From Inverse Optimization to Feasibility to Empirical Risk Minimization

by

Saurabh Mishra

B.Tech., Indian Institute of Technology Roorkee, 2018

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Masters of Science

in the
Department of Computing Science
Faculty of Applied Science

SIMON FRASER UNIVERSITY
Summer 2024

Declaration of Committee

Name:	**Saurabh Mishra**
Degree:	**Masters of Science**
Thesis title:	**From Inverse Optimization to Feasibility to Empirical Risk Minimization**
Committee:	**Chair:** Manolis Savva Professor, Computing Science
	Sharan Vaswani Supervisor Assistant Professor, Computing Science
	Ke Li Committee Member Assistant Professor, Computing Science
	Tamon Stephen Examiner Professor, Department of Mathematics

Abstract

Inverse optimization involves inferring unknown parameters of an optimization problem from known solutions and is widely used in fields such as transportation, power systems, and healthcare. We study the *contextual inverse optimization* setting that utilizes additional contextual information to better predict the unknown problem parameters. We focus on contextual inverse linear programming (`CILP`), addressing the challenges posed by the non-differentiable nature of LPs. For a linear prediction model, we reduce `CILP` to a convex feasibility problem, allowing the use of standard algorithms such as alternating projections. The resulting algorithm for `CILP` is equipped with a linear convergence guarantee without additional assumptions such as degeneracy or interpolation. Next, we reduce `CILP` to empirical risk minimization (ERM) on a smooth, convex loss that satisfies the Polyak-Lojasiewicz condition. This reduction enables the use of scalable first-order optimization methods to solve large non-convex problems while maintaining theoretical guarantees in the convex setting. Subsequently, we use the reduction to ERM to quantify the generalization performance of the proposed algorithm on previously unseen instances. Finally, we experimentally validate our approach on synthetic and real-world problems, and demonstrate improved performance compared to existing methods.

Keywords: Decision Focused Learning, Inverse Optimization, Predict and Optimize

Preface

The main matter of this thesis is based on Mishra et al. [2024] that appeared in ICML 2024.

Saurabh Kumar Mishra, Anant Raj, Sharan Vaswani, "From Inverse Optimization to Feasibility to ERM", Forty-first International Conference on Machine Learning, 2024.

Saurabh Kumar Mishra is the major contributor on this paper in terms of the writing, theoretical results and experimental evaluation. The work was supervised by Sharan Vaswani who provided useful feedback and constantly helped in refining the paper's content. Anant Raj provided useful feedback and helped in refining the paper's content.

An earlier version of this work appeared in the Optimization for Machine Learning workshop at NeurIPS 2023:

Saurabh Kumar Mishra, Sharan Vaswani, "Reducing Predict and Optimize to Convex Feasibility", Optimization for Machine Learning Workshop NeurIPS, 2023.

Acknowledgements

I would like to express my deepest gratitude to my advisor, Dr. Sharan Vaswani, whose unwavering guidance and insightful feedback were instrumental in the completion of this research. He gave me the freedom to explore a new research area of inverse optimization and helped me navigate the challenges at each step. This work would not have been possible without his significant contributions.

Special thanks go to my co-author, Dr. Anant Raj, for his collaboration and dedication throughout this project.

I extend my sincere gratitude to the members of my examination committee: Dr. Ke Li, Dr. Tamon Stephen, and Dr. Manolis Savva, for their time, effort, and valuable insights.

Additionally, I am deeply grateful to Dr. Ke Li for giving me the opportunity to pursue research and for his invaluable feedback during my first year.

I am also indebted to my friends at SFU—Ashwin, Chirag, Ashish, Kumar, and Reza—for their insightful discussions on research and beyond, and for making my stay at SFU enjoyable.

Last but not least, I want to express my heartfelt thanks to my friends Tarannum, Karan, Mohit, Nitin, and Hardik, as well as to my family for their unconditional support throughout this project and beyond.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction and Related works

1.1 Introduction

Inverse optimization [Heuberger, 2004] is the reverse of standard optimization and uses a known output (decision) of an optimization problem to infer the unknown problem parameters. For example, in the context of linear programming (LP), inverse optimization uses the optimal solutions to the LP in order to learn the coefficients (costs) that can produce these solutions. Inverse optimization has found applications in transportation [Bertsimas et al., 2015], power systems [Birge et al., 2017] and healthcare [Chan et al., 2022] (refer to Chan et al. [2023] for a recent survey).

We focus on integrating additional contextual information into the inverse optimization framework. In particular, we leverage historical data and a machine learning (ML) model to predict the (unknown) optimization problem parameters that can render (known) optimal decisions. This setting is commonly referred to as contextual inverse optimization (CIO) [Besbes et al., 2023, Sun et al., 2023] or data-driven inverse optimization [Mohajerin Esfahani et al., 2018]. CIO requires a combination of prediction and optimization and has found applications in optimal transport and vehicle routing [Li et al., 2022], financial modeling [Cornuejols and Tütüncü, 2006], power systems [Bansal, 2005, Li et al., 2018], healthcare [Angalakudati et al., 2014], circuit design [Boyd et al., 2001], robotics [Raja and Pugazhenthi, 2012]. Some use-cases for CIO are as follows.

Example 1: Energy-cost aware scheduling [Wahdany et al., 2023], which involves using weather data to forecast wind-energy generation and hence energy prices (prediction). These predictions can be used to schedule jobs (optimization) to minimize energy costs. For the CIO, the contextual information corresponds to weather data, and the solutions (decisions) correspond to past schedules.

Example 2: Shortest path planning [Guyomarch, 2017], which involves predicting the time taken through different routes or terrain (prediction). These predictions can be used to determine the shortest path between two locations (optimization). For CIO, the contextual

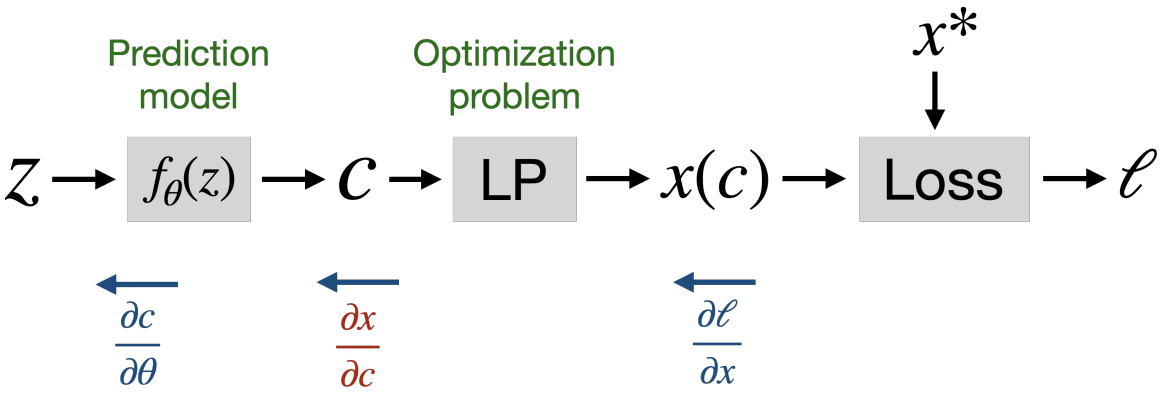

Figure 1.1: CIO framework: model f_θ takes input z and predicts the cost vector $c = f_\theta(z)$. This cost vector is the input of an optimization procedure that outputs decision $x(c)$. Given the optimal decision x^*, the objective is to learn the model parameters such that the predicted decision $x(c)$ is close to the optimal decision. To train the model in an end-to-end fashion, the key challenge is to compute the gradient of c w.r.t decision $x(c)$ (shown in red in the figure).

information corresponds to images or features of the terrain, and the decisions correspond to known shortest paths for pairs of locations. Refer to Fig. C.4 for a sample from the dataset.

Example 3: Inverse reinforcement learning (IRL) [Ng et al., 2000], which involves learning the underlying reward function in a Markov decision process (MDP) from the observed behaviour of a human expert. The learned reward function can be used to infer a good policy for an artificial agent. Assuming that the human expert acts in order to maximize the implicit reward functions, for the CIO, the context corresponds to features of the MDP, and decisions correspond to the observed expert behaviour.

Example 4: In rational choice theory, a common way to model agents (e.g. users interacting with a recommendation system) is to assume that the (i) agent is rational and is making decisions to optimize some unknown implicit utility function and that (ii) the form (but not the parameters) of this utility function is known (e.g. whether it is linear or concave). For recommendation systems, the user's demographics and other metadata correspond to the context. In CIO, this context is used to predict the unknown parameters of the utility function, such that when it is maximized, it can explain the users' past purchases (corresponding to the known decisions) [Wilder et al., 2019]. The estimated utility function can then be used to make better recommendations.

Since numerous combinatorial problems, including shortest path, max-flow, and perfect matching, can be cast as linear programs, we will mainly focus on cases where the optimization problem is a Linear Program (LP) (in Section 2.3.4, we briefly consider more general non-linear problems). The examples of CIO presented earlier fall within the LP framework. For LPs, the key challenge of contextual inverse linear programming (`CILP`) lies in the non-

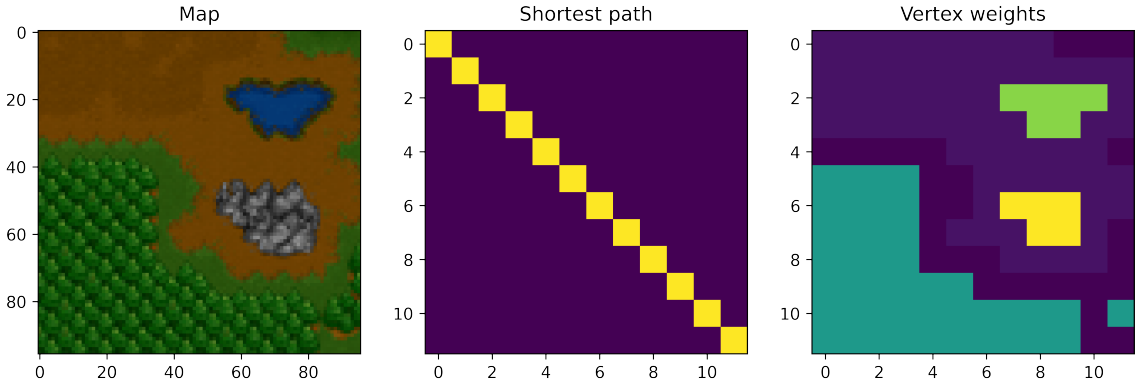

Figure 1.2: Warcraft SP dataset sample; Left: The input (refers to context z) image; Center: ground truth shortest path (refers to x^*); Right: the ground truth vertex weights (a valid c^* that retrieves the same x^*). The task is to learn the edge weights to retrieve the same shortest path.

differentiable nature of LPs. This limitation precludes the direct use of auto-differentiation techniques.

1.2 Thesis Contributions

To overcome the problem, we make the following contributions.

Contribution 1: For a linear prediction model, we reduce `CILP` to a convex feasibility problem (Section 2.3). This reduction enables the use of standard algorithms such as alternating projections. Unlike existing work [Sun et al., 2023], the resulting method (Algorithm 1) guarantees linear convergence to the solution without additional assumptions such as degeneracy or interpolation.

Contribution 2: To efficiently handle large-scale problems, we reduce the feasibility problem (and hence `CILP`) to empirical risk minimization (ERM) on a smooth, convex loss function satisfying the Polyak-Łojasiewicz condition [Polyak, 1964] (Chapter 3). This reduction allows us to employ scalable first-order optimization methods while retaining strong theoretical guarantees.

Contribution 3: In Section 4.1, we discuss the shortcomings of the previous measures of performance for `CILP`, and propose a new sub-optimality metric. Subsequently, we use the reduction to ERM to quantify the performance of the proposed algorithm on previously unseen instances.

Contribution 4: In Chapter 5, we validate the effectiveness of our approach with experiments on synthetic shortest path and fractional knapsack problems [Sun et al., 2023], and real-world Warcraft shortest path and MNIST perfect matching tasks [Vlastelica et al.,

2019]. Our empirical results demonstrate that the proposed algorithm results in improved performance compared to the prior work.

1.3 Related Work

In this section, we review the related works, contrasting them with our proposed approach.

Inverse Optimization: Iyengar and Kang [2005], uses the Karush–Kuhn–Tucker (KKT) optimality conditions for the LP to define the feasible set of cost vectors. Similarly, Mohajerin Esfahani et al. [2018], uses the Wasserstein metric to find a set of robust cost vectors by formulating an inverse optimization problem. However, they do not consider the contextual setting, so no learning is required. In contrast, we use the KKT conditions to train a prediction model, mapping contextual information to optimal decisions. More recently, Besbes et al. [2023] consider solving CIO in both the online and offline settings. Their offline setting is similar to our problem formulation, but does not make any linearity or convexity assumptions. They derive bounds on the worst-case suboptimality for a specific mapping from features to cost vectors. However, it is unclear whether this mapping can be efficiently computed even in the special case of LPs. Furthermore, Besbes et al. [2023] assume realizability i.e. there is no noise in the decisions and the model can perfectly fit (interpolate) the data. This further limits the practical utility of their framework. In contrast, we make no realizability assumptions and develop efficient algorithms for `CILP`.

Using the reduced cost optimality condition: Sun et al. [2023] proposed a method to use the reduced cost optimality conditions Luenberger et al. [1984] for LPs. The method constructs a surrogate loss function that encourages the prediction to satisfy the reduced cost optimality conditions. The resulting method has theoretical convergence guarantees, assuming that the LPs are non-degenerate and that the model can interpolate the training data. Both of these are strong assumptions and are not necessarily satisfied in practice. In contrast, we use the KKT conditions that are equivalent to the reduced cost optimality conditions for non-degenerate LPs (see Section B.2 for proof). Since the KKT conditions can also be used for degenerate LPs, our proposed framework provides theoretical guarantees without relying on this assumption. Moreover, our guarantees hold even without assuming interpolation.

Differentiating through LPs: Vlastelica et al. [2019] estimate the gradient "through" the LP by calculating the change in the decision by perturbing the prediction. However, it introduces additional hyper-parameters that are non-trivial to tune. Another common technique is to use the straight-through-estimator (ST) [Sahoo et al., 2022]. Given a set of predictions from the model, the ST method uses the LP to estimate the decisions. However, it does not consider the LP (treats the corresponding Jacobian as an identity matrix) while back-propagating the gradient from the decisions to the model parameters. Though successful in practice, this method is not theoretically principled. Moreover, the method in [Tan et al.,

2020] attempts to learn linear programs from optimal decisions. However, for the setting considered in this work, the gradient updates for this method are equivalent to that of the ST method. The method in Berthet et al. [2020] computes expected gradients by perturbing the prediction target in different directions. While this method accurately models the gradient, it is not practically feasible because of the computational cost of solving LPs multiple times for each update to the model. One advantage of these techniques is their "black-box" nature, meaning that they only rely on the outputs from an LP, thus allowing the use of faster problem-specific solvers. In contrast, our work leverages LP properties (and some other problems described in Section 2.3.4), allowing us to develop a more efficient and principled approach.

Implicit Differentiation: The methods in [Amos and Kolter, 2017, Amos, 2019] focuses on (strongly)-convex optimization problems. It calculates the gradient through such problems by differentiating through its optimality (KKT) conditions. However, since the solutions of LPs are located at the corners of the feasible polytope, this method will yield zero gradients for LPs. To address this, QPTL [Wilder et al., 2019] add a quadratic regularization to the LP, thus relaxing the problem to a non-linear strongly-convex quadratic program (QP) and then use the technique in Amos and Kolter [2017]. Similarly, Cameron et al. [2022] relax Mixed Integer Programs (MIP) by using a log-barrier regularization followed by the use of the techniques in Amos and Kolter [2017]. These approaches suffer from two notable limitations: (i) they introduce additional hyper-parameters (the regularization strength), and (ii) they are only guaranteed to converge in the vicinity of the optimal solution (because of the bias introduced by the regularization). While our proposed framework also uses KKT conditions, it ensures convergence to the optimal solution without introducing additional hyper-parameters.

Predict and Optimize: The CIO problem is related to the *predict and optimize* (PO) framework [Elmachtoub and Grigas, 2022]. In contrast to the CIO, PO requires the knowledge of ground-truth costs. Our work does not assume access to this additional information, but we note that the proposed algorithms can be directly used in the PO setting.

In the next chapter, we formally formulate the problem and highlight the technical challenges.

Chapter 2

Problem Formulation and Reduction to Feasibility

2.1 Problem Formulation

We consider the optimization procedure to be a linear program (LP). Without loss of generality, we assume the standard form of the LP and define $\hat{x}(c)$ as the solution to the LP with cost-vector $c \in \mathbb{R}^m$,

$$\hat{x}(c) := \arg\min_x \langle c, x \rangle \text{ s.t } Ax = b,\ x \geq 0\,,$$

where $A \in \mathbb{R}^{n \times m}$ and $b \in \mathbb{R}^n$. The `CILP` problem consists of a training dataset $\mathcal{D} = \{z_i, x_i^*\}_{i=1}^N$ where $z_i \in \mathbb{R}^{1 \times d}$ is the input ($Z \in \mathbb{R}^{N \times d}$ is the corresponding feature matrix) and $x_i^* \in \mathbb{R}^m$ is the corresponding optimal decision. We assume that the LP parameters (A, b) encoding the constraints are known, whereas the cost vector c is unknown and will be predicted using the data.

Example: In the context of the shortest path problem (Example 2 in Section 1.1), consider an arbitrary $x, c \in \mathbb{R}^m$; for all $j \in [m]$, $x_j^* \in \{0, 1\}$ variables denote whether an edge is included in the shortest path and the weight of each edge is represented by the cost c_j. To ensure a valid path from the start vertex s to the target vertex t, the "flow" constraints are encoded via A and b. These constraints ensure that every vertex, except s and t, maintains an equal number of incoming and outgoing edges. Vertex s is constrained to have exactly one outgoing edge, and vertex t has precisely one incoming edge.

When using a model f_θ with parameters θ to predict the cost-vector, we define $\hat{x}(\hat{c})$ as:

$$\hat{x}(\hat{c}) := \arg\min_x \langle \hat{c}, x \rangle \text{ s.t } Ax = b,\ x \geq 0,\ \hat{c} = f_\theta(z)\,.$$

Given the dataset $\mathcal{D}$ and knowledge of (A, b), the `CILP` objective is to learn θ s.t. $\hat{x}(f_\theta(z_i)) \approx x_i^*$ for all $i \in [N]$. During inference, we only have access to input z and we predict $\hat{x}(f_\theta(z))$.

2.2 Challenge in Gradient Estimation

To gain some intuition as to why the typical end-to-end learning approach via auto-differentiation Paszke et al. [2019] will not work for `CILP`, consider using the squared loss to quantify the discrepancy between $\hat{x}(f_\theta(z_i))$ and x_i^*, i.e. $\ell(\theta) := \frac{1}{2}\sum_{i=1}^{N} \|\hat{x}(f_\theta(z_i)) - x_i^*\|^2$. Using the chain rule to compute the gradient with respect to θ, we get that $\frac{\partial l}{\partial \theta} = \frac{\partial l}{\partial x}\frac{\partial x}{\partial c}\frac{\partial c}{\partial \theta}$. The first and last terms can be easily calculated. However, for an LP, the decision x is piece-wise constant with respect to c, and $\frac{\partial x}{\partial c}$ is either 0 or undefined.

Consequently, in the next section, we aim to develop an algorithm that does not rely on directly calculating $\frac{\partial x}{\partial c}$.

2.3 Reduction to Convex Feasibility

For a linear model, we reduce the `CILP` problem to convex set feasibility (Section 2.3.1). In Section 2.3.2, we use alternating projections onto convex sets (POCS) to solve the feasibility problem and completely instantiate Algorithm 1. In Section 2.3.3, we describe some practical considerations when using Algorithm 1. In Section 2.3.4, we describe how to extend these ideas to handle non-linear but convex objectives and constraints.

2.3.1 Reduction

Recall that for an input $(z, x^*) \in \mathcal{D}$, we aim to find a c such that $\hat{x}(c) = x^*$. However, due to the non-uniqueness of the mapping from x to c, there are potentially infinitely many values of c that can yield x^*. We define C to represent the set encompassing all such values of c. The set C can be represented by exploiting the optimality conditions for the LP. KKT conditions [Kuhn and Tucker, 1951] give necessary and sufficient conditions for the optimality of the LP. If x^* is the solution to the standard LP, then the KKT conditions can be written as follows:

$$\nu^T A + \lambda - c = 0\,, x^* \cdot \lambda = 0\,, \lambda \geq 0\,, Ax^* = b\,, x^* \geq 0$$

where $\lambda \in \mathbb{R}_+^m, \nu \in \mathbb{R}^n$ are the dual variables, $x_i^*\lambda_i = 0$ (for all i) represents the complementary slackness condition and $Ax^* = b$, $x^* \geq 0$ represents the feasibility of x^*. At optimality, there exist dual variables (λ, ν) such that the tuple (x^*, λ, ν, c) satisfies the KKT conditions.

Since the KKT optimality conditions are both necessary and sufficient, given an optimal solution x^*, we can identify the set of cost vectors c that satisfy these conditions. This enables us to define the convex set C as:

$$\begin{aligned} C = \{c \,|\, \exists\, \lambda, \nu \text{ s.t. } & \nu^T A + \lambda - c = 0, \\ & x^* \cdot \lambda = 0, \lambda \geq 0\} \end{aligned} \tag{2.1}$$

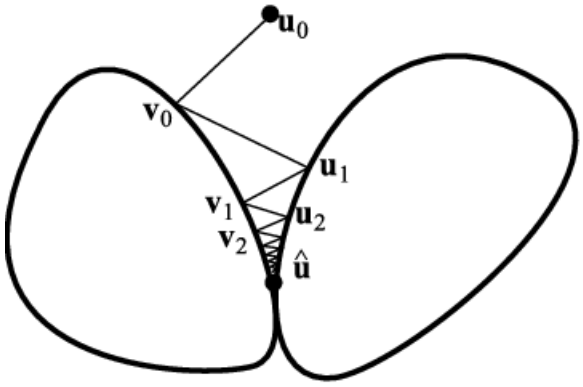

Figure 2.1: Projection onto Convex Sets (POCS) algorithm. The figure is taken from Rzepka et al. [2018]. POCS alternatively projects a point onto the two sets. Consider a point u_0; its projection on the first set is v_0. Then, the projection of point v_0 is u_1 on the second set. By repeating the projection steps, POCS converges to $\hat{u}$.

Note that we omit the condition $Ax^* = b$, $x^* \geq 0$ as it is satisfied by definition for the optimal solution x^*. For any λ, ν, the set C is affine and hence convex in c.

We define F as the set of cost vectors that are realizable by the linear model parameterized by $\theta \in \mathbb{R}^{d \times m}$. Formally, F can be written as:

$$F = \{c \,|\exists \theta \text{ s.t. } c = z\theta\}. \tag{2.2}$$

For a linear model, the set F is linear and hence convex in c. The objective of `CILP` is to find a $c \in C$ (resulting in the optimal solution x^*) and is also realizable by the model, i.e. it lies in set F. Hence, we aim to find a c that lies in the intersection $(C \cap F)$. Therefore, `CILP` is equivalent to a convex feasibility problem in this setting.

2.3.2 Algorithm

The commonly employed method for solving convex feasibility problems is the alternating projections or Projection onto Convex Sets (POCS) algorithm [Von Neumann, 1949, Bauschke and Borwein, 1996]. The POCS algorithm alternatively projects a point onto the two sets. A geometrical interpretation is given in Fig. 2.1. The algorithm is guaranteed to converge to a point in the intersection if the intersection is non-empty; otherwise, it converges to the closest point between the two sets [Deutsch, 1984, Bauschke and Borwein, 1993]. Moreover, the rate of convergence is linear in the number of POCS iterations. In order to use the POCS algorithm for the `CILP` problem for a single training point, we require the projection of an arbitrary point $q \in \mathbb{R}^m$ onto the set C. This corresponds to solving the quadratic program

(QP) as follows:

$$\mathcal{P}_C(q) := \arg\min_c ||c - q||_2^2$$
$$\text{subject to} \qquad \nu^T A - c + \lambda = 0, \lambda \cdot x^* = 0, \lambda \geq 0 \tag{2.3}$$

Eq. (2.3) returns a point $\mathcal{P}_C(q)$, the Euclidean projection of q onto C. For the projection of a point q onto the set F, we require solving the following regression problem,

$$\hat{\theta} := \arg\min_\theta \frac{1}{2} ||q - z\theta||^2 \quad ; \quad \mathcal{P}_F(q) = z\hat{\theta} \tag{2.4}$$

Eq. (2.4) returns a point $\mathcal{P}_F(q)$, the Euclidean projection of q onto F. Hence, POCS consists of alternatively solving the optimization problems in Eq. (2.3) and Eq. (2.4). In

Algorithm 1 for `CILP`

Input: A, b, Training dataset $\mathcal{D} \equiv (z_i, x_i^*)_{i=1}^N$, Model f_θ
Initialize θ_1
for $t = 1, 2, .., T$ **do**
  $\hat{c}_i = f_{\theta_t}(z_i), \forall i \in [N]$
  for $i = 1, 2, .., N$ **do**
    $q_i = \mathcal{P}_{C_i}(\hat{c}_i)$ by solving the optimization problem in Eq. (2.3)
  end
  $\theta_{t+1} = \arg\min_\theta \frac{1}{2N} \sum_{i=1}^N ||q_i - f_\theta(z_i)||^2$
end
Output: θ_{T+1}

order to extend the above idea to N training points, we will define sets C_i analogous to Eq. (2.1) for each $i \in [N]$. We define $\mathcal{C} := C_1 \times C_2 \ldots \times C_N$ to be the Cartesian product of these sets. Hence, $\mathcal{C}$ consists of concatenated vectors $(c_1, c_2, \ldots, c_N) \in \mathbb{R}^{Nm}$ where $c_i \in C_i$. Similarly, we define $\mathcal{F} := \{(c_1, c_2, \ldots, c_N) | \exists \theta \text{ s.t. } \forall i \in [N] \ , \ c_i = z_i\theta\}$. Hence, the projection onto $\mathcal{C}$ corresponds to solving the QP in Eq. (2.3) for every point $i \in [N]$. Projecting an arbitrary point $\tilde{q} = (\tilde{q}_1, \tilde{q}_2, \ldots, \tilde{q}_N) \in \mathbb{R}^{Nm}$ onto $\mathcal{F}$ involves solving the regression problem, $\hat{\theta} := \arg\min_\theta \frac{1}{2N} \sum_{i=1}^N \|\tilde{q}_i - z_i\theta\|^2$. Since the Cartesian product of sets is convex, both $\mathcal{C}$ and $\mathcal{F}$ are convex and hence the resulting POCS algorithm will converge at a linear rate.

Finally, we note that our algorithmic framework can handle a generic model f_θ, though our theoretical results only hold for a linear model. The complete algorithm for a generic model f_θ is described in Algorithm 1. Next, we describe practical considerations when implementing the algorithm.

2.3.3 Practical considerations: Margin

Recently, Sun et al. [2023] have noted the benefits of using a margin with the LP optimality conditions [Luenberger et al., 1984]. In Section B.2, we show how to modify their margin

formulation for the KKT conditions, resulting in the modified set C below:

$$C = \{c \mid \exists \lambda, \nu \text{ s.t. } \nu^T A - c + \lambda = 0,\ \lambda_i \mathcal{I}\{x_i^* \neq 0\} = 0, \lambda_i \mathcal{I}\{x_i^* = 0\} \geq \chi\} \tag{2.5}$$

There are two key motivations for adding the margin: (i) it ensures that the algorithm does not converge to a trivial solution corresponding to $c = 0$, (ii) it ensures that the algorithm will converge to the interior (rather than the boundary) of C making it more robust to perturbations and improving the algorithm's generalization performance [El Balghiti et al., 2019] to previously unseen instances. We note that our framework and the resulting algorithm is not limited to LPs. Next, we describe how to extend these ideas to handle non-linear but convex objectives and constraints.

2.3.4 Handling non-linear optimization problems

Similar to the linear case, the KKT optimality conditions can be used to derive a convex set C for a class of non-linear convex objectives and constraints [Iyengar and Kang, 2005]. As an example, we instantiate C for a specific quadratic program (QP) below and defer the general case to Chapter B.

$$\hat{x}(c) := \arg\min_{x \geq 0} -\langle c, x \rangle + \frac{\gamma}{2} x^T Q x \text{ s.t } \sum_{i=1}^{m} x_i = 1 \tag{2.6}$$

Example: For the portfolio optimization problem [Fabozzi et al., 2008] common in econometrics, x_j and c_j in Eq. (2.6) represent the fraction of investment and the expected return for stock $j \in [m]$ respectively. The matrix $Q \in \mathbb{R}^{m \times m}$ represents the risk associated with selecting similar stocks, and γ is the given risk tolerance. The task is to maximize the return while minimizing the risk, subject to simplex constraints. Given historical data and a model that can be used to predict the expected return and risk matrix, and the "best" portfolios in hindsight, the CIO problem is to infer the model parameters. In this case, the convex set C consisting of $\chi := \{c^*, Q^*\}$ that satisfy the KKT conditions of the QP is given as: $C = \{c, Q \mid \exists \lambda, \nu \text{ s.t. } \nu \mathbf{1}_m + \lambda + c - \frac{\gamma}{2}(Q + Q^T)x^* = 0,\ x^* \cdot \lambda = 0,\ \lambda \geq 0\}$ where $\lambda \in \mathbb{R}_+^m$, $\nu \in \mathbb{R}^1$, $\mathbf{1}_m = (1, 1, \ldots, 1)$. The set C is linear and therefore convex in $\{c, Q\}$. Consequently, when using a linear prediction model, the inverse problem for portfolio optimization can also be reduced to convex feasibility.

For a general non-linear convex objective $\phi(x, \omega)$ where ω represents a general cost vector, set C is convex in ω if $\frac{\partial \phi(x,\omega)}{\partial x}|_{x=x^*}$, the derivative of the objective function ϕ w.r.t x evaluated at x^* is convex in ω. For instance, this condition also applies to semi-definite programs. Finally, we note that our framework is not limited to linear constraints, and can easily handle non-linear convex constraints. Please refer to Section B.1 for the derivation.

2.3.5 Challenges for solving large-scale problems

The POCS approach described above requires computing the exact projection of point q onto the set $\mathcal{F}$. For high-dimensional problems, computing these exact projections is computationally expensive. Moreover, for non-linear models such as neural networks, $\mathcal{F}$ is non-convex and the resulting projection is ill-defined. Additionally, computing the exact projection requires iterating through the entire dataset of N points, which can be prohibitive for large datasets typical in practice.

Consequently, in the next chapter, we reduce the problem to empirical risk minimization on an appropriate smooth, convex loss satisfying the PL condition. This reduction enables the use of computationally efficient (stochastic) first-order optimization algorithms common in the machine learning literature.

Chapter 3

Reduction to Empirical Risk Minimization

For a linear model, in Section 3.1, we reduce the feasibility problem to empirical risk minimization (ERM) on an appropriate smooth, convex loss satisfying the PL condition and prove that the preconditioned gradient method (with a specific preconditioner) on this loss is equivalent to the POCS approach of Algorithm 1. Subsequently, in Section 3.3, we consider using computationally efficient (stochastic) first-order methods for minimizing the loss functions.

3.1 Reduction

We define the loss function $h(\theta)$ as follows:

$$h(\theta) := \frac{1}{2N} \sum_{i=1}^{N} \min_{q_i \in C_i} \|f_\theta(z_i) - q_i\|^2 , \tag{3.1}$$

where C_i represents the set of feasible cost vectors (defined in Eq. (2.1)) for data-point i. Hence, $h(\theta)$ represents the mean (across the data-points) of squared distances between the predicted cost vector $f_\theta(z_i)$ and the set C_i. In order to better interpret $h(\theta)$, consider a point $c_\theta = (c_1, c_2, \ldots, c_N) \in \mathbb{R}^{Nm}$ such that $c_i = f_\theta(z_i)$. Hence, $h(\theta) = \frac{d^2(c_\theta, \mathcal{C})}{N}$ where $d^2(w, \mathcal{W})$ is the squared Euclidean distance of point w to the set $\mathcal{W}$. Since $c_\theta \in \mathcal{F}$, minimizing $h(\theta)$ is related to minimizing the distance between the sets $\mathcal{F}$ and $\mathcal{C}$. Formally, in Proposition 3.1.1 (proved in Section B.4), we can reduce the feasibility problem in Section 2.3 to minimizing $h(\theta)$.

Proposition 3.1.1. *Point $\hat{c} := (c_1, c_2, \ldots, c_N)$ where $c_i = z_i\tilde{\theta}$ and $\tilde{\theta} \in \arg\min h(\theta)$ lies in the intersection $\mathcal{C} \cap \mathcal{F}$ if it exists, else $\hat{c} \in \mathcal{F}$ is the point closest to $\mathcal{C}$.*

3.2 Properties of $h(\theta)$

For a linear model where $f_\theta(z) = z\theta$, we show that $h(\theta)$ has desirable properties that allow it to be minimized efficiently. In the proposition below, we establish the convexity and smoothness of $h(\theta)$. The smoothness of a function means that its gradients are Lipschitz continuous and that the function can be upper-bounded by its quadratic approximation. This is formally defined in Eq. (Smoothness).

Proposition 3.2.1. *For a linear model $f_\theta(z) = z\theta$ parameterized by $\theta \in \mathbb{R}^{d\times m}$, assuming (without loss of generality) that $\forall i$, $\|z_i\| \leq 1$, $h(\theta)$ is a 1-smooth convex function.*

The proof of the above proposition is included in Section B.3. In addition to convexity, we prove that when using a linear model, $h(\theta)$ satisfies the Polyak-Lojasiewicz (PL) inequality [Polyak, 1964, Karimi et al., 2016]. The PL condition is a gradient domination property that implies curvature near the minima and entails that every stationary point is a global minimum. Formally, the PL inequality states that there exists a constant $\mu > 0$ such that for all θ,

$$h(\theta) - h^* \leq \frac{1}{2\mu}||\nabla h(\theta)||^2\,, \tag{3.2}$$

where h^* is the minimum of h.

Proposition 3.2.2. *For a linear model $f_\theta(z) = z\theta$ and assuming (i) (without loss of generality) that $\forall i$, $\|z_i\| \leq 1$ and (ii) $\lambda_{\min}[Z^T Z] > 0$, $h(\theta)$ is not necessarily strongly-convex but satisfies the PL inequality with $\mu = \lambda_{\min}\left[\frac{\sum_{i=1}^N z_i z_i^\top}{N}\right]$.*

We include the proof in Section B.6 and note that such a result showing that square distance functions to (non)-convex sets is PL was also recently shown in [Garrigos, 2023]. Importantly, we note that convexity coupled with the PL condition implies that the function satisfies the restricted secant inequality (RSI) [Zhang and Yin, 2013], a stronger condition than PL but weaker than strong-convexity [Karimi et al., 2016, Theorem 2].

Since we have reduced the `CILP` to a problem of minimizing a loss function with desirable properties, we can use computationally efficient techniques like gradient descent and its stochastic and adaptive variants [Kingma and Ba, 2014, Duchi et al., 2011].

3.3 First-order Methods

We first show that for a linear model, Algorithm 1 is equivalent to the preconditioned gradient method on $h(\theta)$. With $Z \in \mathbb{R}^{N\times d}$ being the feature matrix, the preconditioned gradient update for minimizing $h(\theta)$ at iteration $t \in [T]$ with step-size η and the preconditioner equal to $\left[\frac{Z^T Z}{N}\right]^{-1}$, is given as:

$$\theta_{t+1} = \theta_t - \eta\left[Z^T Z\right]^{-1} Z^T(Z\theta_t - q_t)\,, \tag{3.3}$$

where $q_t = \mathcal{P}_{\mathcal{C}}(Z\theta_t)$ and $\|Z\theta_t - q_t\|$ is the Euclidean distance to the set $\mathcal{C}$ at iteration t. Consequently, point $Z\theta_{t+1}$ is exactly the Euclidean projection of q_t onto the set $\mathcal{F}$. In Section B.5, we prove the following proposition.

Proposition 3.3.1. *For a linear model $f_\theta(z) = z\theta$, the iterates corresponding to the preconditioned gradient method on $h(\theta)$ with $\eta = 1$ are identical to Algorithm 1.*

We note that for general non-linear models, this connection to POCS and hence Algorithm 1 does not necessarily hold.

Next, we consider minimizing $h(\theta)$ using gradient descent (GD) with step-size η_t at iteration t. This results in the following general update:

$$\theta_{t+1} = \theta_t - \eta_t \frac{\sum_{i=1}^{N} \frac{\partial f_\theta(z_i)}{\partial \theta}|_{\theta=\theta_t} [f_\theta(z_i) - q_{i,t}]}{N} \tag{3.4}$$

where $q_{i,t} = \mathcal{P}_{C_i}(f_{\theta_t}(z_i))$ and $\frac{\partial f_\theta(z_i)}{\partial \theta}|_{\theta=\theta_t}$ is the Jacobian of f_θ at iterate θ_t. For a linear model, this simplifies to:

$$\theta_{t+1} = \theta_t - \frac{\eta_t}{N} \left[Z^T (Z\theta_t - q_t) \right], \tag{3.5}$$

where $q_t = \mathcal{P}_{\mathcal{C}}(Z\theta_t)$ and $\|Z\theta_t - q_t\|$ is the Euclidean distance to the set $\mathcal{C}$ at iteration t. The update in Eq. (3.5) can be interpreted as an inexact projection of q_t onto $\mathcal{F}$.

If $\tilde{\theta} \in \arg\min_\theta h(\theta)$, standard convergence results [Karimi et al., 2016] for smooth, convex and PL loss functions guarantee that GD, after T iterations, returns θ_T such that $h(\theta_T) - h(\tilde{\theta}) = O(\exp(-T))$. To illustrate what this convergence rate implies for the feasibility and consequently the `CILP` problem, consider the case where $\mathcal{C} \cap \mathcal{F}$ is non-empty. In this case, Proposition 3.1.1 guarantees that $h(\tilde{\theta}) = 0$ and hence, using GD with $T = O\left(\ln(\sqrt{N}/\epsilon)\right)$ iterations is guaranteed to return a point $\hat{c} := (c_1, c_2, \ldots, c_N) \in \mathcal{F}$ where $c_i = z_i\theta_T$ that is ϵ close to $\mathcal{C}$. Compared to Algorithm 1, we see that GD retains the fast linear convergence rate to a point in $\mathcal{C} \cap \mathcal{F}$.

GD requires iterating through the entire dataset for each update, which is inefficient for large datasets. To address this, we use stochastic gradient descent (SGD) [Robbins and Monro, 1951]. Writing $h(\theta) = \frac{1}{N}\sum_{i=1}^{N} h_i(\theta)$ where $h_i(\theta) = \frac{1}{2}\min_{q_i \in C_i} \|f_\theta(z_i) - q_i\|^2$, the SGD update with step-size η_t at iteration t is:

$$\theta_{t+1} = \theta_t - \eta_t \frac{\partial f_\theta(z_{i_t})}{\partial \theta}\bigg|_{\theta=\theta_t} [f_\theta(z_{i_t}) - q_{i_t,t}], \tag{3.6}$$

where $i_t \in [N]$ is the index of the loss function sampled uniformly at random at iteration t. For a linear model,

$$\theta_{t+1} = \theta_t - \eta_t \, z_{i_t}^T (z_{i_t}\theta_t - q_{i_t,t}), \tag{3.7}$$

where $q_{i_t,t} = \mathcal{P}_{\mathcal{C}_i}(z_{i_t}\theta_t)$. Similar to GD, the update in Eq. (3.7) can be interpreted as an inexact projection of q_{i_t} onto C_i. Compared to GD, which has an $O(N)$ per-iteration cost, SGD has an $O(1)$ iteration cost, making it preferable for large datasets. However, in general, SGD has a slower rate of convergence compared to GD. Specifically, when minimizing smooth, convex functions and PL functions, T iterations of SGD with a decreasing $O(1/T)$ step-size is guaranteed to return θ_T such that $\mathbb{E}[h(\theta_T)] - h(\tilde{\theta}) = O(1/T)$ [Karimi et al., 2016, Gower et al., 2021], where the expectation is over the random sampling in each iteration.

If an additional *interpolation* property is satisfied, SGD with a constant step-size can match the convergence rate of GD [Ma et al., 2018, Vaswani et al., 2019, Bassily et al., 2018, Raj and Bach, 2021]. Formally, for convex loss functions, interpolation is satisfied when $\tilde{\theta} := \arg\min h(\theta)$ also simultaneously minimizes each h_i, i.e. $||\nabla h_i(\tilde{\theta})|| = 0$ for all $i \in [N]$. In the context of the feasibility problem, interpolation is satisfied if $f_{\tilde{\theta}}(z_i) \in C_i$ and hence $h_i(\tilde{\theta}) = 0$ for all $i \in [N]$. This implies that the intersection $\mathcal{C} \cap \mathcal{F}$ is non-empty and in this case, SGD with a constant step-size requires $T = O\left(\ln(\sqrt{N}/\epsilon)\right)$ iterations to return a point ϵ-close to $\mathcal{C} \cap \mathcal{F}$. Notably, the PL condition is not required for convergence; smooth and convex functions (even without the PL condition) ensure convergence with first-order methods, though at a slower rate (e.g., $O(1/\sqrt{T})$ instead of $O(1/T)$ for SGD).

The above results hold when using a linear model, ensuring convexity in the resulting function $h(\theta)$. Similar guarantees extend to non-parametric techniques like kernel methods, demonstrating the generality of our results. However, for expressive models such as deep neural networks, convexity is not necessarily satisfied. In certain regimes of over-parametrized neural networks, conditions resembling PL or variations thereof are satisfied [Liu et al., 2022, 2023]. In these cases, SGD can still achieve linear convergence [Vaswani et al., 2019, Bassily et al., 2018], matching the results in the convex case.

The above results are concerned with minimizing the loss on the training dataset. In the next chapter, we study the generalization performance of SGD on previously unseen instances sampled from the same distribution.

Chapter 4

Generalization Guarantees and Sub-optimality

4.1 Generalization Guarantees

In this section, we use the existing results on algorithmic stability [Bousquet and Elisseeff, 2002, Hardt et al., 2016, Lei and Ying, 2020] to control the generalization error and subsequently bound the suboptimality for `CILP`.

We first define the necessary notation and recall the necessary results from the algorithmic stability literature. We define ρ to be the probability measure on the sample space $\mathcal{Y} = \mathcal{Z} \times \mathcal{X}^*$, where $\mathcal{Z} \subseteq \mathbb{R}^d$ and $\mathcal{X}^* \subseteq \mathbb{R}^m$. We assume that the training dataset $\mathcal{D} = \{(z_1, x_1^*), \cdots, (z_N, x_N^*)\}$, is drawn independently and identically from ρ. We define $h(\theta, (z, x^*)) := \frac{1}{2} \min_{q \in C(x^*)} \|f_\theta(z) - q\|^2$, where $C(x^*)$ is the set constructed according to Eq. (2.1). Furthermore, we denote the *population loss* for parameter θ as: $\hat{h}(\theta) = \mathbb{E}_{(z,x^*)\sim\rho}[h(\theta, (z, x^*))]$ where $(z, x^*) \sim \rho$ implies the sample (z, x^*) is drawn independently from ρ.s

Based on algorithmic stability, Lei and Ying [2021] prove the following generalization result for learning with smooth loss functions satisfying the PL condition.

Theorem 4.1.1. *(Theorem 1 in [Lei and Ying, 2021]) Let $\theta_{\mathcal{D}}$ denote the output of a randomized algorithm $\mathcal{A}$ when minimizing an L-smooth function h that satisfies PL inequality with constant μ. Under the condition $N \geq 4L/\mu$, we have,*

$$\mathbb{E}[\hat{h}(\theta_{\mathcal{D}}) - \inf_\theta h(\theta)] = O\left(\frac{\mathbb{E}[\inf_\theta h(\theta)]}{N\mu} + \frac{\mathbb{E}[h(\theta_{\mathcal{D}}) - \inf_\theta h(\theta)]}{\mu}\right) \tag{4.1}$$

The expectation in the above theorem is w.r.t the randomness in selecting the training dataset of size N and w.r.t the stochasticity in the learning algorithm. In our context, since the randomized algorithm $\mathcal{A}$ is SGD, the bound on its generalization is a direct consequence of Theorem 4.1.1. In particular, since $\mathbb{E}_{\mathcal{D}}[\inf_\theta h(\theta)] \leq \inf_\theta \mathbb{E}_{\mathcal{D}}[h(\theta)] = \inf_\theta \hat{h}(\theta)$, we can obtain the following result from Lei and Ying [2021].

Corollary 4.1.2. *(Theorem 6 in [Lei and Ying, 2021]) When minimizing an L-smooth, μ-RSI function, SGD with step-size $\eta_t = \frac{1}{\mu(t+1)}$ for all $t > 0$ has the following guarantee,*

$$\mathbb{E}[\hat{h}(\theta_T)] - \inf_\theta \hat{h}(\theta) = O\left(\frac{1}{N\mu} + \frac{1}{\mu^2 T}\right).$$

The expectation in the above result is only over the stochasticity in SGD. The LHS represents the *excess risk*, while the first term on the RHS decreases as N increases, and the second term on the RHS represents the average (over $\mathcal{D}$) optimization error that decreases as T increases.

In the interpolation setting, since the model can fit *any* training dataset of size N using SGD, $\inf_\theta \mathbb{E}[h(\theta)] = 0$. In this case, we obtain the following result from Lei and Ying [2021, Theorem 7].

Corollary 4.1.3. *When minimizing an L-smooth, μ-RSI function and if $\inf_\theta \mathbb{E}[h(\theta)] = 0$ for any choice of training dataset $\mathcal{D}$ of size N, SGD with step-size $\eta_t = \eta = \frac{1}{L}$ for all $t > 0$ has the following guarantee,*

$$\mathbb{E}[\hat{h}(\theta_T)] = O\left(\frac{L(1-\frac{\mu}{L})^T}{2\mu}\right).$$

The above result shows that in the interpolation setting, the expected (over the randomness in SGD) population loss decreases at a linear rate (depending on T). Importantly, the above bound does not depend on N. Intuitively, if $h(\theta)$ is smooth and N is large enough s.t. it satisfies the PL condition with $\mu > 0$, minimizing the loss over a single dataset results in minimizing the population loss.

The above results bound the population loss $\hat{h}(\theta)$, which serves as a proxy for the decision quality of SGD. Subsequently, we establish a connection between $h(\theta)$ and the suboptimality in the CIO framework.

4.2 Sub-optimality

In this section, we first argue about the shortcomings of previous definitions of sub-optimality to measure performance for `CILP` and then propose a new sub-optimality metric.

Recently, Sun et al. [2023] define the suboptimality gap as $\Gamma_1(\theta, (z, x^*)) := \langle c_\theta,\, x^* - \hat{x}(c_\theta)\rangle$ where $c_\theta = f_\theta(z)$, and prove theoretical guarantees for this loss. We argue that $\Gamma_1(z, x^*)$ is not the right metric as the predicted c_θ can be made arbitrarily small, resulting in smaller values of $\Gamma_1(z, x^*)$ without ensuring that $x^* \approx \hat{x}(c_\theta)$. On the other hand, work in predict-and-optimize [Elmachtoub and Grigas, 2022] assumes access to the ground-truth cost-vector c^* and proposes to use a sub-optimality $\Gamma_2(\theta, (z, c^*, x^*)) := \langle c^*,\, \hat{x}(c_\theta) - x^*\rangle$. Since we do not

have access to c^*, we cannot directly use this measure of sub-optimality. Consequently, we use the projection of c_θ onto $\mathcal{C}$ as a proxy for the ground-truth c^* and define the suboptimality gap as:

$$\Gamma(z, x^*) = \left\langle \frac{\mathcal{P}_C(c_\theta)}{\|\mathcal{P}_C(c_\theta)\|_2}, \hat{x}(c_\theta) - x^* \right\rangle \tag{4.2}$$

It is important to note that we divide $\mathcal{P}_C(c_\theta)$ by its corresponding ℓ_2 norm to make the sub-optimality scale-invariant i.e. small values of $\mathcal{P}_C(c_\theta)$ do not necessarily imply small sub-optimality (unlike Γ_1). We now relate the sub-optimality to the loss $h(\theta)$ and prove the following result in Section B.7.

Proposition 4.2.1. *For $c_\theta \in \mathbb{R}^m := f_\theta(z)$, assuming that $\forall j \in [m], [\hat{x}(c_\theta)]_j, x^*_j \in [0, 1]$, $\Gamma(\theta, (z, x^*)) \leq \frac{\sqrt{2m\, h(\theta)}}{\delta}$ where $\delta := O(\chi/\sqrt{m})$, χ is the margin and the O notation hides constants that depend on the LP.*

As the sub-optimality is upper-bounded by $O(\sqrt{h(\theta)})$, we can control it by controlling the loss $h(\theta)$. Putting together the results in Corollaries 4.1.2 and 4.1.3 and Proposition 4.2.1, we observe that T iterations of SGD result in the following bounds on the expected sub-optimality:

$$\mathbb{E}_{(z,x^*)\sim\rho}\left[\mathbb{E}_{\mathcal{D}\sim\rho}\left[\mathbb{E}[\Gamma(\theta_T, (z, x^*))]\right]\right] = O\left(\frac{\sqrt{2m}}{\delta}\left[\inf_\theta \hat{h}(\theta) + \left[\frac{1}{N\mu} + \frac{1}{\mu^2 T}\right]\right]^{1/2}\right) \tag{4.3}$$

in the general setting and $O\left(\frac{\sqrt{2m}}{\delta}\left[\exp(-\mu\, T)\right]^{1/2}\right)$ (independent of N) in the interpolation setting. Compared to these results, Sun et al. [2023] derive an $O(1/\sqrt{N})$ bound on the expected sub-optimality in terms of Γ_1 for both the interpolation and general settings. In the next chapter, we compare our method against several baselines on real-world and synthetic datasets and present the results.

Chapter 5

Experimental Results

In this chapter, we present the experimental setup and present our findings.[1]

5.1 Datasets and Model

To validate the effectiveness of our approach, we experiment with both synthetic and real-world benchmarks. We consider two real-world tasks [Vlastelica et al., 2019] – Warcraft Shortest Path and Perfect Matching below and defer the synthetic experiments to Chapter C.

Warcraft Shortest Path (SP): The dataset consists of (z, x^*) pairs where the input z is an RGB image generated from the Warcraft II tileset. The output x^* corresponds to the shortest path between given source-target pairs. The model predicts the edge weights for each tile in a $k \times k$ grid (where $k \in \{12, 18\}$). Given these edge-weights, the optimization problem is to find the shortest path.

Perfect Matching (PM): The dataset consists of (z, x^*) pairs where the input z is a grey-scale image consisting of MNIST digits on a $k \times k$ grid. The output x^* is a matching for each digit to one of its neighbors on the grid. The model predicts the edge-weights between each pair of neighbouring digits. Given these edge-weights, the optimization problem is to find a matching that has the minimal cumulative weight of the selected edges.

Both datasets consist of 10000 training samples, 1000 validation samples and 1000 test samples each. For both SP and PM, we use Resnet-18 [He et al., 2016] followed by a softplus function $s(x) = \log(1 + \exp(x))$ to ensure the predicted cost is non-negative. Please refer to Section C.2 for additional details about the model and datasets.

Methods: We compare the proposed method against several existing methods, including ST [Sahoo et al., 2022], MOM [Sun et al., 2023], BB [Vlastelica et al., 2019], QPTL [Wilder

[1]The code is available here.

et al., 2019] and SPO+ [Bertsimas and Kallus, 2020]. We use adaptive first-order methods: AdaGrad [Duchi et al., 2011] and Adam [Kingma and Ba, 2014] to minimize the loss in Eq. (3.1) for our method and the corresponding losses for the other baselines. We train all the methods for 50 epochs with a batch size of 100. We employ a grid search to find the best constant step size in $\{0.1, 0.05, 0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00005\}$, across both the Adam and Adagrad optimizers. The optimal settings are determined based on performance on the validation set. For optimal settings, we consider 5 independent runs and plot the average result and standard deviation. Following Sun et al. [2023], we set $\chi = 1$ for all our experiments. In Section C.4, we also provide an ablation study varying χ in Table C.1 and show that the algorithm is robust to χ. We note that the QPTL approach is excluded from real-world experiments as it is prohibitively slow for large problems [Amos and Kolter, 2017, Geng et al., 2023]. For all the methods, we implement the LPs and QPs using the CVXPY library Diamond and Boyd [2016]. For LPs, we use the ECOS solver Domahidi et al. [2013], and for QPs, we use the OSQP solver Stellato et al. [2020].

5.2 Metrics

For each method, we plot the standard metrics: *estimate-loss* and *decision-loss* on both the train and test set, defined as:

$$\text{Estimate-Loss}(\theta) = \sum_{i=1}^{N} \langle c_i^*, \hat{x}(f_\theta(z_i)) \rangle - \langle c_i^*, x_i^* \rangle \tag{5.1}$$

$$\text{Decision-Loss}(\theta) = \sum_{i=1}^{N} \|\hat{x}(f_\theta(z_i)) - x_i^*\|^2 \tag{5.2}$$

Since all the datasets consist of (z_i, c_i^*, x_i^*) pairs where $x_i^* = \hat{x}(c_i^*)$, the estimate-loss and decision-loss are commonly used to measure performance in these tasks[2]. We note that SPO+ requires access to the ground-truth cost-vector c^*, while other methods, including ours, do not. Though the MOM method does not require access to c^* in principle, the paper's implementation[3] uses this ground-truth information to calculate the basis and use the LP optimality conditions. We continue using this information for MOM, thereby overestimating its performance.

[2]Experimentally, we found that the sub-optimality in Eq. (4.2) has a similar trend as the estimate-loss.

[3]See MOM code

5.3 Results

In Fig. 5.1 with respect to the decision-loss, our method consistently outperforms the baselines by a considerable margin across tasks. In Fig. 5.2, with respect to the estimate-loss, our method outperforms all the baselines except for SPO+ on the SP problem. In Section C.3, we plot the wall-clock time/epoch for all the methods, and observe that our method is comparable to the baselines and scales gracefully as the dimension and the number of training examples increase. These results demonstrate the strong empirical performance of our method compared to other baselines.

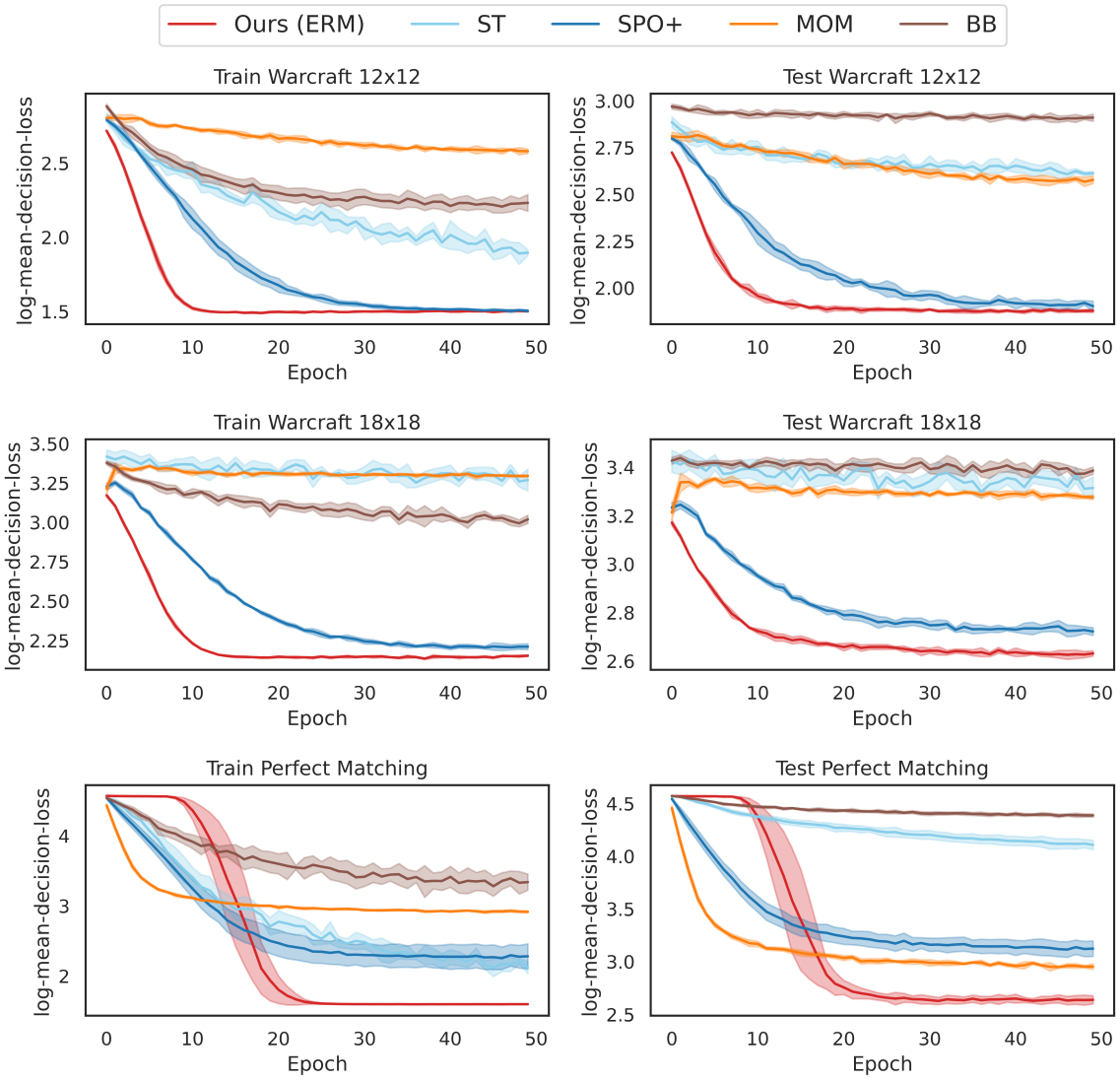

Figure 5.1: Decision loss: Training and Test plot for the real world experiments. Our method significantly outperforms the other methods (ST, BB, MOM, SPO+).

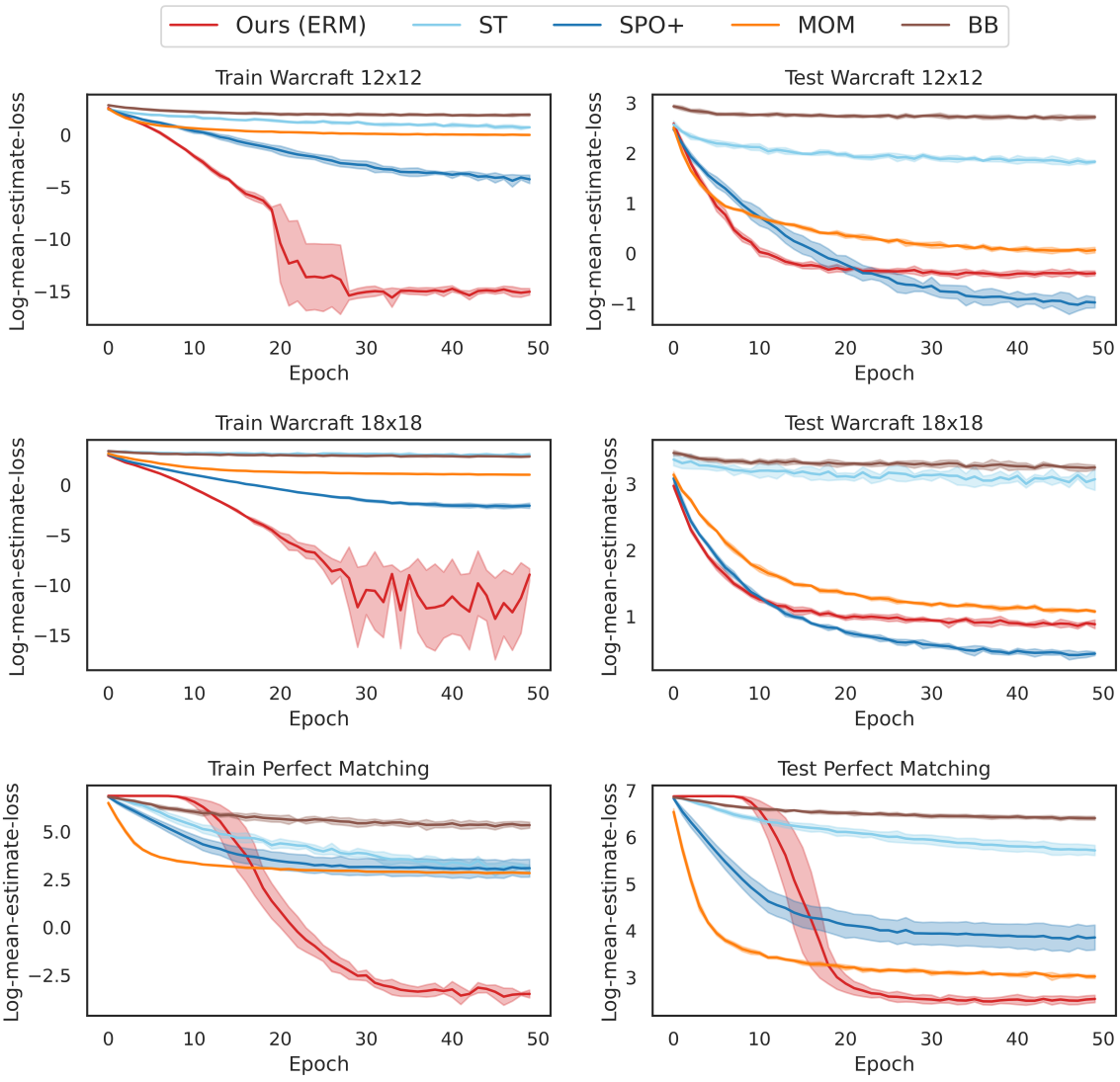

Figure 5.2: Estimate loss: training and test plots for real-world experiments. Our method significantly outperforms existing methods (ST, BB, MOM) and is comparable to SPO+, which uses the knowledge of c^*.

Chapter 6

Discussion

We presented a reduction of CIO to convex feasibility, which enabled us to guarantee linear convergence to the solution without additional assumptions such as degeneracy or interpolation. We further reduced it to ERM on a smooth, convex loss that satisfies the PL condition. This enabled us to use first-order optimizers and demonstrate strong empirical performance on real-world tasks while being computationally efficient. For future work, we aim to address the following areas: (1) since solving the QP takes a substantial amount of time, we plan to incorporate techniques from Lavington et al. [2023] to allow for multiple updates to the model for every solve of the QP, (2) we intend to extend our framework to accommodate unknown constraints, broadening its applicability (3) finally, we aim to experiment with general non-linear convex objectives.

Bibliography

Saurabh Mishra, Anant Raj, and Sharan Vaswani. From inverse optimization to feasibility to erm. In *Forty-first International Conference on Machine Learning*, 2024. (cited on iv)

Chunlin Sun, Shang Liu, and Xiaocheng Li. Maximum optimality margin: A unified approach for contextual linear programming and inverse linear programming. *arXiv preprint arXiv:2301.11260*, 2023. (cited on vii, 1, 3, 4, 9, 17, 18, 19, 20, 33, 34, 43)

Dominik Rzepka, Marek Miśkowicz, Dariusz Kościelnik, and Nguyen T Thao. Reconstruction of signals from level-crossing samples using implicit information. *IEEE Access*, 6:35001–35011, 2018. (cited on ix, 8)

Clemens Heuberger. Inverse combinatorial optimization: A survey on problems, methods, and results. *Journal of combinatorial optimization*, 8:329–361, 2004. (cited on 1)

Dimitris Bertsimas, Vishal Gupta, and Ioannis Ch Paschalidis. Data-driven estimation in equilibrium using inverse optimization. *Mathematical Programming*, 153:595–633, 2015. (cited on 1)

John R Birge, Ali Hortaçsu, and J Michael Pavlin. Inverse optimization for the recovery of market structure from market outcomes: An application to the miso electricity market. *Operations Research*, 65(4):837–855, 2017. (cited on 1)

Timothy CY Chan, Maria Eberg, Katharina Forster, Claire Holloway, Luciano Ieraci, Yusuf Shalaby, and Nasrin Yousefi. An inverse optimization approach to measuring clinical pathway concordance. *Management Science*, 68(3):1882–1903, 2022. (cited on 1)

Timothy CY Chan, Rafid Mahmood, and Ian Yihang Zhu. Inverse optimization: Theory and applications. *Operations Research*, 2023. (cited on 1)

Omar Besbes, Yuri Fonseca, and Ilan Lobel. Contextual inverse optimization: Offline and online learning. *Operations Research*, 2023. (cited on 1, 4)

Peyman Mohajerin Esfahani, Soroosh Shafieezadeh-Abadeh, Grani A Hanasusanto, and Daniel Kuhn. Data-driven inverse optimization with imperfect information. *Mathematical Programming*, 167:191–234, 2018. (cited on 1, 4)

Bingjie Li, Guohua Wu, Yongming He, Mingfeng Fan, and Witold Pedrycz. An overview and experimental study of learning-based optimization algorithms for the vehicle routing problem. *IEEE/CAA Journal of Automatica Sinica*, 9:1115–1138, 2022. (cited on 1)

Gerard Cornuejols and Reha Tütüncü. *Optimization methods in finance*, volume 5. Cambridge University Press, 2006. (cited on 1)

RC Bansal. Optimization methods for electric power systems: An overview. *International Journal of Emerging Electric Power Systems*, 2, 2005. (cited on 1)

Jia Li, Feng Liu, Zhaojian Wang, Steven H Low, and Shengwei Mei. Optimal power flow in stand-alone dc microgrids. *IEEE Transactions on Power Systems*, 33:5496–5506, 2018. (cited on 1)

Mallik Angalakudati, Siddharth Balwani, Jorge Calzada, Bikram Chatterjee, Georgia Perakis, Nicolas Raad, and Joline Uichanco. Business analytics for flexible resource allocation under random emergencies. *Management Science*, 60:1552–1573, 2014. (cited on 1)

Stephen P Boyd, Thomas H Lee, et al. Optimal design of a cmos op-amp via geometric programming. *IEEE Transactions on Computer-aided design of integrated circuits and systems*, 20:1–21, 2001. (cited on 1)

Purushothaman Raja and Sivagurunathan Pugazhenthi. Optimal path planning of mobile robots: A review. *International journal of physical sciences*, 7:1314–1320, 2012. (cited on 1)

Dariush Wahdany, Carlo Schmitt, and Jochen L Cremer. More than accuracy: end-to-end wind power forecasting that optimises the energy system. *Electric Power Systems Research*, 221:109384, 2023. (cited on 1)

J Guyomarch. Warcraft ii open-source map editor. *URL http://github. com/war2/war2edit*, 2017. (cited on 1)

Andrew Y Ng, Stuart Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000. (cited on 2)

Bryan Wilder, Bistra Dilkina, and Milind Tambe. Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1658–1665, 2019. (cited on 2, 5, 19, 43)

Boris T Polyak. Gradient methods for solving equations and inequalities. *USSR Computational Mathematics and Mathematical Physics*, 4(6):17–32, 1964. (cited on 3, 13)

Marin Vlastelica, Anselm Paulus, Vít Musil, Georg Martius, and Michal Rolínek. Differentiation of blackbox combinatorial solvers. *arXiv preprint arXiv:1912.02175*, 2019. (cited on 3, 4, 19, 43, 45, 46)

Garud Iyengar and Wanmo Kang. Inverse conic programming with applications. *Operations Research Letters*, 33(3):319–330, 2005. (cited on 4, 10)

David G Luenberger, Yinyu Ye, et al. *Linear and nonlinear programming*, volume 2. Springer, 1984. (cited on 4, 9, 41)

Subham Sekhar Sahoo, Anselm Paulus, Marin Vlastelica, Vít Musil, Volodymyr Kuleshov, and Georg Martius. Backpropagation through combinatorial algorithms: Identity with projection works. *arXiv preprint arXiv:2205.15213*, 2022. (cited on 4, 19, 43)

Yingcong Tan, Daria Terekhov, and Andrew Delong. Learning linear programs from optimal decisions. *Advances in Neural Information Processing Systems*, 33:19738–19749, 2020. (cited on 4)

Quentin Berthet, Mathieu Blondel, Olivier Teboul, Marco Cuturi, Jean-Philippe Vert, and Francis Bach. Learning with differentiable pertubed optimizers. *Advances in neural information processing systems*, 33:9508–9519, 2020. (cited on 5)

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. (cited on 5, 20)

Brandon Amos. Differentiable optimization-based modeling for machine learning. *Ph. D. thesis*, 2019. (cited on 5)

Chris Cameron, Jason Hartford, Taylor Lundy, and Kevin Leyton-Brown. The perils of learning before optimizing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 3708–3715, 2022. (cited on 5)

Adam N Elmachtoub and Paul Grigas. Smart "predict, then optimize". *Management Science*, 68(1):9–26, 2022. (cited on 5, 17, 43)

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. (cited on 7)

Harold W Kuhn and Albert W Tucker. Nonlinear programming, paper presented at proceedings of the second berkeley symposium on mathematical statistics and probability, 1951. (cited on 7)

John Von Neumann. On rings of operators. reduction theory. *Annals of Mathematics*, pages 401–485, 1949. (cited on 8)

Heinz H Bauschke and Jonathan M Borwein. On projection algorithms for solving convex feasibility problems. *SIAM review*, 38(3):367–426, 1996. (cited on 8)

Frank Deutsch. Rate of convergence of the method of alternating projections. *Parametric optimization and approximation (Oberwolfach, 1983)*, 72:96–107, 1984. (cited on 8)

Heinz H Bauschke and Jonathan M Borwein. On the convergence of von neumann's alternating projection algorithm for two sets. *Set-Valued Analysis*, 1:185–212, 1993. (cited on 8)

Othman El Balghiti, Adam N Elmachtoub, Paul Grigas, and Ambuj Tewari. Generalization bounds in the predict-then-optimize framework. *Advances in neural information processing systems*, 32, 2019. (cited on 10)

Frank J Fabozzi, Harry M Markowitz, and Francis Gupta. Portfolio selection. *Handbook of finance*, 2, 2008. (cited on 10)

Hamed Karimi, Julie Nutini, and Mark Schmidt. Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part I 16*, pages 795–811. Springer, 2016. (cited on 13, 14, 15)

Guillaume Garrigos. Square distance functions are polyak-{\L} ojasiewicz and vice-versa. *arXiv preprint arXiv:2301.10332*, 2023. (cited on 13)

Hui Zhang and Wotao Yin. Gradient methods for convex minimization: better rates under weaker conditions. *arXiv preprint arXiv:1303.4645*, 2013. (cited on 13)

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. (cited on 13, 20)

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011. (cited on 13, 20)

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951. (cited on 14)

Robert Gower, Othmane Sebbouh, and Nicolas Loizou. Sgd for structured nonconvex functions: Learning rates, minibatching and interpolation. In *International Conference on Artificial Intelligence and Statistics*, pages 1315–1323. PMLR, 2021. (cited on 15)

Siyuan Ma, Raef Bassily, and Mikhail Belkin. The power of interpolation: Understanding the effectiveness of sgd in modern over-parametrized learning. In *International Conference on Machine Learning*, pages 3325–3334. PMLR, 2018. (cited on 15)

Sharan Vaswani, Francis Bach, and Mark Schmidt. Fast and faster convergence of sgd for over-parameterized models and an accelerated perceptron. In *The 22nd international conference on artificial intelligence and statistics*, pages 1195–1204. PMLR, 2019. (cited on 15)

Raef Bassily, Mikhail Belkin, and Siyuan Ma. On exponential convergence of sgd in non-convex over-parametrized learning. *arXiv preprint arXiv:1811.02564*, 2018. (cited on 15)

Anant Raj and Francis Bach. Explicit regularization of stochastic gradient methods through duality. In *International Conference on Artificial Intelligence and Statistics*, pages 1882–1890. PMLR, 2021. (cited on 15)

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. (cited on 15)

Chaoyue Liu, Dmitriy Drusvyatskiy, Mikhail Belkin, Damek Davis, and Yi-An Ma. Aiming towards the minimizers: fast convergence of sgd for overparametrized problems. *arXiv preprint arXiv:2306.02601*, 2023. (cited on 15)

Olivier Bousquet and André Elisseeff. Stability and generalization. *The Journal of Machine Learning Research*, 2:499–526, 2002. (cited on 16)

Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *International conference on machine learning*, pages 1225–1234. PMLR, 2016. (cited on 16)

Yunwen Lei and Yiming Ying. Fine-grained analysis of stability and generalization for stochastic gradient descent. In *International Conference on Machine Learning*, pages 5809–5819. PMLR, 2020. (cited on 16)

Yunwen Lei and Yiming Ying. Sharper generalization bounds for learning with gradient-dominated objective functions. In *International Conference on Learning Representations*, 2021. (cited on 16, 17)

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. (cited on 19)

Dimitris Bertsimas and Nathan Kallus. From predictive to prescriptive analytics. *Management Science*, 66:1025–1044, 2020. (cited on 20)

Haoyu Geng, Han Ruan, Runzhong Wang, Yang Li, Yang Wang, Lei Chen, and Junchi Yan. Rethinking and benchmarking predict-then-optimize paradigm for combinatorial optimization problems. *arXiv preprint arXiv:2311.07633*, 2023. (cited on 20)

Steven Diamond and Stephen Boyd. Cvxpy: A python-embedded modeling language for convex optimization. *The Journal of Machine Learning Research*, 17:2909–2913, 2016. (cited on 20)

Alexander Domahidi, Eric Chu, and Stephen Boyd. Ecos: An socp solver for embedded systems. In *2013 European control conference (ECC)*, pages 3071–3076. IEEE, 2013. (cited on 20)

Bartolomeo Stellato, Goran Banjac, Paul Goulart, Alberto Bemporad, and Stephen Boyd. Osqp: An operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 12(4):637–672, 2020. (cited on 20)

Jonathan Wilder Lavington, Sharan Vaswani, Reza Babanezhad, Mark Schmidt, and Nicolas Le Roux. Target-based surrogates for stochastic optimization. *arXiv preprint arXiv:2302.02607*, 2023. (cited on 24)

Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004. (cited on 37)

Larry Armijo. Minimization of functions having Lipschitz continuous first partial derivatives. *Pacific Journal of Mathematics*, 16(1):1 – 3, 1966. (cited on 43)

Supplementary material

Organization of the Appendix

Appendix A

Definitions

If the function f is differentiable and L-smooth, then for all v and w,

$$f(v) \leq f(w) + \langle \nabla f(w), v - w \rangle + \frac{L}{2} \|v - w\|^2, \qquad \text{(Smoothness)}$$

If f is convex, then for all v and w,

$$f(v) \geq f(w) + \langle \nabla f(w), v - w \rangle, \qquad \text{(Convexity)}$$

If f is μ strongly-convex, then for all v and w,

$$f(v) \geq f(w) + \langle \nabla f(w), v - w \rangle + \frac{\mu}{2} \|v - w\|^2 \qquad \text{(Strong Convexity)}$$

Appendix B

Theoretical Results

B.1 Generalizing our method to other classes of optimization problem:

In this section, we relax our assumption on the class of problem from LP to non-linear convex optimization objectives and constraints. We specify the condition on the objective for our reduction to hold; there is no extra condition (apart from convexity) on optimization constraints for reduction to hold. Similar to Section 2.3.1, we consider a single data-point $(z, x^*) \in \mathcal{D}$ with $|\mathcal{D}| = N$, where $z \in \mathbb{R}^d, x^* \in \mathbb{R}^m$ we aim to find a $c \in \mathbb{R}^p$ such that $\hat{x}(c) = x^*$. Consider a non-linear convex optimization problem defined as :

$$\begin{aligned} \hat{x}(c) = \ & \arg\min_x \phi(x, c) \\ \text{subject to} \quad & G(x) = 0 \\ & H(x) \leq 0 \end{aligned} \tag{B.1}$$

where $G(x), H(x)$ consist of k, l convex constraints respectively and, $\phi(x, c), G_i(x), H_i(x)$ are (non-linear) convex function with respect to parameter x.

Assumptions: For this reduction to hold, the condition we have is $\frac{\partial \phi(c,x)}{\partial x}|_{x=x^*}$ should be convex. *Example:* For LPs, the condition $\frac{\partial <c,x>}{\partial x}|_{x=x^*}$ simplifies to c which is convex in c.

Writing the KKT optimality conditions for Eq. (B.1), we get:

$$\begin{aligned} \frac{\partial \phi(x,c)}{\partial x} + \frac{\nu \cdot \partial G(x)}{\partial x} + \frac{\lambda \cdot \partial H(x)}{\partial x} \in 0 \quad & \text{(stationary condition)} \\ \lambda \geq 0 \quad & \text{(dual feasibility)} \\ \lambda \cdot H(x) = 0 \quad & \text{(complementary slackness)} \\ G(x) = 0 \quad & \text{(primal feasibility)} \\ H(x) \leq 0 \quad & \text{(primal feasibility)} \end{aligned}$$

where $\lambda \in \mathbb{R}^l, \nu \in \mathbb{R}^k$ are referred to as the dual-variables. The equations, $G(x) = 0, H(x) \leq 0$ come from problem definition. The first term is derived by differentiating the Lagrangian. The term $\lambda \cdot H(x) = 0$ represents the complementary slackness condition.

Morever, after substituing the value of x^* in Eq. (primal feasibility), we get:

$$\begin{aligned} \frac{\partial \phi(x,c)}{\partial x}\Big|_{x=x^*} + \frac{\nu \cdot \partial G(x)}{\partial x}\Big|_{x=x^*} + \frac{\lambda \cdot \partial H(x)}{\partial x}\Big|_{x=x^*} &\in 0 \\ \lambda &\geq 0 \\ \lambda \cdot H(x^*) &= 0 \end{aligned} \tag{B.2}$$

Now, for a given optimal decision x^*, $H(x^*), G(x^*)$ are inherently satisfied. Thus, we omit them from the Eq. (B.2). The term $\frac{\partial G(x)}{\partial x}|_{x=x^*}$ represents gradient of $G(x)$ taken w.r.t x and evaluated at x^*.

From our assumption, $\frac{\partial \phi(x,c)}{\partial x}\Big|_{x=x^*}$ is convex in c and as λ, ν are multiplied with constants. Thus the Eq. (B.2) is convex in c, λ, ν.

Therefore, the feasible set C encompassing all the values of c s.t. $\hat{x}(c) = x^*$ can be written as:

$$C = \Big\{ c \mid \exists\, \lambda, \nu \text{ s.t. } \frac{\partial \phi(x,c)}{\partial x}\Big|_{x=x^*} + \frac{\nu \cdot \partial G(x)}{\partial x}\Big|_{x=x^*} + \frac{\lambda \cdot \partial H(x)}{\partial x}\Big|_{x=x^*} \in 0,\ \lambda \geq 0,\ \lambda \cdot H(x^*) = 0 \Big\} \tag{B.3}$$

Moreover, the projection of point $\hat{c}$ onto set C can be attained by solving the below QP:

$$q(\hat{c}) = \arg\min_{c} \|c - \hat{c}\|^2 \tag{B.4}$$

$$\text{subject to} \quad \frac{\partial \phi(x,c)}{\partial x}\Big|_{x=x^*} + \frac{\nu \cdot \partial G(x)}{\partial x}\Big|_{x=x^*} + \frac{\lambda \cdot \partial H(x)}{\partial x}\Big|_{x=x^*} \in 0 \tag{B.5}$$

$$\lambda \geq 0 \tag{B.6}$$

$$\lambda \cdot H(x^*) = 0 \tag{B.7}$$

Thus, our framework can be used for a non-linear convex class of functions where the first-order KKT conditions are both sufficient and necessary and $\frac{\partial \phi(x,c)}{\partial x}|_{x=x^*}$ is convex in c. Moreover, we do not assume any additional condition on constraints G, H apart from convexity.

B.2 Equivalence between margin in KKT formulation and Sun et al. [2023]

In this section, we derive the equivalent margin for the KKT formulation as in Sun et al. [2023]. We further show that our margin formulation can be extended to the degenerate

LPs. Additionally, our margin formulation does not require specific handling in the case of degenerate/non-degenerate cases.

First, we will derive the same margin formulation as in Sun et al. [2023] for non-degenerate LPs in terms of KKT conditions.

In the case of non-degenerate LP, we denote B as the basis set defined as $B := \{j \mid x_j^* > 0\}$ and M as the set of indices not in B, i.e. $M = [n] - B$. A_B represents the columns corresponding to the indices in set B, is invertible by definition. The reduced cost is given as $c_M - c_B(A_B)^{-1}A_M \geq 0$ from LP optimality conditions. And to add margin χ in the reduced cost optimality conditions, Sun et al. [2023] proposed the following modification: $c_M - c_B(A_B)^{-1}A_M \geq \chi$.

Recall the the KKT conditions for LP:

$$\nu^T A - c + \lambda = 0 \tag{B.8}$$
$$\lambda \geq 0 \tag{B.9}$$
$$\lambda \cdot x^* = 0 \tag{B.10}$$

From KKT conditions, we have the condition $\nu^T A - c + \lambda = 0$. Separating it row-wise for index B, M respectively for matrix A , we get:

$$\nu^T A_B - c_B + \lambda_B = 0 \tag{B.11}$$
$$\nu^T A_M - c_M + \lambda_M = 0 \tag{B.12}$$

where A_B, A_M are the the corresponding columns of matrix A defined by set B and M respectively.

Now, from Eq. (B.10), we have $\lambda \cdot x^* = 0$. For non-degenerate LPs, we have $x_B > 0$; this implies that $\lambda_B = 0$. Substituting this value in Eq. (B.11), we get:

$$\nu^T = c_B(A_B)^{-1} \qquad \text{(substituting } \lambda_B = 0\text{)}$$
$$\lambda_M = c_M - \nu^T A_M \qquad \text{(rearranging Eq. (B.12))}$$
$$\lambda_M = c_M - c_B(A_B)^{-1}A_M \tag{B.13}$$

From Eq. (B.9), we have $\lambda \geq 0$ that implies $\lambda_M \geq 0$. Thus, in Eq. (B.13), we retrieve the reduced cost optimality condition from KKT formulation. Therefore, the equivalent of $c_M - c_B(A_B)^{-1}A_M \geq \chi$ to the KKT formulation would be to impose the same constraint on λ_M, i.e. $\lambda_M \geq \chi$.

B.2.1 Extension to degenerate case

In the case of degenerate LPs, $\exists i \in B$ s.t. $x_i = 0$. Moreover, the set B is no longer unique. Our experiments found that a choice of B affects the results in Sun et al. [2023].

Here, let us define a separate set of basis B_d, M_d as: $B_d := \{i \mid x_i^* > 0\}$, $M_d := \{i \mid x_i^* = 0\}$. Note that B_d, M_d differ from the standard definition of basis in LPs.

For the degenerate case, we propose the following modification for margin, $\lambda_{M_d} \geq \chi$. This can be written as:

$$\forall i \in [n]; \;\; \lambda_i \mathcal{I}\{x_i^* = 0\} \geq \chi \tag{B.14}$$

Note that margin modification in Eq. (B.14) is exactly the same as in the non-degenerate case. Thus, unifying the margin formulation for the degenerate and non-degenerate LPs. Moreover, this also resolves the problem of determining the basis for degenerate LPs.

B.3 Proof of Proposition 3.2.1

Proposition 3.2.1. *For a linear model $f_\theta(z) = z\theta$ parameterized by $\theta \in \mathbb{R}^{d\times m}$, assuming (without loss of generality) that $\forall i$, $\|z_i\| \leq 1$, $h(\theta)$ is a 1-smooth convex function.*

To prove that the loss function $h(\theta)$ is a smooth, convex function for a linear model, we define $\zeta_i(\theta)$ such that,

$$h(\theta) = \frac{1}{N}\sum_{i=1}^{N} \zeta_i(\theta) \text{ where } \zeta_i(\theta) := \frac{1}{2}\min_{q\in C_i} \|z_i\theta - q\|^2 = \frac{1}{2}\, d^2(z_i\theta, C_i)\,, \tag{B.15}$$

where $d^2(z_i\theta, C_i)$ represents squared Euclidean distance of a point $z_i\theta$ from set C_i and C_i represents the set of feasible cost vectors (defined in Eq. (2.1)) for corresponding solution x_i^*.

We will prove that for an arbitrary z, $\zeta(\theta) := \frac{1}{2}\min_{q\in C} \|z\theta - q\|^2$ is 1-smooth and convex. This will prove the desired statement since the mean of 1-smooth convex functions is 1-smooth and convex.

For this, we define

$$\psi(\theta) := \min_{c\in C} \|z\theta - c\| = d(z\theta, C)\,, \tag{B.16}$$

where $d(z\theta, C)$ represents Euclidean distance of a point $z\theta$ from set C and C represents the set of feasible cost vectors (defined in Eq. (2.1)) for corresponding solution x^*. We will first prove that $\psi(\theta)$ is 1-Lipschitz and convex and use that to prove the smoothness and convexity of $\zeta(\theta)$.

Lemma B.3.1. *For $\|z\| \leq 1$, loss $\psi(\theta)$ is 1-Lipschitz.*

Proof. : We first prove that the projection of a point onto a closed convex set C is non-expansive. Consider two arbitrary points $x, y \in \mathbb{R}^d$ for this. Now, for a point $p \in C$ s.t. p is the projection of y on C, we know that $d(y, C) = d(y, p)$ and $d(x, C) \leq d(x, p)$.

$$\begin{aligned} & d(x, C) \leq d(x, p) \leq d(x, y) + d(y, p) = d(x, y) + d(y, C) && \text{(Triangle inequality)} \\ \implies & d(x, C) - d(y, C) \leq d(x, y) && \text{(B.17)} \end{aligned}$$

Similarly, now consider point $q \in C$ s.t. q is the projection of x on C, we know $d(x, C) = d(x, q)$ and $d(y, C) \leq d(y, q)$. For the same reason,

$$d(y, C) \leq d(y, q) \leq d(y, x) + d(x, C) = d(y, x) + d(x, C) \tag{B.18}$$

$$\implies d(y, C) - d(x, C) \leq d(x, y) \tag{B.19}$$

As, $d(y, x) = d(x, y)$, from Eq. (B.17), Eq. (B.19), we get $|d(y, C) - d(x, C)| \leq d(x, y) = \|x - y\|$. Thus, the projection to a convex closed set C is non-expansive.

Now consider two points θ_1, θ_2 for function ψ,

$$\begin{aligned}
\|\psi(\theta_1) - \psi(\theta_2)\| &= \|d(z\theta_1, C) - d(z\theta_2, C)\| && \text{(By definition of } \psi) \\
&\leq d(z\theta_1, z\theta_2) && \text{(From the above relation)} \\
&= \|z\theta_1 - z\theta_2\| && \text{(B.20)} \\
&\leq \|z\|\|\theta_1 - \theta_2\| && \text{(Cauchy-Schwartz)} \\
&\leq 1\|\theta_1 - \theta_2\| && \text{(Since } \|z\| \leq 1 \text{ by assumption)}
\end{aligned}$$

□

Lemma B.3.2. *Loss function $\psi(\theta)$ is convex.*

Proof. Consider two parameter values θ_1, θ_2 and let the projection of $z\theta_1$, $z\theta_2$ on C be c_1, c_2 respectively. Additionally, consider θ_3 to be the convex combination of θ_1, θ_2 i.e. $\theta_3 := \lambda\theta_1 + (1 - \lambda)\theta_2$ for a arbitrary $\lambda \in (0, 1)$.

Now, we can write:

$$\begin{aligned}
\lambda\psi(\theta_1) + (1 - \lambda)\psi(\theta_2) &= \lambda\|z\theta_1 - c_1\| + (1 - \lambda)\|z\theta_2 - c_2\| && \text{(B.21)} \\
&\geq \|\lambda(z\theta_1 - c_1) + (1 - \lambda)(z\theta_2 - c_2)\| && \text{(Triangle Inequality)} \\
&= \|\lambda\, z\theta_1 + (1 - \lambda)\, z\theta_2 - \lambda c_1 - (1 - \lambda)c_2\| && \text{(B.22)} \\
&= \|\lambda z\theta_1 + (1 - \lambda)z\theta_2 - c\| && (c := \lambda c_1 + (1 - \lambda)c_2) \\
&= \|z\theta_3 - c\| && \text{(By definition of } \theta_3) \\
&= d(z\theta_3, c) \geq d(z\theta_3, C) && \text{(Since } C \text{ is convex, } c \in C) \\
&= \psi(\theta_3) && \text{(By definition of } \psi) \\
\implies \lambda\psi(\theta_1) + (1 - \lambda)\psi(\theta_2) &\geq \psi(\theta_3) && \text{(B.23)} \\
\implies \lambda\psi(\theta_1) + (1 - \lambda)\psi(\theta_2) &\geq \psi(\lambda\theta_1 + (1 - \lambda)\theta_2) && \text{(B.24)}
\end{aligned}$$

Thus, the function ψ is convex from the definition of convexity. □

Lemma B.3.3. *For $\|z\| \leq 1$, loss $\zeta(\theta)$ is 1-smooth.*

Proof. Consider two parameter values θ_1, θ_2 and denote the projection of $z\theta_1$ and $z\theta_2$ on C as c_1 and c_2 respectively. Function ζ is L-smooth if its gradient is Lipschitz continuous with constant L.

$$\begin{aligned}
\|\nabla\zeta(\theta_1) - \nabla\zeta(\theta_2)\| &= \left\|\nabla\left(\frac{1}{2}\|z\theta_1 - c_1\|^2\right) - \nabla\left(\frac{1}{2}\|z\theta_2 - c_2\|^2\right)\right\| && \text{(By definition of } \zeta) \\
&= \|z^T(z\theta_1 - c_1) - z^T(z\theta_2 - c_2)\| && \text{(B.25)} \\
&\leq \|z\|\|d(z\theta_1, C) - d(z\theta_2, C)\| && \text{(Cauchy Schwarz)} \\
&\leq \|\psi(\theta_1) - \psi(\theta_2)\| \\
& && \text{(Since } \|z\| \leq 1 \text{ by assumption and by definition of } \psi) \\
&\leq \|\theta_1 - \theta_2\| && \text{(Lemma B.3.1)}
\end{aligned}$$

Thus, the function $\zeta(\theta)$ is 1-smooth. □

Lemma B.3.4. *Loss $\zeta(\theta)$ is convex.*

Proof. Consider a function $g(x) = \frac{1}{2}x^2$ and note that $\zeta(\theta) = g(\psi(\theta))$. From Lemma B.3.2, we know that ψ is convex. $g(x)$ is convex and non-decreasing for $x \in \{\mathbb{R}^+ \cup 0\}$ and $\psi(\theta)$ is always non-negative. The composition of two functions is convex if g is convex and non-decreasing and ψ is convex [Boyd and Vandenberghe, 2004]. Thus, the composite function ζ is convex. □

B.4 Proof for Proposition 3.1.1

Proposition 3.1.1. *Point $\hat{c} := (c_1, c_2, \ldots, c_N)$ where $c_i = z_i\tilde{\theta}$ and $\tilde{\theta} \in \arg\min h(\theta)$ lies in the intersection $\mathcal{C} \cap \mathcal{F}$ if it exists, else $\hat{c} \in \mathcal{F}$ is the point closest to $\mathcal{C}$.*

Proof. Loss $h(\theta)$ is defined as:

$$h(\theta) = \frac{1}{2N}\sum_{i=1}^{N} \min_{q_i \in C_i} \|f_\theta(z_i) - q_i\|^2 \tag{B.26}$$

where C_i represents the set of feasible cost vectors (defined in Eq. (2.1)) for data-point i. We assume f_θ is a linear model for this proof for which the function $h(\theta)$ is convex.

In order to better interpret $h(\theta)$, consider a point $c_\theta = (c_1, c_2, \ldots, c_N) \in \mathbb{R}^{Nm}$ such that $c_i = f_\theta(z_i)$. Since $c_\theta \in \mathcal{F}$, $h(\theta) = \frac{d^2(c_\theta, \mathcal{C})}{N}$ where $d^2(w, \mathcal{W})$ is the squared Euclidean distance of point w to the set $\mathcal{W}$. Hence, minimizing $h(\theta)$ is related to minimizing the distance between the sets $\mathcal{F}$ and $\mathcal{C}$.

Consequently, our loss can be reformulated as:

$$h(\theta) = \frac{1}{2N} d(c_\theta, \mathcal{C})^2 \tag{B.27}$$

Let us denote $\tilde{\theta} \in \arg\min h(\theta)$ and the predicted point as $\hat{c} := Z\,\tilde{\theta} \in \mathcal{F}$. Therefore, $h(\tilde{\theta}) = \frac{1}{2N} d(\hat{c}, \mathcal{C})^2$.

We can prove the proposition by contradiction. Assume, $\hat{c} := Z\,\tilde{\theta} \in \mathcal{F}$ and is not the closest point to set $\mathcal{C}$. Conversely, assume the closest point to the set $\mathcal{C}$ in $\mathcal{F}$ is given by $Z\,\theta_p$. Now, the loss $h(\theta_p) = \frac{1}{2N} d(Z\theta_p, \mathcal{C})^2$ and since, $Z\theta_p$ is the closest point in the set $\mathcal{C}$ in $\mathcal{F}$, this means, that θ_p is also the arg min of $h(\theta)$. As $\tilde{\theta}$ is also the arg min, this implies that $h(\tilde{\theta}) = h(\theta_p)$, Thus, $d(Z\tilde{\theta}, \mathcal{C}) = d(Z\theta_p, \mathcal{C})$. This implies that minimizing $h(\theta)$ leads to convergence to a point $\hat{c} = Z\tilde{\theta}$, which is the closest distance to $\mathcal{C}$. In the case where an intersection exists, the closest distance to $\mathcal{C} = 0$; therefore, it converges in $\mathcal{F} \cap \mathcal{C}$.

□

B.5 Proof of Proposition 3.3.1

Proposition 3.3.1. *For a linear model $f_\theta(z) = z\theta$, the iterates corresponding to the preconditioned gradient method on $h(\theta)$ with $\eta = 1$ are identical to Algorithm 1.*

Proof. Consider iteration t, and the current value of model parameters at iteration t is denoted by θ_t. Let us denote the updated parameter at time $t+1$ as θ_{t+1}(POCS) and θ_{t+1}(ERM) for POCS update and first order pre-conditioned update on $h(\theta)$ respectively.

Let $q_t = \mathcal{P}_\mathcal{C}(Z\theta_t)$ denote the projection of $Z\theta_t$.

In POCS, the $\theta_{t+1} = \arg\min_\theta \frac{1}{2N} \|Z\theta - q_t\|^2$. Solving it exactly gives the projection of q_t onto the set $\mathcal{F}$, which can be written as:

$$\mathcal{P}_F(q_t) = Z\theta_{t+1} \text{ s.t. } \theta_{t+1} = (Z^T Z)^{-1} Z^T q_t \tag{B.28}$$

Thus, $\theta_{t+1}(\text{POCS}) = (Z^T Z)^{-1} Z^T q_t$

Considering first order update for function $h(\theta)$ with step-size η and pre-conditioner $\left[\frac{Z^T Z}{N}\right]^{-1}$ can be written as:

$$\theta_{t+1} = \theta_t - \eta \left[\frac{Z^T Z}{N}\right]^{-1} \nabla h(\theta_t) \qquad \text{(from definition of preconditioner update)}$$

where, $\nabla h(\theta_t) = \frac{1}{N} Z^T (Z\theta_t - q_t)$.

Now, putting the values back in preconditioned update, we get the following:

$$\theta_{t+1} = \theta_t - \eta[Z^T Z]^{-1}(Z^T(Z\theta_t - q_t)) \tag{B.29}$$
$$= \theta_t(1-\eta) + [Z^T Z]^{-1} Z^T q_t \tag{B.30}$$
$$= (Z^T Z)^{-1} Z^T q_t \qquad \text{(for } \eta = 1\text{)}$$

Thus, $\theta_{t+1}(\text{ERM}) = (Z^T Z)^{-1} Z^T q_t$. Therefore, we can see that iterates produced by POCS is equivalent to 1-step of preconditioned gradient update with $\eta = 1$. □

B.6 Proof for Proposition 3.2.2

Proposition 3.2.2. *For a linear model $f_\theta(z) = z\theta$ and assuming (i) (without loss of generality) that $\forall i$, $\|z_i\| \leq 1$ and (ii) $\lambda_{\min}[Z^T Z] > 0$, $h(\theta)$ is not necessarily strongly-convex but satisfies the PL inequality with $\mu = \lambda_{\min}\left[\frac{\sum_{i=1}^N z_i z_i^\top}{N}\right]$.*

We prove that $h(\theta)$ is not necessarily a strongly convex function by contradiction. Let us assume that $h(\theta)$ is α-strongly convex function with $\alpha > 0$. From the definition of α-strong convexity, $h(\theta)$ must satisfy this following inequality for all θ_1, θ_2.

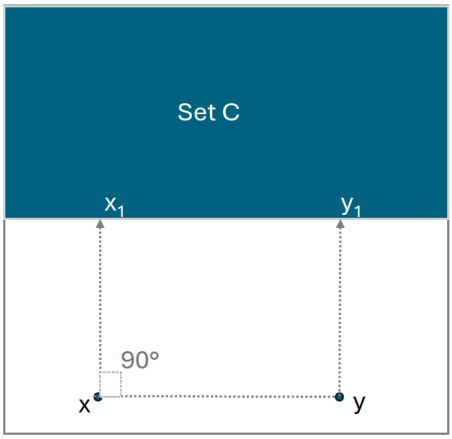

Figure B.1: We can see two point x, y and their projection onto a linear boundary of set C denoted as x_1, y_1 respectively. Moreover, the angle between x, y and x, x_1 is the right angle; thus, the two vectors are orthogonal.

$$h(\theta_1) \geq h(\theta_2) + (\nabla h(\theta_2))^T(\theta_1 - \theta_2) + \frac{\alpha}{2}\|\theta_1 - \theta_2\|^2 \tag{B.31}$$

Consider a special case where $N = 1$ point and $m = d = 1$ and $z = 1$. Let $y = z\theta_1 = \theta_1$ and $x = z\theta_2 = \theta_2$. Consider C as an affine set and two points (x, y) equidistant from C. Define x_1, y_1 to be the projection of x and y onto C respectively (refer to Fig. B.1 above).

Since x and y are equidistant from C, $\|x - x_1\| = \|y - y_1\|$. Moreover, since x and y are on the same side of C, vector $x - y$ is orthogonal vector $x - x_1$. Hence, $\langle y - x, x - x_1\rangle = 0$.

Substituting these values in Eq. (B.31), we get:

$$\frac{1}{2}\|y-y_1\|^2 \geq \frac{1}{2}\|x-x_1\|^2 + \langle x-x_1, y-x\rangle + \frac{\alpha}{2}\|x-y\|^2 \tag{B.32}$$

$$0 \geq \frac{\alpha}{2}\|x-y\|^2 \tag{B.33}$$

$$\implies \alpha = 0 \tag{B.34}$$

As $\alpha = 0$ therefore the function $h(\theta)$ is not strongly convex for this case when C is an affine set. However, we can show that function $h(\theta)$ satisfies the PL inequality with $\mu = \frac{\lambda_{\min}[Z^T Z]}{N}$. Recall, $h(\theta)$ is defined as $\frac{1}{2N}\|Z\theta - q\|^2$ where $q = \mathcal{P}_{\mathcal{C}}(Z\theta)$ is projection of $Z\theta$ on $\mathcal{C}$. Hence, $\nabla h(\theta) = \frac{1}{N} Z^T[Z\theta - q]$.

$$\begin{aligned}
\|\nabla h(\theta)\|^2 &= \frac{1}{N^2}\left\|Z^T[Z\theta - q]\right\|^2 && \text{(By definition of } \nabla h(\theta)) \\
&\geq \frac{1}{N^2}\sigma^2_{\min}(Z^T)\,\|[Z\theta - q]\|^2 && (\|Ax\| \geq \sigma_{\min}(A)\,\|x\|) \\
&= \frac{1}{N^2}\lambda_{\min}(Z^T Z)\,\|[Z\theta - q]\|^2 && (\text{using } \sigma^2_{\min}(Z^T) = \lambda_{\min}(Z^T Z)) \\
&= \frac{2}{N}\lambda_{\min}(Z^T Z)\,h(\theta) && \text{(By definition of } h) \\
&\geq \frac{2}{N}\lambda_{\min}(Z^T Z)[h(\theta) - h(\theta^*)] && \text{(Since } h \text{ is non-negative)} \\
&= \frac{2}{N}\lambda_{\min}\left(\sum_{i=1}^{N} z_i z_i^\top\right)[h(\theta) - h(\theta^*)] && (\text{replacing } Z^T Z \text{ as } \textstyle\sum_{i=1}^N z_i z_i^\top) \\
&= 2\mu\,[h(\theta) - h(\theta^*)] && \left(\text{For } \mu = \lambda_{\min}\left[\frac{\sum_{i=1}^N z_i z_i^\top}{N}\right]\right)
\end{aligned}$$

Hence, $h(\theta)$ is μ-PL.

B.7 Sub-optimality proofs

Proposition 4.2.1. *For $c_\theta \in \mathbb{R}^m := f_\theta(z)$, assuming that $\forall j \in [m], [\hat{x}(c_\theta)]_j, x^*_j \in [0,1]$, $\Gamma(\theta,(z,x^*)) \leq \frac{\sqrt{2m\,h(\theta)}}{\delta}$ where $\delta := O(\chi/\sqrt{m})$, χ is the margin and the O notation hides constants that depend on the LP.*

Proof.

$$
\begin{aligned}
\Gamma(\theta,(z,x^*)) &= \left\langle \frac{\mathcal{P}_C(c_\theta)}{\|\mathcal{P}_C(c_\theta)\|}, \hat{x}(c_\theta) - x^* \right\rangle && \text{(By definition)} \\
&\leq \left\langle \frac{\mathcal{P}_C(c_\theta)}{\delta}, \hat{x}(c_\theta) - x^* \right\rangle && \text{(Since by Lemma B.7.1, } \|\mathcal{P}_C(c_\theta)\| \geq \delta) \\
&\leq \langle \frac{\mathcal{P}_C(c_\theta)}{\delta}, \hat{x}(c_\theta) - x^* \rangle + \frac{1}{\delta}\langle c_\theta, x^* - \hat{x}(c_\theta)\rangle \\
& && \text{(By definition of } \hat{x}(c_\theta),\ \langle c_\theta, x^*\rangle \geq \langle c_\theta, \hat{x}(c_\theta)\rangle) \\
&= \frac{1}{\delta}\langle \mathcal{P}_C(c_\theta) - c_\theta, \hat{x}(c_\theta) - x^* \rangle && \text{(rearranging terms)} \\
&\leq \frac{1}{\delta}\|\mathcal{P}_C(c_\theta) - c_\theta\|\|\hat{x}(c_\theta) - x^*\| && \text{(Cauchy-Schwartz)} \\
&\leq \frac{1}{\delta}\|\mathcal{P}_C(c_\theta) - c_\theta\|\sqrt{m} && \text{(From assumption that } \forall j, \hat{x}_j, x^*_j \in [0,1]) \\
&= \frac{1}{\delta}\sqrt{2h(\theta)}\sqrt{m} && \text{(from definition of } h(\theta)) \\
&= \frac{\sqrt{2m\,h(\theta)}}{\delta} && \text{(B.35)}
\end{aligned}
$$

□

Next, we give the lower-bound on the term δ which depends on the margin χ defined in Section 2.3.3.

Lemma B.7.1. *For $c \in C$ defined using a margin χ, $\|c\|_2 \geq \delta := \frac{\chi}{\sqrt{m}} \max_{j\in M} \min\left\{1, \min_{p\in B} \frac{1}{|\tau_{pj}|}\right\}$.*

Proof. We lower-bound $\|c\|_2$ for an arbitrary $c \in C$ using the reduced cost optimality conditions [Luenberger et al., 1984]. For this, we denote B as the basis set defined as $B := \{j \mid x^*_j > 0\}$ and M as the set of indices not in B, i.e. $M = [m] - B$. Let us define a new term $\tau_{ij} = [(A_B)^\dagger A_j]_i$ where A_B represents the columns corresponding to the indices in set B, and $A_B^\dagger$ is the pseudo-inverse of the matrix A_B.

To prove this proposition, we first consider two arbitrary vectors $a, b \in \mathbb{R}^m$ and show that $\|a\|_1 \geq \frac{\chi}{\max_{p=1}^n |b_p|}$ is a *necessary condition* to ensure that $a^T b \geq \chi$. We do this by contradiction: assume $\|a\|_1 \leq \frac{\chi}{\max_{p=1}^m |b_p|}$, but $a^T b \geq \chi$. In this case,

$$
\|a\|_1 \max_{p=1}^{m} |b_p| \leq \chi \implies \|a\|_1 \|b\|_\infty \leq \chi
$$

By Holders inequality, since $a^T b \leq \|a\|_1 \|b\|_\infty$, the above inequality implies that

$$a^T b \leq \chi\,,$$

which is a contradiction. Since $\|a\|_1 \geq \frac{\chi}{\max_{p=1}^m |b_p|}$, it gives us a lower-bound on $\|a\|_1$. Now, we can use this result to find the lower bound on $\|c\|_1$.

In Section B.2, we have shown that KKT conditions are equivalent to reduced cost optimality conditions. We first find the lower bound to satisfy reduced cost inequality for a specific index $j \in [M]$ and then extend the result for all indices in M to find the lower-bound on $\|c\|_1$.

The reduced cost for an index $j \in M$ and margin χ is given by $r(j) := c_j - c_B(A_B)^\dagger A_j$. The reduced costs conditions imply that for all $j \in M$, $r(j) \geq \chi$. In terms of τ, these conditions imply that $c_j - \sum_{p\in B} c_p \tau_{pj} \geq \chi$. Equivalently, for all $j \in M$, $c^T \alpha_j \geq \chi$, where α_j represents the coefficients of c in $r(j)$.

Using the above result to obtain a lower-bound on $\|c\|_1$, we have that,

$$\begin{aligned}
\|c\|_1 &\geq \frac{\chi}{\max_{p=1}^m |(\alpha_j)_p|} && \text{(B.36)}\\
&= \frac{\chi}{\max\{1, \max_{p\in B} |\tau_{pj}|\}} && \text{(substituting the value of } \alpha_j\text{)}\\
&= \chi \,\min\left\{1, \min_{p\in B} \frac{1}{|\tau_{pj}|}\right\} && \text{(rearranging terms)}
\end{aligned}$$

Since we require that the reduced cost condition be satisfied for all $j \in M$, we get that,

$$\|c\|_1 \geq \chi \max_{j\in M} \min\left\{1, \min_{p\in B} \frac{1}{|\tau_{pj}|}\right\}$$

Finally, we use the relation between norms to lower-bound the value of $\|c\|_2$.

$$\begin{aligned}
\delta &:= \min_{c\in C} \|c\|_2 && \text{(B.37)}\\
&\geq \min_{c\in C} \frac{1}{\sqrt{m}} \|c\|_1 && \text{(by norm inequality)}\\
&\geq \frac{\chi}{\sqrt{m}} \max_{j\in M} \min\left\{1, \min_{p\in B} \frac{1}{|\tau_{pj}|}\right\} && \text{(B.38)}
\end{aligned}$$

□

Appendix C

Experimental Results

C.1 Synthetic Experimental Results

We conduct numerical experiments for two LP problems – the shortest path (SP) problem and the fractional Knapsack problems considered in Sun et al. [2023]. For both SP and Knapsack, we generate 100 samples for training, validation and test sets. We used the codebase provided by the Sun et al. [2023] to generate the dataset.

Shortest Path (SP-synth): The Shortest Path problem is defined on a 5×5 grid with $m = 40$ directed edges associated with the ground truth cost-vector $c^* \in \mathbb{R}^m$. Input $z \in \mathbb{R}^d$ with $d = 6$. Thus, $\theta \in \mathbb{R}^{d \times m}$. To make the problem harder, we use the degree$= 4$ in the data-generation process.

Fractional Knapsack: The Fractional Knapsack problem is defined with input $z \in \mathbb{R}^d$ with $d = 5$. We have 10 items with associated cost-vectors, and slack variables are added to convert the problem to standard form, making the dimension $m = 21$. Thus, $\theta \in \mathbb{R}^{d \times m}$. To make the problem harder, we use the degree$= 2$ in the data-generation process with the attacking noise of attack-power$= 3.0$.

Methods and model: For the experiments, we compare both our variants, POCS (Algorithm 1) and ERM (Chapter 3) with GD and show that they have similar performance. We compare our method against against several existing methods, including ST Sahoo et al. [2022], MOM Sun et al. [2023], BB Vlastelica et al. [2019], QPTL Wilder et al. [2019] and SPO+ Elmachtoub and Grigas [2022]. We train all the models in a deterministic setting employing a linear model.

Training Details For Ours (POCS), we solve the regression problem using closed-form solutions obtained through matrix inversion. For the MOM, Ours (ERM) method, we employ the Armijo line search algorithm [Armijo, 1966] with Gradient Descent. For the BB, ST, QPTL, and SPO+ methods, a grid search is used to find the best constant step-size in $\{10, 1, 0.1, 0.01, 0.001, 0.0001\}$. We also did a grid search for λ, regularizer in QPTL and the perturbation weight in BB with $\lambda \in \{100, 10, 1, 0.1, 0.01, 0.001\}$. The optimal settings are determined based on performance on the validation set, and we plot the training and test plot for the best-performing model.

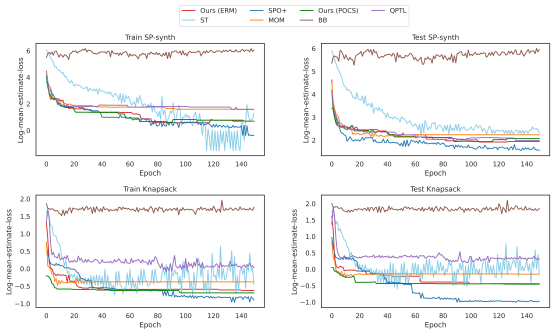

Figure C.1: Estimate loss

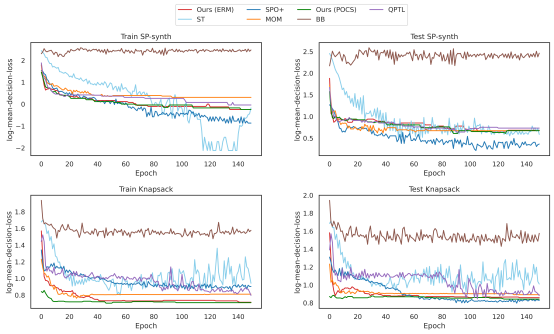

Figure C.2: Decision loss

Figure C.3: Training and Test plot for synthetic tasks. For both problems, our method significantly outperforms the other methods (ST, QPTL, BB, MOM) and is comparable to SPO+, which uses the knowledge of c^*

Results: In Fig. C.2 w.r.t to the decision-loss, all the methods have similar performance except SPO+, BB. For Knapsack, our method outperforms all the other baselines except SPO+. In Fig. C.1, w.r.t to the estimate-loss, our method significantly outperforms the other methods (ST, QPTL, BB, MOM) and is comparable to SPO+, which uses the knowledge of c^*. Moreover, we can see that both our variants, POCS and ERM, have negligible performance differences.

C.2 Additional real-world experiment details

C.2.1 Warcraft shortest Path

In this section, we define the LP for the shortest path problem. Consider x_{ij} represents the edge from vertex i to vertex j. Since all edges are bidirectional, x_{ij} is not the same as x_{ji}. s and t denote source and target vertices, respectively. Let c_{ij} be the cost of selecting edge x_{ij}. Before initializing the LP, let $N(i)$ be the set of vertices with an outgoing edge from vertex i, and $I(i)$ be the set of vertices with an incoming edge to vertex i. The LP can be written as follows:

$$
\begin{aligned}
& \underset{x}{\text{minimize}} && c^T x \\
\text{subject to} \quad & \forall i \notin \{s,t\}, \sum_{j \in N(i)} x_{ij} = \sum_{j \in I(i)} x_{ji} && \text{(flow conservation)} \\
& \sum_j x_{sj} = 1 && \text{(source has one outgoing edge)} \\
& \sum_j x_{jt} = 1 && \text{(target has one incoming edge)} \\
& x \geq 0
\end{aligned}
$$

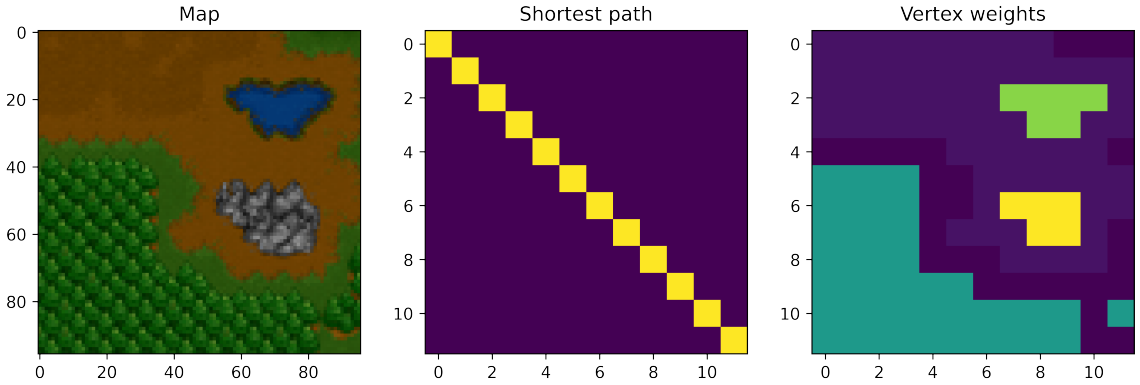

Figure C.4: Warcraft SP dataset sample: The input image (left), ground truth shortest path (center), and the ground truth vertex weights (right). The task is to learn the edge weights to retrieve the same shortest path.

In this case, the source is always in the top-left grid, and the target is in the bottom-right grid. As in the dataset, ground-truth weights are defined on the vertex. To run it as LP defined in Eq. (C.1), we directly predict the weights of the edges. Given that the ground-truth cost $c^*(BB)$ is defined for vertex weights in the dataset, we define c^* for the edge-weighted shortest path as:

$$\forall i,\ \forall j \in N(i),\ c^*_{ij} = c^*_i(BB) \tag{C.1}$$

Both approaches yield the same shortest path, validating the conversion. Please see Vlastelica et al. [2019] for more details regarding the dataset.

C.2.2 Perfect Matching

In this section, we define the LP associated with the perfect matching problem. x_{ij} represents the edge from vertex i to vertex j. $N(i)$ represents the neighbors of vertex i. All the edges in the graph are unidirectional, i.e. x_{ij} and x_{ji} represent the same edge. c_{ij} represent the cost of selecting the edge x_{ij}. Thus, the LP can be written as:

$$\begin{aligned}
\underset{c}{\text{minimize}} \quad & c^T x \\
\text{subject to} \quad & \forall i \sum_{j \in N(i)} x_{ij} = 1 \qquad \text{(each vertex should have exactly one incident edge)} \\
& x \geq 0
\end{aligned}$$

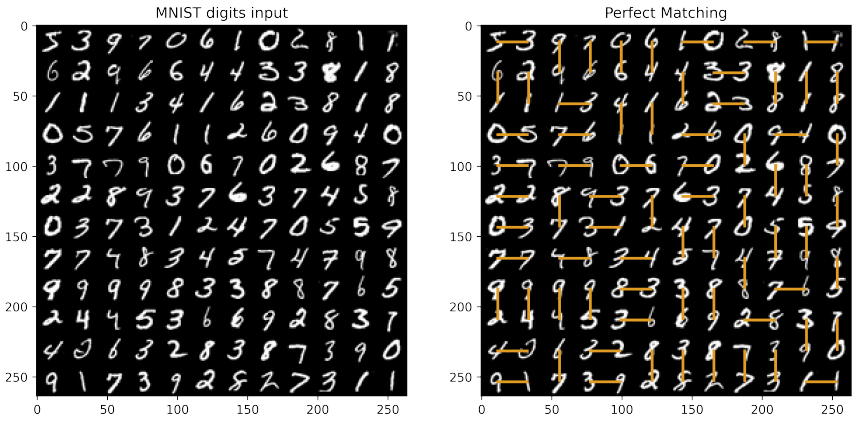

Figure C.5: Perfect Matching dataset sample: This figure shows the input image (left) and the corresponding min-cost perfect matching overlayed on the input image on the right. Each input is a 12×12 grid, with each grid containing an MNIST digit. In Perfect Matching(PM), edges highlighted by the orange lines represent the edge selected by the solving min-cost perfect matching optimization problem. Ground truth edge weights are inferred by reading the digits connected by the edge as a two-digit number. The task is to predict edge weights such that we get the same PM.

In our case, ground-truth edge weights are inferred by reading the digits on the two ends of the vertex as two-digit numbers. Please see Vlastelica et al. [2019] for more details regarding the dataset.

C.3 Runtime Comparison

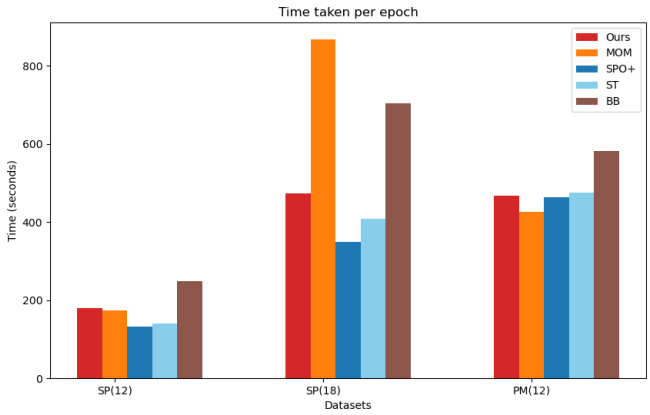

Figure C.6: Training Time (in seconds) per epoch vs method for three real-world experiments. We can see that our method is comparable to other methods and scales well with the dimension of the problem

To benchmark the computational efficiency of our method, we plot the average training time per epoch for all methods in Fig. C.6. Our method is competitive with ST and SPO+, and faster than MOM and BB. BB requires two solver calls per gradient evaluation, while MOM, despite not needing solver calls, involves inverting a matrix A_B, which scales poorly with dimension. As shown in the plot, our method scales well with both the problem dimension and dataset size.

C.4 Ablation study for margin

In order to verify the robustness of Algorithm 1 for varying χ, we do an ablation study varying χ from [10, 0.01] for the synthetic datasets in the deterministic setting (refer to Chapter C for details). We train each model for 150 epochs according to Algorithm 1 and report the mean decision error (Eq. (5.2)) for both the train/test data for the synthetic shortest-path (SP) and Knapsack (K) problems. We see that increasing the margin (from 0.01 to 1) leads to small sub-optimality and that increasing it beyond 1 does not improve the performance.

Margin	Train Sp-synth	Test Sp-synth	Train Knapsack	Test Knapsack
10	0.779	1.95	2.033	2.32
1	0.779	1.89	2.033	2.32
0.1	0.699	2.11	2.08	2.39
0.01	2.63	3.23	2.11	2.32

Table C.1: Ablation results varing margin (χ) from 10 to 0.01 and their performance on train/test set for the synthetic dataset