is a variable representing the robust margin to be maximized, and positive coefficient to balance different loss terms. The first term ensures that perturbed points within the ball  $B(x; \epsilon)$  satisfy the constraints, while the second term encourages maximizing the margin

### C.2. Prediction-aware training

In contrast, when a trained NN predictor  $F(\cdot)$  or a dataset of optimal solutions  $(s, x^*)$  is available, the objective shifts to identifying interior points with minimized eccentricity within a local region of the constraint boundary. Given the bound on the bisection-induced projection distance from Equation (36):

$$\|x - x^*\| \leq \|x - F(x)\| + \|F(x) - x^*\| \leq \min_{x \in \mathcal{C}} \|x - x^*\| \quad (56)$$

We define the following Prediction-Aware MEIP:

Definition C.3 (Prediction-aware MEIP)

$$\min_x \|x - x^*\| \quad (57)$$

$$\text{st: } \begin{aligned} & \max_x \\ & \text{st: } B(x; \epsilon) \subset \mathcal{C} \end{aligned} \quad (58)$$

To effectively minimize the bisection-induced projection distance when the optimal solution dataset or NN predictor  $F(\cdot)$  is available, we design the following Prediction-Aware loss function:

$$L(\cdot; \epsilon) = E_u [P(x + u; \epsilon)] + \log(\cdot) + \alpha \|x - x^*\| \quad (59)$$

- The first two terms are analogous to the Prediction-Agnostic loss, promoting feasibility and robustness.
- The third term penalizes the deviation of the IP from the known optimal solution or initial NN prediction  $F(\cdot)$ , enhancing optimality.
- $\alpha$  is a positive coefficient that balances the trade-off between maximizing the robust margin and minimizing the distance to the optimal solution.
- Further, to reduce training complexity, we can initialize the IPNN with the one trained NN solution predictor, then follow the loss to re-tune the IPNN to find interior points.

Notably, when  $\alpha = 0$ , the loss function reduces to regular supervised training with a constraint violation penalty, as commonly used in NN-based constrained optimization solvers (DRK20). By maximizing the robust margin, the training process follows the preventive/adversarial approach (ZPC(20)), seeking an NN predictor that maintains feasibility by keeping distance from constraint boundaries. Rather than directly using such a feasible IP with its larger optimality gap, we apply bisection between the IP and an infeasible but near-optimal NN solution, effectively balancing the feasibility-optimality trade-off.

## D. Data and Experiment Setting

### D.1. Formulation for Optimization Problems

We test the Bisection Projection framework for four constrained optimization problems, including two convex optimization problems and two real-world non-convex problems. We follow the established procedures from available codes in previous works (DRK20; LCL23), where parameter configuration and sampling strategy are publicly available.

#### D.1.1. CONVEX PROBLEM FORMULATION

The Quadratic Program (QP) is a fundamental optimization problem where the objective function is quadratic and the constraints are linear. The QP problem can be formulated as:

$$\text{QP: } \min_{x \in \mathbb{R}^n} \frac{1}{2} x^T Q x + p^T x \quad (60)$$

$$\text{subject to } Gx \leq h; \tag{61}$$

$$Ax = b; \tag{62}$$

where:  $Q \in \mathbb{S}_{++}^n$  is a positive definite matrix, ensuring the convexity of the objective function,  $p \in \mathbb{R}^n$  is a vector of linear coefficients,  $A \in \mathbb{R}^{n_{eq} \times n}$  is a matrix defining equality constraints,  $G \in \mathbb{S}^n$  is a matrix defining inequality constraints,  $h \in \mathbb{R}^{n_{ineq}}$  is a vector specifying the upper bounds for the inequality constraints, and  $b \in \mathbb{R}^{n_{eq}}$  is a vector specifying the right-hand side of the equality constraints.

The Convex QCQP extends the QP by including quadratic constraints. The Convex QCQP problem is given by:

$$\text{Convex QCQP: } \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2}x^T Qx + p^T x \tag{63}$$

$$\text{subject to } x^T H_i x + g_i^T x \leq h_i; \quad i = 1; \dots; n_{ineq}; \tag{64}$$

$$Ax = b; \tag{65}$$

where each  $H_i \in \mathbb{S}_{++}^n$  is a positive definite matrix corresponding to the  $i$ th quadratic constraint,  $g_i \in \mathbb{R}^n$  is a vector of linear coefficients for the quadratic constraints, and  $h_i \in \mathbb{R}$  represents the upper bound for the  $i$ th quadratic constraint.

The SOCP is a convex optimization problem that generalizes linear and quadratic programs by allowing conic constraints. A SOCP problem is formulated as follows:

$$\text{SOCP: } \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2}x^T Qx + p^T x \tag{66}$$

$$\text{subject to } \|G_i x + h_i\|_2 \leq c_i^T x + d_i; \quad i = 1; \dots; n_{ineq}; \tag{67}$$

$$Ax = b; \tag{68}$$

where  $G_i \in \mathbb{R}^{m \times n}$  and  $h_i \in \mathbb{R}^m$  define the second-order cone,  $c_i \in \mathbb{R}^n$  and  $d_i \in \mathbb{R}$  are the coefficients and scalar terms of the conic constraints, respectively.

The Semidefinite Program (SDP) is an optimization problem where the goal is to minimize a linear objective function subject to semidefinite constraints. The standard formulation of an SDP is given by:

$$\text{SDP: } \underset{X \in \mathbb{S}^n}{\text{minimize}} \quad \text{tr}(CX) \tag{69}$$

$$\text{subject to } X \succeq 0; \tag{70}$$

$$\text{tr}(A_i X) = b_i; \quad i = 1; \dots; n_{eq}; \tag{71}$$

where  $X \in \mathbb{S}^n$  is the symmetric matrix variable,  $C \in \mathbb{S}^n$  is a given symmetric matrix of coefficients for the objective function,  $\text{tr}(\cdot)$  is the trace of a matrix,  $A_i \in \mathbb{S}^n$  are given symmetric matrices that define the equality constraints,  $b_i \in \mathbb{R}$  are the given scalars that specify the right-hand side of the equality constraints, and  $n_{eq}$  is the number of equality constraints. We discuss the penalty design for PSD constraint in Appendix D.3.

### D.1.2. JOINT CHANCE CONSTRAINED INVENTORY MANAGEMENT (JCC-IM)

We consider the Joint Chance-Constrained Inventory Management (JCC-IM) problem, which seeks to optimize inventory levels across multiple warehouses under conditions of demand uncertainty, ensuring a high probability of meeting that demand. The JCC-IM problem is formally defined as:

$$\text{JCC-IM : } \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad c^T x \tag{72}$$

$$\text{subject to } \text{Prob}(Ax \leq b) \geq 1 \tag{73}$$

$$Gx \leq h; \quad x^{\min} \leq x \leq x^{\max}; \tag{74}$$

where  $n$  denotes the number of warehouses located in distinct regions, the decision variable  $x$  represents the inventory order quantity to be determined in advance for each warehouse, in order to satisfy future demand. The vector  $c$  encapsulates the historical average demand, and the term  $p(\cdot)$  models the stochastic deviations from this average, capturing the inherent uncertainty of demand. The matrix  $A$  characterizes the interdependencies among different warehouses, which may arise

Figure 10. This figure visualizes a sample-based individual chance constraint defined as  $\text{Prob}(a_1x_1 + a_2x_2 \leq 1) \geq 90\%$ , where  $a_1, a_2$  are independent Gaussian variables. The probability of satisfying this constraint is estimated from samples and the indicator function  $I(\cdot)$  as:  $\frac{1}{N} \sum_{i=1}^N I(a_1^i x_1 + a_2^i x_2 \leq 1) \geq 90\%$ . The visualizations underscore the non-smooth geometry and optimization difficulty (PAS09). We remark that BP was tested in a high-dimensional scenario with 400 decision variables and joint constraints in our experiments.

from shared types of inventory or geographical proximity. The parameter  $\epsilon$  specifies the acceptable risk level, thus ensuring that the probability of meeting demand across all warehouses is at least  $1 - \epsilon$ . The additional constraints  $Gx \leq h$  and  $x^{\min} \leq x \leq x^{\max}$ , represent warehouse-specific capacity limitations and inventory bounds, respectively.

The Joint chance constraint (JCC) represents the probability of the joint event of feasibility for each constraint as:

$$\text{Prob}(Ax \leq b + \xi) = \text{Prob}(A_1x \leq b_1 + \xi_1; \dots; A_mx \leq b_m + \xi_m) \geq 1 - \epsilon \quad (75)$$

When  $m = 1$  and with Gaussian uncertainty, the probability constraint can be reformulated into a second-order cone constraint (BV04). However, in the general case, given the absence of an analytical reformulation for the JCC, we employ a Sample-Average (SA) approach to approximate the chance-constrained problem. This technique involves generating a finite set of scenarios  $\xi_j, j=1, \dots, N$  from the underlying distribution of  $\xi$ . The SA variant of the JCC-IM is formulated as:

$$\text{SA-JCC-IM} : \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad c^T x \quad (76)$$

$$\text{subject to} \quad P_N = \frac{1}{N} \sum_{j=1}^N I(Ax \leq b + \xi_j) \geq 1 - \epsilon \quad (77)$$

$$Gx \leq h; \quad x^{\min} \leq x \leq x^{\max} \quad (78)$$

where  $I(\cdot)$  is the indicator function. A solution is deemed to have a probabilistic JCC feasibility guarantee if  $P_N \geq 1 - \epsilon$ . This empirical evaluation provides a practical measure of the reliability of the SA-based solution in adhering to the demand satisfaction requirements stipulated by the JCC-IM problem.

In practice, the problem can be solved as mixed-integer programming but is intractable for high-dimension problems with a large number of scenarios; therefore, for iterative-solver-based baselines, we solve the robust version of this problem by setting  $\epsilon = 0$ , such that the problem becomes convex with a large number of constraints. But for BP methods, we can still project the solution to the chance constraint in (77) through bisection with easy feasibility checking shown in Alg. 1.

#### D.1.3. ALTERNATING CURRENT OPTIMAL POWER FLOW (AC-OPF)

The Alternating Current Optimal Power Flow (AC-OPF) problem is pivotal in ensuring the efficient and safe operation of power grids. It requires real-time decision-making and adherence to operational constraints to maintain system integrity. The

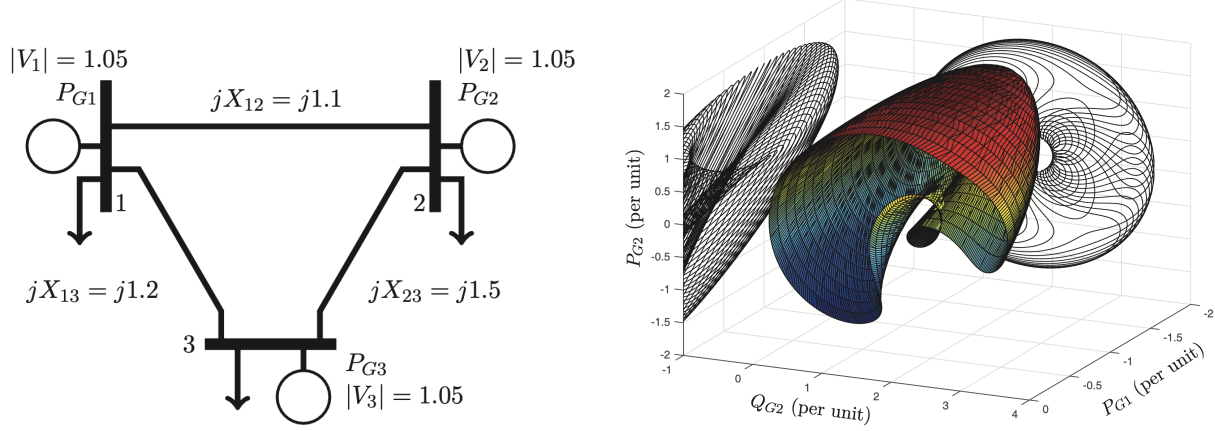


Figure 11. The **left** figure displays a simple 3-node power network, while the **right** figure illustrates a part of its constraint set ( $MH^+ 19$ ). These visuals highlight the complex geometry and inherent challenges of the ACOPF problem. It is noteworthy that in our experiments, BP was tested on a 200-node power network involving more than 1,000 constraints.

AC-OPF is inherently a non-convex Quadratically Constrained Quadratic Program (QCQP) and is recognized as NP-hard, posing significant computational challenges. The formal mathematical formulation of the AC-OPF problem is as follows:

$$\text{AC-OPF} : \min_{p_g, q_g, v} p_g^T Q p_g + b^T p_g \quad (79)$$

$$\text{subject to } jv_i(v_i - v_j)w_{ij} \in S_{ij}^{\max}; \quad \delta(i; j) \in E; \quad (80)$$

$$(p_g \quad p_d) + (q_g \quad q_d) \mathbf{i} = \text{diag}(v) W v; \quad \delta \mathbf{i} \in N; \quad (81)$$

$$p_g^{\min} \leq p_g \leq p_g^{\max}; \quad q_g^{\min} \leq q_g \leq q_g^{\max}; \quad v^{\min} \leq |v_j| \leq v^{\max}; \quad (82)$$

where the power network comprises  $n$  nodes, indexed by the set  $N$ . The vectors  $p_d, q_d \in \mathbb{R}^n$  represent the real and reactive power demand at each node, respectively. The vectors  $p_g, q_g \in \mathbb{R}^n$  denote the real and reactive power generation, which are the decision variables of the optimization problem. The vector  $v \in \mathbb{C}^n$  signifies the nodal voltage phasors. The admittance matrix  $W \in \mathbb{C}^{n \times n}$  characterizes the physical properties and topology of the power network, with  $W$  denoting its complex conjugate transpose. The generation cost is represented by a quadratic function with matrix  $Q \in \mathbb{R}^{n \times n}$  and vector  $b \in \mathbb{R}^n$ . The constraints include generation limits ( $p_g^{\min}; p_g^{\max}; q_g^{\min}; q_g^{\max}$ ), voltage magnitude bounds ( $v^{\min}; v^{\max}$ ), thermal line limits ( $S_{ij}^{\max}$ ), and power flow balance equations. The set  $E$  denotes the set of edges (transmission lines) connecting the nodes in the power network. The equality constraint represents the complex power flow balance at each node, ensuring that the generation and demand are matched while accounting for power losses.

## D.2. Experiment Setting

**Computational Infrastructure:** All NN-based methods are implemented in Pytorch and executed on an Ubuntu server with an NVIDIA A800 GPU. Iterative algorithms were executed in parallel on an AMD EPYC 7763 64-Core Processor. For convex optimization problems, we employed the MOSEK optimizer under an academic license. The Joint Chance-Constrained Inventory Management (JCC-IM) problem was approximated using sampled scenarios and solved with MOSEK. Alternating Current Optimal Power Flow (AC-OPF) problems were addressed using the open-source PyPower toolkit (ZMS11). Additional experimental configurations are detailed in the respective sections and table footnotes.

**Baseline Methods:** We compare our approach against the following baselines:

- **Optimizer:** For convex optimization problems, we employ MOSEK as the baseline solver. For AC-OPF problems, we use PyPower (ZMS11) as the specialized solver.
- **NN:** A vanilla neural network that directly maps input parameters to solutions without feasibility guarantees. It is trained with an optimal solution dataset in a supervised setting.

- **Proj**: Infeasible predictions from NN are corrected via orthogonal projections. The projection problem is formulated and solved by Optimizer.
- **WS**: Infeasible NN predictions serve as warm-start initializations for iterative solvers, which may accelerate the convergence.
- **D-Proj**: The differentiable projection method from DC3 (DRK20), which employs gradient descent to minimize the constraint violation for constraint satisfaction. The gradient is derived via the automatic differentiation mechanism in PyTorch.
- **H-Proj**: Homeomorphic projection applied to infeasible predictions (LCL23). The invertible neural networks (INN) are trained in advance for each constraint type.
- **B-Proj**: Our proposed bisection-based projection (Algorithm 1) applied to predicted interior points for feasibility recovery.

**Evaluation Metrics:** We evaluate all methods using the following criteria on 1,024 test instances:

- **Feasibility**: The percentage of solutions satisfying all equality and inequality constraints within a tolerance of  $10^{-5}$ .
- **Optimality**: The relative solution and objective optimality gap, defined as  $\frac{kx - x}{kx - k}$  and  $\frac{jf(x) - f(x)}{jf(x)}$ , respectively, where  $f(x)$  is the objective value of the predicted/projected solution and  $f(x)$  is the optimal objective value.
- **Runtime**: Wall-clock time for inference, including raw NN prediction and any post-processing steps.

### D.3. Hyper-parameters of NN and IPNN

We employ a fully connected neural network with residual connections (HZRS16; LLC24) and equality reconstruction (PZC19; DRK20), denoted as  $F$ , to predict optimal solutions or interior points for constrained optimization problems given input parameters  $\mathcal{Z}$ . The network architecture incorporates skip connections to facilitate gradient flow and a reconstruction module to enforce equality constraints.

**Data Generation for NN predictor:** Training and test datasets are generated by solving optimization instances across diverse parameter configurations using established solvers:

- **Convex problems**: MOSEK optimizer
- **AC-OPF problems**: PyPower (ZMSG97)
- **JCC-IM problems**: Sample Average Approximation (SAA) solved with MOSEK

**Loss Function for NN predictor:** The neural network is trained via supervised learning with a composite loss function that balances solution accuracy, constraint satisfaction, and objective quality:

$$L(F) = \mathbb{E}_D \left[ \frac{1}{2} \|kF(\cdot) - x\|_2^2 + \frac{1}{2} \sum_j \text{ReLU}(g_j(F(\cdot); \cdot)) + \frac{1}{2} f(F(\cdot); \cdot) \right] \quad (83)$$

where:

- The first term minimizes prediction error with respect to optimal solutions
- The second term penalizes inequality constraint violations, with  $g_j$  representing the  $j$ -th inequality constraint.
- The third term encourages objective function minimization to reduce the objective optimality gap directly.
- $\frac{1}{2}; \frac{1}{2} > 0$  are hyperparameters controlling the trade-off between feasibility and optimality

**Constraint Handling in NN and IPNN:**

- **Equality constraints:** we employ variable selection and completion (detailed in Appendix A.4), which guarantees exact satisfaction for equality constraints.
- **Penalty function:** For standard differentiable inequality constraints  $g(x) \leq 0$ , we compute the constraint violation directly as  $k\text{ReLU}(g(x))k$ , which naturally vanishes when constraints are satisfied.
- **Positive semidefinite constraints:** For matrix variables  $X \succeq 0$ , We offer two penalty computation approaches
  - We can compute the exact penalty using negative eigenvalues via  $\text{ReLU}(-\text{eig}(X))$ , where eigenvalues are obtained through `torch.linalg.eigvalsh`. While this method provides exact gradients through automatic differentiation, it may encounter numerical instability (e.g., singularity) during training.
  - We can estimate the penalty as  $\text{ReLU}(\sum_i v_i^T X v_i)$  using linear measurements  $\hat{r}_{v_i}^k$ , where the vectors  $v_i$  are computed iteratively following (NSW22). This approximation offers improved computational efficiency and numerical stability compared to the eigenvalue approach.

**NN and IPNN Training**

We train the NN predictor following a regular ML training scheme with parameters in Table 4. The Interior Point Neural Network (IPNN) shares the same base architecture as the standard NN predictor, with the same input and output dimensions. Furthermore, for the IPNN training for constrained optimization problems, we directly initialize it with parameters from a trained NN predictor to reduce its training time. Detailed architectural specifications and hyperparameters for both models are provided in Table 4.

Table 4. Structure of IPNN/NN predictor in experiments

Parameter	Value
NN/IPNN structure	
dimension of input layer	$d$
dimension of output layer	$n$
dimension of hidden layer	$b(d + n)/2c$
activation function	$\text{ReLU}()$
number of layer	3
last-layer activation	$\text{Sigmoid}()$
NN/IPNN training parameters	
number of training samples	10,000
number of testing samples	1,024
number of iteration	10,000
optimizer	AdamW
learning rate	0.0001
batch size	64
the coefficient for objective value	0.001
the coefficient for inequality penalty	0.01
the coefficient for robust margin	0.01

**E. Supplementary Experiment Results**

**E.1. IPNN Training and Bisection Projection over Various Constraint Sets**

We evaluate the effectiveness of our BP framework on two challenging non-convex constraint sets.

**Test Cases:** We consider two geometrically distinct non-convex sets:

$$\text{Case 1: } C_1(\cdot) = \{x \in \mathbb{R}^d \mid x^T Q_i x + q_i^T x + b_i \leq 0; i = 1; \dots; 6\} \tag{84}$$

$$\text{Case 2: } C_2(\cdot) = \bigcap_{i=1}^4 B(c_i; r_i); \text{ where } B(c; r) = \{x \mid \|x - c\|_2 \leq r\} \tag{85}$$

where  $\mathcal{C}_1 = \{f, Q_i; q_i; b_i; g_{i=1}^6\}$  and  $\mathcal{C}_2 = \{f, C_i; r_i; g_{i=1}^4\}$  parameterize the constraint sets. Case 1 represents a ball-homeomorphic set defined by quadratic constraints, exhibiting complex non-convex geometry. Case 2 consists of a union of disjoint balls, presenting the additional challenge of disconnectivity—a property that violates the assumptions of many projection-based methods. For each case, we:

- Train the IPNN to learn the chebyshev centers.
- Evaluate bisection projection on unseen test parameters
- Compare against Homeomorphic Projection (H-Proj) using identical test instances

Results and Discussion. Results are presented in Figures 12 - 17. We make the following key observations:

- The robust margin maximization scheme provides two key benefits: it increases the interior-to-boundary distance, improving IPNN prediction feasibility on unseen samples, and enables the trained IPNN to predict central interior points for projections.
- For ball-homeomorphic constraint sets (Case 1), bisection projection achieves comparable performance to homeomorphic projection. However, when constraints are disconnected (i.e., non-ball-homeomorphic in Case 2), the homeomorphic projection fails to identify a valid INN for projection, resulting in infeasibility. In contrast, our bisection projection consistently identifies central interior points and maintains solution feasibility.

### E.2. Sensitivity Analysis on $\gamma$ for Out-of-Sample Feasibility and Projection Distance

We empirically evaluate the impact of the robust margin parameter  $\gamma$  maximization on two critical performance metrics: (i) out-of-sample feasibility rates for interior point predictions on unseen parameter instances (Fig. 6), and (ii) incurred projection distances after bisection-based feasibility recovery (Fig. 7).

We compare three IPNN training strategies:

- **No** : Trained exclusively to minimize constraint violations without margin regularization ( $\gamma = 0$ ).
- **Fixed** : Trained with a constant margin parameter (e.g.,  $\gamma = 10^{-2}$ ), analogous to randomized smoothing techniques employed in adversarial training.
- **Train** : Trained with our proposed  $\gamma$  maximization regularization, initialized at  $\gamma = 10^{-2}$ .

To ensure statistical reliability, we trained each model configuration five times with different random seeds and report results with standard deviations.

### E.3. Sensitivity Analysis on Bisection Steps on Optimality Gap and Iteration Complexity

As established in Theorem 1, the bisection algorithm exhibits a linear convergence rate with low per-step computational cost. We empirically validate this theoretical property by systematically varying the number of bisection steps across multiple problem classes, as illustrated in Fig. 18.

The results confirm our theoretical analysis, demonstrating exponential reduction in optimality gap as the number of bisection steps increases, while computational time grows linearly. This favorable trade-off enables practitioners to select an appropriate number of iterations based on their specific accuracy requirements and computational constraints. Notably, most practical applications achieve acceptable convergence within 5-10 bisection steps, making our approach well-suited for real-time decision-making scenarios.

As shown in Fig. 2, when the line segment between an interior point and an infeasible prediction intersects the constraint boundary at multiple points, our bisection algorithm converges to one such intersection. The specific convergence point depends on the bisection parameter  $\epsilon$ , which controls the search granularity.

The bisection update rule can be generalized as:

$$m = l + (1 - \epsilon) \cdot u \tag{86}$$

where  $\alpha \in (0; 1)$  determines the bisection stepsize. The standard choice  $\alpha = 0.5$  yields binary search.

A smaller values (e.g.,  $\alpha = 0.1$ ) create a conservative search biased toward the infeasible prediction, and the algorithm tends to converge to boundary points closer to the original NN prediction, better preserving the learned solution structure. However, this comes at a computational cost: The number of bisection steps increases under the same convergence tolerance.

Figure 19 empirically demonstrates this trade-off, showing that practitioners can tune  $\alpha$  based on their specific requirements for solution quality versus computational budget.

Figure 12. **Non-convex Case:** Perturbed penalty loss (Left) and IPNN prediction feasibility rate (Middle Left) during training. Robust margin  $\log(\gamma)$  (Middle Right) and estimated eccentricity (Right) during training.

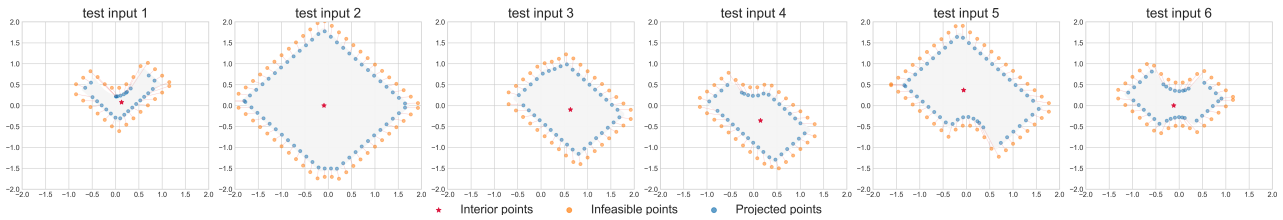


Figure 13. **Non-convex Case:** Bisection Projection with IPNN prediction given test input parameters.

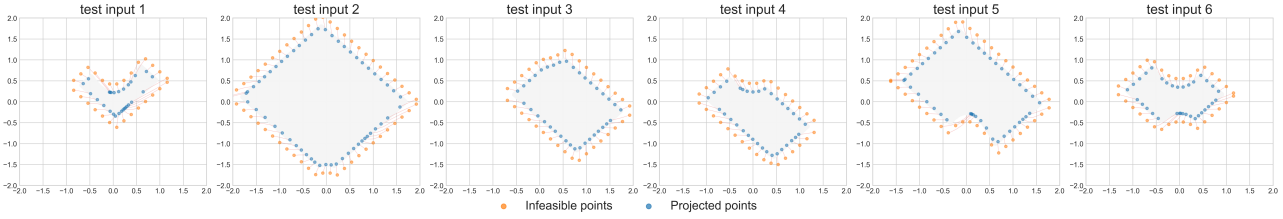


Figure 14. **Non-convex Case:** Homeomorphic Projection with trained INN given test input parameters.

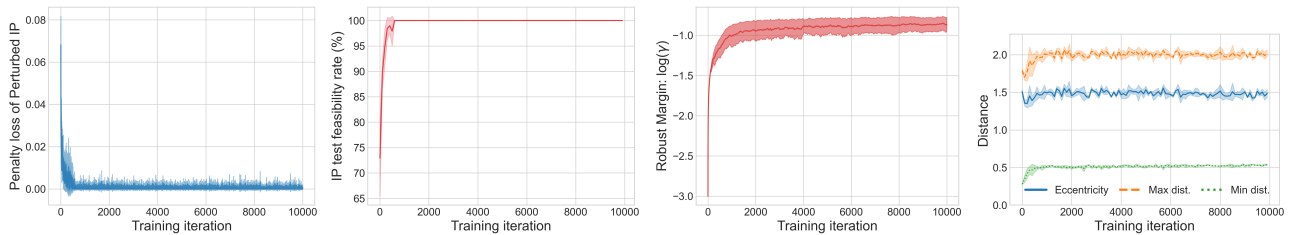


Figure 15. **Disconnected Case:** Perturbed penalty loss (Left) and IPNN prediction feasibility rate (Middle Left) during training. Robust margin  $\log(\gamma)$  (Middle Right) and estimated eccentricity (Right) during training.

