Inverse Optimization and Inverse Multiobjective Optimization

by

Chaosheng Dong

B.S. in Mathematics and Applied Mathematics, University of Science and Technology of China, 2014

Submitted to the Graduate Faculty of
the Swanson School of Engineering in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2020

UNIVERSITY OF PITTSBURGH

SWANSON SCHOOL OF ENGINEERING

This dissertation was presented

by

Chaosheng Dong

It was defended on

March 25, 2020

and approved by

Bo Zeng, Ph.D., Associate Professor, Department of Industrial Engineering

Jayant Rajgopal, Ph.D., Professor, Department of Industrial Engineering

Oleg A. Prokopyev, Ph.D., Professor, Department of Industrial Engineering

Yiran Chen, Ph.D., Professor, Department of Electrical and Computer Engineering, Duke University

Dissertation Director: Bo Zeng, Ph.D., Associate Professor, Department of Industrial Engineering

Inverse Optimization and Inverse Multiobjective Optimization

Chaosheng Dong, PhD

University of Pittsburgh, 2020

Inverse optimization is a powerful paradigm for learning preferences and restrictions that explain the behavior of a decision maker, based on a set of external signal and the corresponding decision pairs. However, most inverse optimization algorithms are designed specifically in a batch setting, where all data is available in advance. As a consequence, there has been rare use of these methods in an online setting that is more suitable for real-time applications. To change such a situation, we propose a general framework for inverse optimization through online learning. Specifically, we develop an online learning algorithm that uses an implicit update rule which can handle noisy data.

We also note that the majority of existing studies assumes that the decision making problem is with a single objective function, and attributes data divergence to noises, errors or bounded rationality, which, however, could lead to a corrupted inference when decisions are tradeoffs among multiple criteria. We take a data-driven approach and design a more sophisticated inverse optimization formulation to explicitly infer parameters of a multiobjective decision making problem from noisy observations. This framework, together with our mathematical analyses and advanced algorithm developments, demonstrates a strong capacity in estimating critical parameters, decoupling interpretable components from noises or errors, deriving the denoised optimal decisions, and ensuring statistical significance. In particular, for the whole decision maker population, if suitable conditions hold, we will be able to understand the overall diversity and the distribution of their preferences over multiple criteria.

Additionally, we propose a distributionally robust approach to inverse multiobjective optimization. Specifically, we study the problem of learning the objective functions or constraints of a multiobjective decision making model, based on a set of observed decisions. In particular, these decisions might not be exact and possibly carry measurement noises or are generated with the bounded rationality of decision makers. We use the Wasserstein metric

to construct the uncertainty set centered at the empirical distribution of these decisions. We show that this framework has statistical performance guarantees. We also develop an algorithm to solve the resulting minmax problem and prove its finite convergence.

Keywords: inverse optimization, utility estimation, online learning, inverse multiobjective optimization, distributionally robustness, clustering, manifold learning.

Table of Contents

List of Tables

List of Figures

Preface

I would like to thank the many people who have helped me through the completion of my Ph.D. over the past five years. Without the help of these people, this dissertation would not have been possible.

First, I would like to acknowledge my advisor, Dr. Bo Zeng, who provided me with the academic, emotional, and personal support I needed to accomplish my Ph.D. over the past five years. Many of Zeng's great spirits, such as honestness, resilience, hardworking, and dedication to science, have a huge impact on me and shape part of me. Particularly, Zeng's enthusiasm for research is infectious and I am in awe of his sharpness and insight as a researcher. I thank Zeng for his great generosity with his time, wisdom, and invaluable advice with my research and my career in general. I am eternally grateful to him for his mentorship and kindness.

I also would like to thank the other members of my committee - Dr. Jayant Rajgopal, Dr. Oleg Prokopyev and Dr. Yiran Chen - who dedicated time to helping me complete this dissertation. Each member has a valuable set of skills that they graciously shared throughout the dissertation process and I thank them. I feel grateful to have had such a supportive committee, and their continued advice and mentorship is valuable to me.

Among the many amazing people in IE, I especially thank my fellow past and present graduate students, Ruichen Sun, Yanfei Chen, Shan Gong, Imrul Kayes, Wei Wang, Liang Xu, Zhaohui Geng, Yuwen Yang, Xueyu Shi, Ke Ren, Jing Yang, Shadi Sanoubar, Tarik Bilgiç, Shaoning Han, Junfeng Gao, Ibrahim El Shar, Dhaifallah Alghamdi, Kamal Basulaiman, and many others. My thanks also go to many friends at the Cost Sports Center who have kept me active and reasonably fit. I would also like to thank Dr. Jeffrey P. Kharoufeh and Dr. Paul W. Leu from whom I draw inspriation on how to become a good instructor.

Finally, I would like to express my thanks and love to my family. I thank my parents and sister who have unconditionally supported me all these years. I thank my wife, Yijia Wang, for her loving support, understanding, and encouragement. It is a miracle that we found each other in Pitt. Being with her has made this journey fun and much more meaningful.

1.0 Introduction

In business and management practice, humans, enterprises and organizations keep making various decisions. A common assumption made is that these decision makers are rational, i.e., they acquire and carry out optimal decisions in their decision making problems. These decision making problems can be classified as the single objective optimization problem and the multiobjective optimization problem. The fundamental problem we consider is how to learn these decision making schemes. Based on the type of the decision making problem, the research problems we study generally consist of two parts: inverse optimization problem and inverse multiobjective optimization problem. Specifically, we consider interactions between inverse optimization, inverse multiobjective optimization, and other research topics, such as online learning, clustering, manifold learning, and robust optimization, in the area of machine learning and optimization. We summarize these interactions in Figure 1.

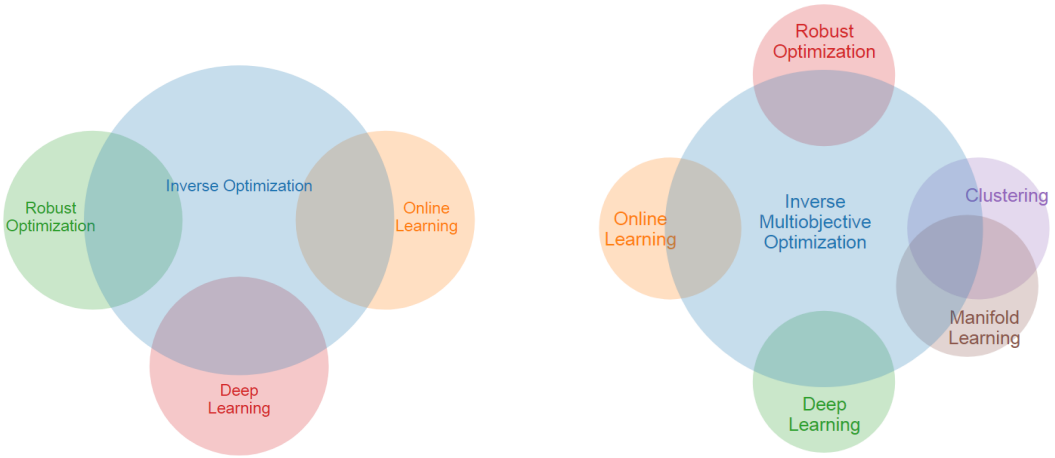


Figure 1: An overview of the topics we consider in our research.

1.1 Inverse Optimization through online learning

Possessing the ability to elicit customers' preferences and restrictions (PR) is crucial to the success for an organization in designing and providing services or products. Nevertheless, as in most scenarios, one can only observe their decisions or behaviors corresponding to external signals (e.g., prices, assortments) while cannot directly access their decision making schemes. Indeed, decision makers probably do not have exact information regarding their own decision making process [1]. To bridge that discrepancy, inverse optimization has been proposed and received significant research attention, which is to infer or learn the missing information of the underlying decision models from observed data, assuming that human decision makers are rationally making decisions [2, 3, 4, 5, 1, 6, 7, 8, 9, 10, 11]. Nowadays, extending from its initial form that only considers a single observation [2, 3, 4, 5] with clean data, inverse optimization has been further developed and applied to handle more realistic cases that have many observations with noisy data [1, 6, 7, 9, 10, 11].

Despite of these remarkable achievements, traditional inverse optimization (typically in a batch setting) has not proven fully applicable for supporting recent attempts in AI to automate the elicitation of human decision maker's PR in real time. Consider, for example, recommender systems (RSs) used by online retailers to increase product sales. The RSs first elicit one customer's PR from the historical sequence of her purchasing behaviors, and then make predictions about her future shopping actions. Indeed, building RSs for online retailers is challenging because of the sparsity issue. Given the large amount of products available, one customer's shopping vector, each element of which represents the quantity of one product purchased, is highly sparse. Moreover, the shift of the customer's shopping behavior along with the external signal (e.g., price, season) aggravates the sparsity issue. Therefore, it is particularly important for RSs to have access to large data sets to perform accurate elicitation [12]. Considering the complexity of the inverse optimization problem (IOP), it will be extremely difficult and time consuming to extract user's PR from large, noisy data sets using conventional techniques. Thus, incorporating traditional inverse optimization into RSs is impractical for real time elicitation of user's PR.

To automate the elicitation of human decision maker's PR, we aim to unlock the potential of inverse optimization through online learning. Specifically, we formulate such learning problem as an IOP considering noisy data, and develop an online learning algorithm to derive unknown parameters occurring in either the objective function or constraints. At the heart of our algorithm is taking inverse optimization with a single observation as a subroutine to define an implicit update rule. Through such an implicit rule, our algorithm can rapidly incorporate sequentially arrived observations into this model, without keeping them in memory. Indeed, we provide a general mechanism for the incremental elicitation, revision and reuse of the inference about decision maker's PR.

To the best of authors' knowledge, we propose the first general framework for eliciting decision maker's PR using inverse optimization through online learning. This framework can learn general convex utility functions and constraints with observed (signal, noisy decision) pairs. In Figure 2, we provide the comparison of inverse optimization through batch learning versus through online learning. Moreover, we prove that the online learning algorithm, which adopts an implicit update rule, has a $\mathcal{O}(\sqrt{T})$ regret under certain regularity conditions. In addition, this algorithm is statistically consistent when the data satisfies some rather common conditions, which guarantees that our algorithm will asymptotically achieves the best prediction error permitted by the inverse model we consider. Numerical results show that our algorithm can learn the parameters with great accuracy, is robust to noises even if some assumptions do not hold, and achieves a dramatic improvement over the batch learning approach on computational efficacy.

1.2 Inverse Multiobjective Optimization

Human decision makers are often confronted with multiple objectives when making decisions. For example, comfort and cost are two often conflicting criteria when customers make purchases [15]. Actually, in economics, science and engineering, the multiobjective decision making problem (DMP) is quite common and many decision making processes naturally involve multiple conflicting objectives [16]. These underlying multiobjective decision making

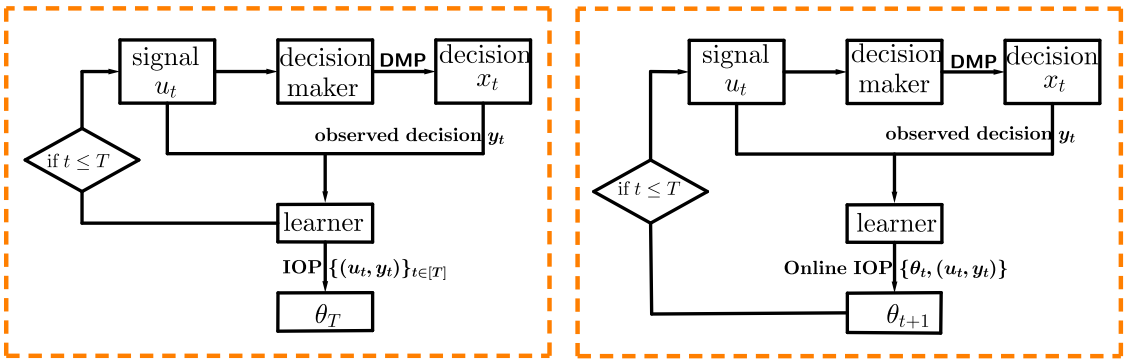


Figure 2: An overview of inverse optimization through batch learning versus through online learning. **Left:** Framework of inverse optimization in the batch setting. **Right:** Framework of the generalized inverse optimization in the online setting proposed.

schemes, once learned by artificial intelligence (AI) system, would presumably assist and accelerate the human expert's decision making process, such as supporting organizations in designing products or in providing services to customers.

Mathematically, we observe a set of decisions $\{\mathbf{y}_i\}_{i\in[N]}$, where each $\mathbf{y}_i$ with $i \in [N]$ is an observation of the Pareto optimal solution of the multiobjective optimization problem

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & \{f_1(\mathbf{x},\theta), f_2(\mathbf{x},\theta), \ldots, f_p(\mathbf{x},\theta)\} \\ s.t. \quad & \mathbf{x} \in X(\theta), \end{aligned}$$

where $\theta \in \mathbb{R}^n$ is the true but unknown parameter for the expert's multiobjective decision making problem. Formally, we investigate the following fundamental question

$$\textit{how do we learn } \theta \textit{ given } \{\mathbf{y}_i\}_{i\in[N]}?$$

This question naturally occurs in many settings. For example, a portfolio manager typically uses the Markovitz mean-variance model to make investment decisions [17]. One analyst might be interested in learning the key parameter of this model, e.g., the expected returns of the assets, by observing the portfolio manager's historical investment records. To learn the parameter θ, we formulate an inverse multiobjective optimization problem (IMOP), assuming that human expert is rational or bounded rational. Given the fact that the learner often only

Table 1: Comparisons between IOP and IMOP from the machine learning point of view.

Task	# of obj	Signal	Supervised learning	Paper
IOP	single	yes	yes	[1, 13, 7] [10, 6, 14]
IMOP	multiple	no	no	this dissertation

has access to human expert's decisions and no other data, we show that IMOP essentially is an unsupervised learning task.

We reveal connections between IMOP and two seemingly unrelated unsupervised learning problems. The first one is the K-means clustering problem [18, 19, 20]. Specifically, we prove that every K-means clustering problem can be transformed equivalently to an IMOP. As solving K-means clustering problem is NP-hard [21, 22] even when $n = 2$, we thus show the NP-hardness of IMOP. Furthermore, we note that solving IMOP will automatically assign the observations to different clusters, while the centroids of these clusters are restricted to be Pareto optimal solutions of the estimated DMP. Hence, IMOP can be interpreted as a constrained K-means clustering problem [23].

The second one is the manifold learning problem, which constructs low-dimensional manifolds from data points embedded in high-dimensional spaces [24, 25]. We note that the Pareto optimal set is a piecewise continuous manifold with an intrinsic dimension of $p - 1$ (p is the number of objective functions) under suitable smoothness conditions, regardless of the dimension of the decision space. Since the dimension of the decision space is usually much larger than p, the Pareto optimal set is a low-dimensional manifold embedded in the ambient decision space. Moreover, given that solving IMOP is to construct a DMP whose Pareto optimal set closely matches observations, IMOP can also be interpreted as a manifold learning problem.

1.3 Wasserstein Distributionally Robust IMOP

Inverse multiobjective optimization is a compelling tool for learning humans and robots' behaviors and preferences. Similar to the extensively studied area of inverse optimization where the decision making model consists of only one objective function [2, 1, 13, 7, 10, 26, 14], the performance of inverse multiobjective optimization also relies critically on the availability of an accurate decision making model, sufficient decisions of high quality, and a parameter space that contains as much information about the objective functions or constraints of the underlying decision making model as possible. In practice, however, it is highly unlikely that all of these critical factors would be satisfied. Consider, for example, presence of outliers in a limited amount of decisions would render the empirical distribution of decisions deviate from the true distribution, and thus significantly weaken the predictive power of the inverse multiobjective optimization estimator.

To hedge against these uncertainties contained in the hypothetical decision making model, the data and the selected parameter space, we investigate the distributionally robust approach for inverse multiobjective optimization. More specifically, motivated by [27, 28, 29, 30], etc., we use the Wasserstein metric [31] to construct the uncertainty set centered at the empirical distribution of the observed decisions. Subsequently, we propose a distributionally robust inverse multiobjective optimization program that minimizes the worst-case risk of loss, where the worst case is taken over all distributions in the uncertainty set. Through such a distributionally robust framework, we aim to bridge the discrepancy between lack of certainties in the information and the expectation for the accurate prediction of human or robot's future behavior.

We present a novel Wasserstein distributionally robust framework for constructing inverse multiobjective optimization estimator. Our research is motivated by the fast developing area of distributionally robust optimization [27, 32, 33, 34]. We use the prominent Wasserstein metric to construct the uncertainty set centered at the empirical distribution of observed decisions. Moreover, by employing recent minmax statistical learning results [35], we show that the proposed framework has statistical performance guarantees, and the excess risk of the distributionally robust inverse multiobjective optimization estimator would converge to

zero with a sub-linear rate as the number of observed decisions approaches to infinity. To solve the resulting minmax problem, we reformulate it as a semi-infinite program and develop a cutting-plane algorithm which converges to an approximate solution in finite iterations. We demonstrate the effectiveness of our method on both a multiobjective quadratic program and a portfolio optimization problem.

2.0 Generalized Inverse Optimization through Online Learning

2.1 Literature Review

Up to now, many studies on parameter estimations through inverse optimization have been designed and developed, where almost all of them assume that the underlying DMP is of a single objective function. According to the model development and the treated observations, they can be classified into four groups, i.e., inverse optimization with (i) a single observation without noise, (ii) a single observation subject to noise, (iii) multiple observations without noise, and (iv) multiple observations subject to noises.

In the first group, structured inverse network and combinatorial optimization problems are probably the first set of IOP studies in the literature, where costs of individual arcs are estimated to render the given solution (e.g., network flows, paths, spanning trees) optimal [36, 37, 38, 39, 40, 41, 42]. General linear programming IOP with a single observation is investigated in the seminal paper by [2], where the distance between the estimated objective function, to which the observation is an optimal solution, and a nominal objective function serves as the loss function. this dissertation shows that its IOP using L_1 or L_∞ norm is also a linear program. This research is then further extended to study IOPs of more general decision making schemes, including inverse conic problems [3], inverse optimization for linearly constrained convex separable programming problems [43], constrained inverse quadratic programming problems [44], and inverse integer programming problems [4, 5]. In addition, [45] considered the problem of inverse reinforcement learning that seeks to extract a reward function given optimal behavior in a Markov decision process.

Different from studies in group one that assume the observation is an optimal decision, which is rather restrictive in practice, IOP studies in the second group allow the observation to be noisy. To the best of our knowledge, the research in [46] probably produces the first general study considering noisy observation. They adopt the bilevel optimization to construct an IOP, where the lower level problem receives the utility function estimation and generates an optimal solution of the underlying DMP, and the upper level problem is to

determine a utility function that minimizes the distance between that optimal solution and the noisy observation. The research in [47] analyzes a similar linear programming IOP for a noisy observation, where closed form solutions for several special cases are derived with clear geometric intuitions. Actually, we point out that, although implicitly, the popular O-D matrix estimation problem that in fact is an IOP, has also been treated traditionally as a bilevel model, e.g., [48]. Hence, similar to [7], we believe that bilevel optimization scheme probably provides the most appropriate modeling tool to connect the inference intention and the underlying DMP.

Studies of IOP in the third group extend to consider multiple optimal observations, which can been found in the research on model predictive control (MPC) [49, 50, 51, 52, 53]. In this context, a control law, which might be a piecewise function with each piece representing an optimal solution over a region in a polyhedral partition of the parameter space, will be used to recover parameters of the underlying DMP. Note that multiple pieces of that function, which are treated as multiple optimal observations, should be considered simultaneously in the associated IOP mode [50, 51, 52, 53].

The research of IOP in the fourth group, which takes the data-driven approach to directly consider multiple noisy observations, recently has received a substantial attention [54, 1, 13, 7, 10]. In [54], an IOP formulation that minimizes the *decisional regret*, which is the value differences between observed decisions and expected solutions associated with the cost estimation, is developed and then is illustrated for cost estimation in production planning. The research in [1] presents an IOP framework to impute a convex objective function by minimizing the residuals of Karush-Kuhn-Tucker (KKT) conditions incurred by noisy data. Similarly, an inverse variational inequalities problem, which is a more general scheme, is introduced in [13], noting that solutions of an optimization problem can be represented as solutions to a set of variational inequalities. Then, parameter estimation is derived to minimize the slackness needed to render observations to (approximately) satisfy those variational inequalities. We mention that in [7] a bilevel optimization based IOP that minimizes the differences between observations and expected optimal solutions is introduced, whose, for the first time, statistical consistency properties with respect to noisy observations are systematically analyzed and established. In the most recent paper [10], the authors propose

to adopt the suboptimality loss in IOP, which has a clear advantage in the computational tractability over that in [7], and formulate a distributionally robust IOP model to achieve some out-of-sample guarantees.

2.2 Problem Setting

2.2.1 Decision Making Problem

We consider a family of parameterized decision making problems, in which $\mathbf{x} \in \mathbb{R}^n$ is the decision variable, $u \in \mathcal{U} \subseteq \mathbb{R}^m$ is the external signal, and $\theta \in \Theta \subseteq \mathbb{R}^p$ is the parameter.

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & f(\mathbf{x}, u, \theta) \\ s.t. \quad & \mathbf{g}(\mathbf{x}, u, \theta) \leq \mathbf{0}, \end{aligned} \tag{2.1}$$

where $f : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \mapsto \mathbb{R}$ is a real-valued function, and $\mathbf{g} : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \mapsto \mathbb{R}^q$ is a vector-valued function. We denote $X(u, \theta) = \{x \in \mathbb{R}^n : \mathbf{g}(\mathbf{x}, u, \theta) \leq \mathbf{0}\}$ the feasible region of 2.1. We let $S(u, \theta) = \arg\min\{f(\mathbf{x}, u, \theta) : x \in X(u, \theta)\}$ be the optimal solution set of 2.1.

Throughout this dissertation we assume that the signal-decision pair $(u, \mathbf{x})$ is distributed according to some unknown distribution $\mathbb{P}$ supported on $\{(u, \mathbf{x}) : u \in \mathcal{U}, \mathbf{x} \in X(u, \theta)\}$.

2.2.2 Inverse Optimization and Online Setting

Consider a learner who monitors the signal $u \in \mathcal{U}$ and the decision maker' decision $\mathbf{x} \in X(u, \theta)$ in response to u. We assume that the learner does not know the decision maker's utility function or constraints in 2.1. Since the observed decision might carry measurement error or is generated with a bounded rationality of the decision maker, i.e., being suboptimal, we denote $\mathbf{y}$ the observed noisy decision for $u \in \mathcal{U}$. Note that $\mathbf{y}$ does not necessarily belong to $X(u, \theta)$, i.e., it might be infeasible with respect to $X(u, \theta)$. Throughout the paper, we assume that the (signal,noisy decision) pair $(u, \mathbf{y})$ is distributed according to some unknown distribution $\mathbb{P}$ supported on $\{(u, \mathbf{y}) : u \in \mathcal{U}, \mathbf{y} \in \mathcal{Y}\}$.

In our inverse optimization model, the learner aims to learn the decision maker's objective function or constraints from (signal, noisy decision) pairs. More precisely, the goal of the learner is to estimate the parameter θ of the 2.1. In our online setting, the (signal, noisy decision) pair become available to the learner one by one. Hence, the learning algorithm produces a sequence of hypotheses $(\theta_1, \ldots, \theta_{T+1})$. Here, T is the total number of rounds, and θ_1 is an arbitrary initial hypothesis and θ_t for $t \geq 2$ is the hypothesis chosen after observing the $(t-1)$th (signal,noisy decision) pair. Let $l(\mathbf{y}_t, u_t, \theta_t)$ denote the loss the learning algorithm suffers when it tries to predict the tth decision given u_t based on $\{(u_1, \mathbf{y}_1), \cdots, (u_{t-1}, \mathbf{y}_{t-1})\}$. The goal of the learner is to minimize the regret, which is the cumulative loss $\sum_{t\in[T]} l(\mathbf{y}_t, u_t, \theta_t)$ against the possible loss when the whole batch of (signal,noisy decision) pairs are available. Formally, the regret is defined as

$$R_T = \sum_{t\in[T]} l(\mathbf{y}_t, u_t, \theta_t) - \min_{\theta\in\Theta} \sum_{t\in[T]} l(\mathbf{y}_t, u_t, \theta). \tag{2.2}$$

In the following, we make a few assumptions to simplify our understanding, which are actually mild and frequently appear in the inverse optimization literature [1, 13, 10, 7].

Assumption 2.2.1. Set Θ is a convex compact set. There exists $D > 0$ such that $\|\theta\|_2 \leq D$ for all $\theta \in \Theta$. In addition, for each $u \in \mathcal{U}, \theta \in \Theta$, both $\mathbf{f}(\mathbf{x}, u, \theta)$ and $\mathbf{g}(\mathbf{x}, u, \theta)$ are convex in $\mathbf{x}$.

2.3 Learning the Parameters

2.3.1 The Loss Function

Different loss functions that capture the mismatch between predictions and observations have been used in the inverse optimization literature. In particular, the (squared) distance between the observed decision and the predicted decision enjoys a direct physical meaning, and thus is most widely used [48, 46, 47, 7]. Hence, we take the (squared) distance as our loss function in this dissertation. In batch setting, statistical properties of inverse optimization

with such a loss function have been analyzed extensively in [7]. In this dissertation, we focus on exploring the performance of the online setting.

Given a (signal,noisy decision) pair $(u, \mathbf{y})$ and a hypothesis θ, we define the following loss function as the minimum (squared) distance between $\mathbf{y}$ and the optimal solution set $S(u, \theta)$.

$$l(\mathbf{y}, u, \theta) = \min_{\mathbf{x} \in S(u,\theta)} \|\mathbf{y} - \mathbf{x}\|_2^2. \tag{2.3}$$

2.3.2 Online Implicit Updates

Once receiving the tth (signal,noisy decision) pair $(u_t, \mathbf{y}_t)$, θ_{t+1} can be obtained by solving the following optimization problem:

$$\theta_{t+1} = \arg\min_{\theta \in \Theta} \quad \tfrac{1}{2}\|\theta - \theta_t\|_2^2 + \eta_t l(\mathbf{y}_t, u_t, \theta), \tag{2.4}$$

where η_t is the learning rate in round t, and $l(\mathbf{y}_t, u_t, \theta)$ is defined in (2.3).

The updating rule (2.4) seeks to balance the tradeoff between "conservativeness" and correctiveness", where the first term characterizes how conservative we are to maintain the current estimation, and the second term indicates how corrective we would like to modify with the new estimation. As there is no closed form for θ_{t+1} in general, we call (2.4) an implicit update rule [55, 56].

To solve (2.4), we can replace $\mathbf{x} \in S(u, \theta)$ by KKT conditions of the 2.1, and get a mixed integer nonlinear program. Consider, for example, a decision making problem that is a quadratic optimization problem. Namely, the 2.1 has the following form:

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} \quad & \tfrac{1}{2}\mathbf{x}^T Q \mathbf{x} + \mathbf{c}^T \mathbf{x} \\ s.t. \quad & A\mathbf{x} \geq \mathbf{b}. \end{aligned} \tag{2.5}$$

Suppose that $\mathbf{b}$ changes over time t. That is, $\mathbf{b}$ is the external signal for 2.5 and equals to $\mathbf{b}_t$ at time t. If we seek to learn $\mathbf{c}$, the optimal solution set for 2.5 can be characterized by KKT conditions as $S(\mathbf{b}_t) = \{\mathbf{x} : \mathbf{A}\mathbf{x} \geq \mathbf{b}_t,\ \mathbf{u} \in \mathbb{R}_+^m,\ \mathbf{u}^T(\mathbf{A}\mathbf{x} - \mathbf{b}_t) = 0,\ Q\mathbf{x} + \mathbf{c} - \mathbf{A}^T\mathbf{u} = 0\}$.

Here, $\mathbf{u}$ is the dual variable for the constraints. Then, the single level reformulation of the update rule by solving (2.4) is

$$\begin{aligned} \min_{\mathbf{c}\in\Theta} \quad & \tfrac{1}{2}\|\mathbf{c}-\mathbf{c}_t\|_2^2+\eta_t\|\mathbf{y}_t-\mathbf{x}\|_2^2 \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x}\geq\mathbf{b}_t, \\ & \mathbf{u}\leq M\mathbf{z}, \\ & \mathbf{A}\mathbf{x}-\mathbf{b}_t\leq M(1-\mathbf{z}), \\ & Q\mathbf{x}+\mathbf{c}-\mathbf{A}^T\mathbf{u}=0, \\ & \mathbf{c}\in\mathbb{R}^m,\ \ \mathbf{x}\in\mathbb{R}^n,\ \ \mathbf{u}\in\mathbb{R}_+^m,\ \ \mathbf{z}\in\{0,1\}^m, \end{aligned} \tag{2.6}$$

where $\mathbf{z}$ is the binary variable used to linearize KKT conditions, and M is an appropriate number used to bound the dual variable $\mathbf{u}$ and $\mathbf{A}\mathbf{x}-\mathbf{b}_t$. Clearly, 2.6 is a mixed integer second order conic program (MISOCP). More examples are given in supplementary material.

Our application of the implicit updates to learn the parameter of 2.1 proceeds in Algorithm 1.

Algorithm 1 Implicit Online Learning for Generalized Inverse Optimization

Input: (signal,noisy decision) pairs $\{(u_t,\mathbf{y}_t)\}_{t\in[T]}$
Initialization: θ_1 could be an arbitrary hypothesis of the parameter.
for $t=1$ to T **do**
  receive $(u_t,\mathbf{y}_t)$
  suffer loss $l(\mathbf{y}_t,u_t,\theta_t)$
  if $l(\mathbf{y}_t,u_t,\theta_t)=0$ **then**
    $\theta_{t+1}\leftarrow\theta_t$
  else
    set learning rate $\eta_t\propto 1/\sqrt{t}$
    update $\theta_{t+1}=\arg\min_{\theta\in\Theta}\frac{1}{2}\|\theta-\theta_t\|_2^2+\eta_t l(\mathbf{y}_t,u_t,\theta)$ (solve (2.4))
  end if
end for

Remark 2.3.1. (i) In Algorithm 1, we let $\theta_{t+1} = \theta_t$ if the prediction error $l(\mathbf{y}_t, u_t, \theta_t)$ is zero. But in practice, we can set a threshold $\epsilon > 0$ and let $\theta_{t+1} = \theta_t$ once $l(\mathbf{y}_t, u_t, \theta_t) < \epsilon$. (ii) Normalization of θ_{t+1} is needed in some situations, which eliminates the impact of trivial solutions.

Remark 2.3.2. To obtain a strong initialization of θ in Algorithm 1, we can incorporate an idea in [1], which imputes a convex objective function by minimizing the residuals of KKT conditions incurred by the noisy data. Assume we have a historical data set $\widetilde{T}$, which may be of bad qualities for the current learning. This leads to the following initialization problem:

$$\begin{array}{lll} \min\limits_{\theta\in\Theta} & \frac{1}{|\widetilde{T}|}\sum\limits_{t\in[\widetilde{T}]}\left(r_c^t + r_s^t\right) & \\ \text{s.t.} & |\mathbf{u}_t^T\mathbf{g}(\mathbf{y}_t, u_t, \theta)| \le r_c^t, & \forall t \in \widetilde{T}, \\ & \|\nabla f(\mathbf{y}_t, u_t, \theta) + \nabla\mathbf{u}_t^T\mathbf{g}(\mathbf{y}_t, u_t, \theta)\|_2 \le r_s^t, & \forall t \in \widetilde{T}, \\ & \mathbf{u}_t \in \mathbb{R}_+^m, \;\; r_c^t \in \mathbb{R}_+, \;\; r_s^t \in \mathbb{R}_+, & \forall t \in \widetilde{T}, \end{array} \tag{2.7}$$

where r_c^t and r_s^t are residuals corresponding to the complementary slackness and stationarity in KKT conditions for the t-th noisy decision $\mathbf{y}_t$, and $\mathbf{u}_t$ is the dual variable corresponding to the constraints in 2.1. Note that (2.7) is a convex program. It can be solved quite efficiently compared to solving the inverse optimization problem in batch setting [7]. Other initialization approaches using similar ideas e.g., computing a variational inequality based approximation of inverse model [13], can also be incorporated into our algorithm.

2.3.3 Theoretical Analysis

Note that the implicit online learning algorithm is generally applicable to learn the parameter of any convex 2.1. In this section, we prove that the average regret R_T/T converges at a rate of $\mathcal{O}(1/\sqrt{T})$ under certain regularity conditions. Furthermore, we will show that the proposed algorithm is statistically consistent when the data satisfies some common regularity conditions. We begin by introducing a few assumptions that are rather common in literature [1, 13, 10, 7].

Assumption 2.3.1. (a) For each $u \in \mathcal{U}$ and $\theta \in \Theta$, $X(u,\theta)$ is closed, and has a nonempty relative interior. $X(u,\theta)$ is also uniformly bounded. That is, there exists $B > 0$ such that $\|\mathbf{x}\|_2 \leq B$ for all $\mathbf{x} \in X(u,\theta)$.

(b) $f(\mathbf{x},u,\theta)$ is λ-strongly convex in $\mathbf{x}$ on $\mathcal{Y}$ for fixed $u \in \mathcal{U}$ and $\theta \in \Theta$. That is, $\forall \mathbf{x},\mathbf{y} \in \mathcal{Y}$,

$$\Big(\nabla f(\mathbf{y},u,\theta) - \nabla f(\mathbf{x},u,\theta)\Big)^T(\mathbf{y}-\mathbf{x}) \geq \lambda\|\mathbf{x}-\mathbf{y}\|_2^2.$$

Remark 2.3.3. For strongly convex program, there exists only one optimal solution. Therefore, Assumption 2.3.1.(b) ensures that $S(u,\theta)$ is a single-valued set for each $u \in \mathcal{U}$. However, $S(u,\theta)$ might be multivalued for general convex 2.1 for fixed u. Consider, for example, $\min_{x_1,x_2}\{x_1+x_2 : x_1+x_2 \geq 1\}$. Note that all points on line $x_1+x_2=1$ are optimal. Indeed, we find such case is quite common when there are many variables and constraints. Actually, it is one of the major challenges when learning parameters of a function that's not strongly convex using inverse optimization.

For convenience of analysis, we assume below that we seek to learn the objective function while constraints are known. Then, the performance of Algorithm 1 also depends on how the change of θ affects the objective values. For $\forall \mathbf{x} \in \mathcal{Y}, \forall u \in \mathcal{U}, \forall \theta_1,\theta_2 \in \Theta$, we consider the difference function

$$h(\mathbf{x},u,\theta_1,\theta_2) = f(\mathbf{x},u,\theta_1) - f(\mathbf{x},u,\theta_2). \tag{2.8}$$

Assumption 2.3.2. $\exists \kappa > 0$, $\forall u \in \mathcal{U}, \forall \theta_1,\theta_2 \in \Theta$, $h(\cdot,u,\theta_1,\theta_2)$ is Lipschitz continuous on $\mathcal{Y}$:

$$|h(\mathbf{x},u,\theta_1,\theta_2) - h(\mathbf{y},u,\theta_1,\theta_2)| \leq \kappa\|\theta_1-\theta_2\|_2\|\mathbf{x}-\mathbf{y}\|_2, \forall \mathbf{x},\mathbf{y} \in \mathcal{Y}.$$

Basically, this assumption says that the objectives functions will not change very much when either the parameter θ or the variable $\mathbf{x}$ is perturbed. It actually holds in many common situations, including the linear program and quadratic program.

Lemma 2.3.1. Under Assumptions 2.2.1 - 2.3.2, the loss function $l(\mathbf{y},u,\theta)$ is uniformly $\frac{4(B+R)\kappa}{\lambda}$-Lipschitz continuous in θ. That is, $\forall \mathbf{y} \in \mathcal{Y}, \forall u \in \mathcal{U}, \forall \theta_1,\theta_2 \in \Theta$, we have

$$|l(\mathbf{y},u,\theta_1) - l(\mathbf{y},u,\theta_2)| \leq \frac{4(B+R)\kappa}{\lambda}\|\theta_1-\theta_2\|_2.$$

The establishment of Lemma 2.3.1 is based on the key observation that the perturbation of $S(u,\theta)$ due to θ is bounded by the perturbation of θ through applying Proposition 6.1 in [57]. Details of the proof are given in supplementary material.

Remark 2.3.4. When we seek to learn the constraints or jointly learn the constraints and objective function, similar result can be established by applying Proposition 4.47 in [58] while restricting not only the Lipschitz continuity of the difference function in (2.8), but also the Lipschitz continuity of the distance between the feasible sets $X(u,\theta_1)$ and $X(u,\theta_2)$ (see Remark 4.40 in [58]).

Assumption 2.3.3. For the 2.1, $\forall \mathbf{y} \in \mathcal{Y}, \forall u \in \mathcal{U}, \forall \theta_1, \theta_2 \in \Theta$, $\forall \alpha, \beta \geq 0$ s.t. $\alpha + \beta = 1$, we have

$$\|\alpha S(u,\theta_1) + \beta S(u,\theta_2) - S(u, \alpha\theta_1 + \beta\theta_2)\|_2 \leq \alpha\beta \|S(u,\theta_1) - S(u,\theta_2)\|_2/(2(B+R)).$$

Essentially, this assumption requires that the distance between $S(u, \alpha\theta_1 + \beta\theta_2)$ and the convex combination of $S(u,\theta_1)$ and $S(u,\theta_2)$ shall be small when $S(u,\theta_1)$ and $S(u,\theta_2)$ are close. Actually, this assumption holds in many situations. We provide an example in supplementary material.

Let θ^* be an optimal inference to $\min_{\theta\in\Theta} \frac{1}{T}\sum_{t\in[T]} l(\mathbf{y}_t, \theta)$, i.e., an inference derived with the whole batch of observations available. Then, the following theorem asserts that $R_T = \sum_{t\in[T]}(l(\mathbf{y}_t,\theta_t) - l(\mathbf{y}_t,\theta^*))$ of the implicit online learning algorithm is of $\mathcal{O}(\sqrt{T})$.

Theorem 2.3.2 (Regret bound)**.** Suppose Assumptions 2.2.1 - 2.3.3 hold. Then, choosing $\eta_t = \frac{D\lambda}{2\sqrt{2}(B+R)\kappa}\frac{1}{\sqrt{t}}$, we have

$$R_T \leq \frac{4\sqrt{2}(B+R)D\kappa}{\lambda}\sqrt{T}. \tag{2.9}$$

Remark 2.3.5. We establish of the above regret bound by extending Theorem 3.2. in [56]. Our extension involves several critical and complicated analyses for the structure of the optimal solution set $S(u,\theta)$ as well as the loss function, which is essential to our theoretical understanding. Moreover, we relax the requirement of smoothness of loss function in that theorem to Lipschitz continuity through a similar argument in Lemma 1 of [59] and [60].

By applying both Theorem 3 in [7] and the regret bound proved in Theorem 2.3.2, we show the risk consistency of the online learning algorithm in the sense that the average cumulative loss converges in probability to the true risk in the batch setting.

Theorem 2.3.3 (Risk consistency)**.** Let $\theta^0 = \arg\min_{\theta\in\Theta}\{\mathbb{E}\left[l(\mathbf{y}, u, \theta)\right]\}$ be the optimal solution that minimizes the true risk in batch setting. Suppose the conditions in Theorem 2.3.2 hold. If $\mathbb{E}[\mathbf{y}^2] < \infty$, then choosing $\eta_t = \frac{D\lambda}{2\sqrt{2}(B+R)\kappa}\frac{1}{\sqrt{t}}$, we have

$$\frac{1}{T}\sum_{t\in[T]} l(\mathbf{y}_t, u_t, \theta_t) \xrightarrow{p} \mathbb{E}\left[l(\mathbf{y}, u, \theta^0)\right]. \tag{2.10}$$

Corollary 2.3.3.1. Suppose that the true parameter $\theta_{true} \in \Theta$, and $\mathbf{y} = \mathbf{x} + \epsilon$, where $\mathbf{x} \in S(u, \theta_{true})$ for some $u \in \mathcal{U}$, $\mathbb{E}[\epsilon] = 0, \mathbb{E}[\epsilon^T\epsilon] < \infty$, and $u, \mathbf{x}$ are independent of ϵ. Let the conditions in Theorem 2.3.2 hold. Then choosing $\eta_t = \frac{D\lambda}{2\sqrt{2}(B+R)\kappa}\frac{1}{\sqrt{t}}$, we have

$$\frac{1}{T}\sum_{t\in[T]} l(\mathbf{y}_t, u_t, \theta_t) \xrightarrow{p} \mathbb{E}[\epsilon^T\epsilon]. \tag{2.11}$$

Remark 2.3.6. (i) Theorem 2.3.3 guarantees that the online learning algorithm proposed in this dissertation will asymptotically achieves the best prediction error permitted by the inverse model we consider. (ii) Corollary 2.3.3.1 suggests that the prediction error is inevitable as long as the data carries noise. This prediction error, however, will be caused merely by the noisiness of the data in the long run.

2.4 Applications to Learning Problems in IOP

In this section, we will provide sketches of representative applications for inferring objective functions and constraints using the proposed online learning algorithm. Our preliminary experiments have been run on Bridges system at the Pittsburgh Supercomputing Center (PSC) [61]. The mixed integer second order conic programs, which are derived from using KKT conditions in (2.4), are solved by Gurobi. All the algorithms are programmed with Julia [62].

2.4.1 Learning Consumer Behavior

We now study the consumer's behavior problem in a market with n products. The prices for the products are denoted by $\mathbf{p}_t \in \mathbb{R}^n_+$ which varies over time $t \in [T]$. We assume throughout that the consumer has a rational preference relation, and we take u to be the utility function representing these preferences. The consumer's decision making problem of choosing her most preferred consumption bundle $\mathbf{x}$ given the price vector $\mathbf{p}_t$ and budget b can be stated as the following utility maximization problem (UMP) [63]:

$$\begin{aligned} \max_{\mathbf{x}\in\mathbb{R}^n_+} \quad & u(\mathbf{x}) \\ s.t. \quad & \mathbf{p}_t^T\mathbf{x} \leq b, \end{aligned} \tag{2.12}$$

where $\mathbf{p}_t^T\mathbf{x} \leq b$ is the budget constraint at time t.

For this application, we will consider a concave quadratic representation for $u(\mathbf{x})$. That is, $u(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T Q\mathbf{x} + \mathbf{r}^T\mathbf{x}$, where $Q \in \mathbf{S}^n_-$ (the set of symmetric negative semidefinite matrices), $\mathbf{r} \in \mathbb{R}^n$.

We consider a problem with $n = 10$ products, and the budget $b = 40$. Q and $\mathbf{r}$ are randomly generated and are given in supplementary material. Suppose the prices are changing in T rounds. In each round, the learner would receive one (price,noisy decision) pair $(\mathbf{p}_t, \mathbf{y}_t)$. Her goal is to learn the utility function or budget of the consumer. The (price,noisy decision) pair in each round is generated as follows. In round t, we generate the prices from a uniform distribution, i.e. $p_i^t \sim U[p_{min}, p_{max}]$, with $p_{min} = 5$ and $p_{max} = 25$. Then, we solve 2.12 and get the optimal decision $\mathbf{x}_t$. Next, the noisy decision $\mathbf{y}_t$ is obtained by corrupting $\mathbf{x}_t$ with noise that has a jointly uniform distribution with support $[-0.25, 0.25]^2$. Namely, $\mathbf{y}_t = \mathbf{x}_t + \epsilon_t$, where each element of $\epsilon_t \sim U(-0.25, 0.25)$.

Learning the utility function In the first set of experiments, the learner seeks to learn $\mathbf{r}$ given $\{(\mathbf{p}_t, \mathbf{y}_t)\}_{t\in[T]}$ that arrives sequentially in $T = 1000$ rounds. We assume that $\mathbf{r}$ is within $[0, 5]^{10}$. The learning rate is set to $\eta_t = 5/\sqrt{t}$. Then, we implement Algorithm 1 with two settings. We report our results in Figure 3. As can be seen in Figure 3a, solving the initialization problem provides quite good initialized estimations of $\mathbf{r}$, and Algorithm 1 with Warm-start converges faster than that with Cold-start. Note that (2.7) is a convex

program and the time to solve it is negligible in Algorithm 1. Thus, the running times with and without Warm-start are roughly the same. This suggests that one might prefer to use Algorithm 1 with Warm-start if she wants to get a relatively good estimation of the parameters in few iterations. However, as shown in the figure, both settings would return very similar estimations on $\mathbf{r}$ in the long run. To keep consistency, we would use Algorithm 1 with Cold-start in the remaining experiments. We can also see that estimation errors over rounds for different repetitions concentrate around the average, indicating that our algorithm is pretty robust to noises. Moreover, Figure 3b shows that inverse optimization in online setting is drastically faster than in batch setting. This also suggests that windowing approach for inverse optimization might be practically infeasible since it fails even with a small subset of data, such as window size equals to 10. We then randomly pick one repetition and plot the loss over round and the average cumulative loss in Figure 3c. We see clearly that the average cumulative loss asymptotically converges to the variance of the noise. This makes sense because the loss merely reflects the noise in the data when the estimation converges to the true value as stated in Remark 2.3.6.

Learning the budget In the second set of experiments, the learner seeks to learn the budget b in $T = 1000$ rounds. We assume that b is within $[0, 100]$. The learning rate is set to $\eta_t = 100/\sqrt{t}$. Then, we apply Algorithm 1 with Cold-start. We show the results in Figure 4. All the analysis for the results in learning the utility function apply here. One thing to emphasize is that learning the budget is much faster than learning the utility function, as shown in Figure 3b and 4b. The main reason is that the budget $\mathbf{b}$ is a one dimensional vector, while the utility vector $\mathbf{r}$ is a ten dimensional vector, making it drastically more complex to solve (2.4).

2.4.2 Learning the Transportation Cost

We now consider the transshipment network $G = (V_s \cup V_d, E)$, where nodes V_s are producers and the remaining nodes V_d are consumers. The production level is y_v for node $v \in V_s$, and has a maximum capacity of w_v. The demand level is d_v^t for node $v \in V_s$ and varies over time $t \in [T]$. We assume that producing y_v incurs a cost of $C^v(y_v)$ for node

$v \in V_s$; furthermore, we also assume that there is a transportation cost $c_e x_e$ associated with edge $e \in E$, and the flow x_e has a maximum capacity of u_e. The transshipment problem can be formulated in the following:

$$\begin{aligned} \min \quad & \sum_{v \in V_s} C^v(y_v) + \sum_{e \in E} c_e x_e \\ s.t. \quad & \sum_{e \in \delta^+(v)} x_e - \sum_{e \in \delta^-(v)} x_e = y_v, \quad \forall v \in V_s, \\ & \sum_{e \in \delta^+(v)} x_e - \sum_{e \in \delta^-(v)} x_e = d_v^t, \quad \forall v \in V_d, \\ & 0 \le x_e \le u_e, \;\; 0 \le y_v \le w_v, \quad \forall e \in E, \forall v \in V_s, \end{aligned} \tag{2.13}$$

where we want to learn the transportation cost c_e for $e \in E$. For this application, we will consider a convex quadratic cost for $C^v(y_v)$. That is, $C^v(y_v) = \frac{1}{2}\lambda_v y_v^2$, where $\lambda_v \ge 0$.

We create instances of the problem based on the network in Figure 5a. λ_1, λ_2, $\{u_e\}_{e \in E}$, $\{w_v\}_{v \in V_s}$ and the randomly generated $\{c_e\}_{e \in E}$ are given in supplementary material. In each round, the learner would receive the demands $\{d_v^t\}_{v \in V_d}$, the production levels $\{y_v\}_{v \in V_s}$ and the flows $\{x_e\}_{e \in E}$, where the later two are corrupted by noises. In round t, we generate the d_v^t for $v \in V_d$ from a uniform distribution, i.e. $d_v^t \sim U[-1.25, 0]$. Then, we solve 2.13 and get the optimal production levels and flows. Next, the noisy production levels and flows are obtained by corrupting the optimal ones with noise that has a jointly uniform distribution with support $[-0.25, 0.25]^8$.

Suppose the transportation cost on edge $(2, 3)$ and $(2, 5)$ are unknown, and the learner seeks to learn them given the (demand,noisy decision) pairs that arrive sequentially in $T = 1000$ rounds. We assume that c_e for $e \in E$ is within $[1, 10]$. The learning rate is set to $\eta_t = 2/\sqrt{t}$. Then, we implement Algorithm 1 with Cold-start. Figure 5b shows the estimation error of c in each round over the 100 repetitions. We also plot the average estimation error of the 100 repetitions. As shown in this figure, c_t asymptotically converges to the true transportation cost c_{ture} pretty fast. Also. estimation errors over rounds for different repetitions concentrate around the average, indicating that our algorithm is pretty robust to noises. We then randomly pick one repetition and plot the loss over round and the average cumulative loss in Figure 5c. Note that the variance of the noise $\mathbf{E}[\epsilon^T \epsilon] = 0.1667$.

We can see that the average cumulative loss asymptotically converges to the variance of the noise.

2.5 Conclustions and Final Remarks

In this dissertation, an online learning method to infer preferences or restrictions from noisy observations is developed and implemented. We prove a regret bound for the implicit online learning algorithm under certain regularity conditions, and show the algorithm is statistically consistent, which guarantees that our algorithm will asymptotically achieves the best prediction error permitted by the inverse model. Finally, we illustrate the performance of our learning method on both a consumer behavior problem and a transshipment problem. Results show that our algorithm can learn the parameters with great accuracy and is very robust to noises, and achieves drastic improvement in computational efficacy over the batch learning approach.iop

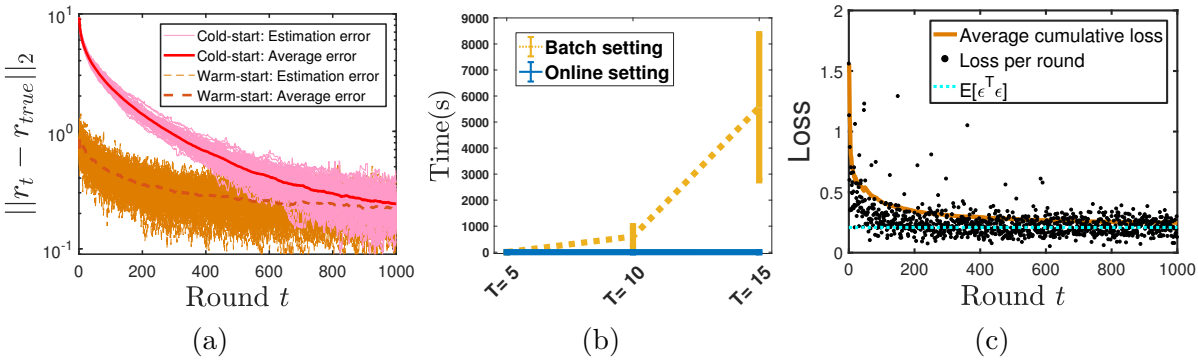


Figure 3: Learning the utility function over $T = 1000$ rounds. (a) We run 100 repetitions of the experiments using Algorithm 1 with two settings. **Cold-start** means that we initialize $\mathbf{r}$ as a vector of zeros. **Warm-start** means that we initialize $\mathbf{r}$ by solving the initialization problem (2.7) with the 1000 (price,noisy decision) pairs. We plot the estimation errors over round t in pink and brown for all the 100 repetitions, respectively. We also plot the average estimation errors of the 100 repetitions in red line and dashed brown line, respectively. (b) The dotted brown line is the error bar plot of the average running time over 10 repetitions in batch setting. The blue line is the error bar plot of the average running time over 100 repetitions in online setting. (c) We randomly pick one repetition. The loss over round is indicated by the dot. The average cumulative loss is indicated by the line. The dotted line is the reference line indicating the variance of the noise. Here, $\mathbf{E}[\epsilon^T\epsilon] = 0.2083$.

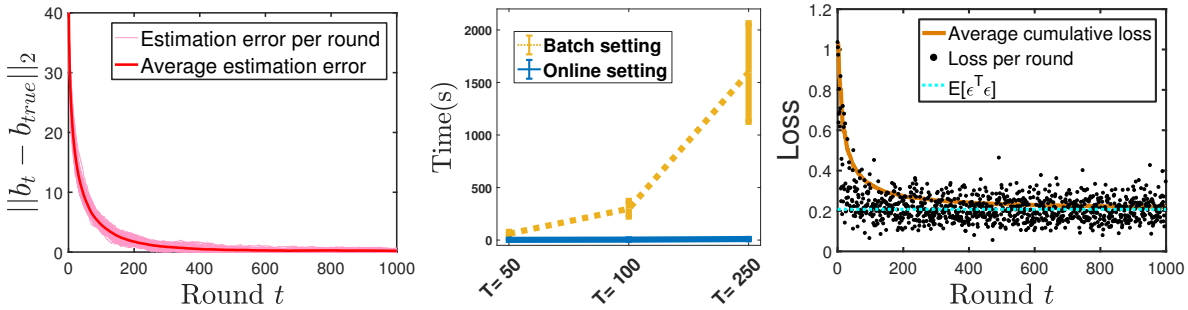
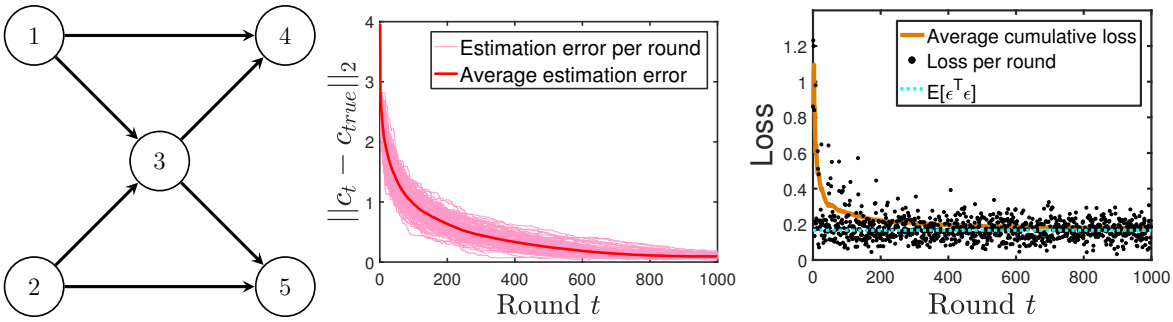


Figure 4: Learning the budget over $T = 1000$ rounds. (a) We run 100 repetitions of the experiments. We plot the estimation error over round t for all the 100 repetitions in pink. We also plot the average estimation error of the 100 repetitions in red. (b) The dotted brown line is the error bar plot of the average running time over 10 repetitions in batch setting. The blue line is the error bar plot of the average running time over 100 repetitions in online setting. (c) We randomly pick one repetition. The loss over round is indicated by the dot. The average cumulative loss is indicated by the line. The dotted line is the reference line indicating the variance of the noise. Here, $\mathbf{E}[\epsilon^T\epsilon] = 0.2083$.



(a) (b) (c)

Figure 5: Learning the transportation cost over $T = 1000$ rounds. (a) We plot the five-node network in our experiment. (b) Denote $c \in \mathbb{R}^{|E|}$ the vector of transportation costs. We run 100 repetitions of the experiments. We plot the estimation error at each round t for all the 100 experiments. We also plot the average estimation error of the 100 repetitions. (c) We randomly pick one repetition. The loss over round is indicated by the dot. The average cumulative loss is indicated by the line. The dotted line is the reference line indicating the variance of the noise. Here, $\mathbf{E}[\epsilon^T\epsilon] = 0.1667$.

3.0 Inferring Parameters Through Inverse Multiobjective Optimization

3.1 Literature Review

The inverse multiobjective optimization research is rather new and much less investigated. The research in [64] considers an IOP for a binary integer DMP given a set of linear objective functions, and develops branch-and-bound and cutting plane algorithms, which are not numerically evaluated yet, to find minimal adjustment of the objective functions such that a given set of feasible solutions becomes Pareto optimal. Research in [8, 65] addresses another situation where preferences or weights of several known criteria in the decision making problem will be inferred based on a single noisy observation. A demonstration on cancer therapy shows that their inversely optimized weights of medical metrics leads to clinically acceptable treatments. Different from those studies, our study follows the data-driven approach to build an unconventional IMOP framework that directly considers many noisy observations to infer multiple objective functions or constraints of a convex DMP with a solid statistical significance. Detailed discussions of the differences are provided in Table 2.

3.2 Inverse Multiobjective Optimization

3.2.1 Decision Making Problem with Multiple Objectives

Consider the following decision making problem with p (≥ 2) objective functions parameterized by θ:

$$\begin{array}{ll} \min\limits_{\mathbf{x}\in\mathbb{R}^n} & \{f_1(\mathbf{x},\theta), f_2(\mathbf{x},\theta), \ldots, f_p(\mathbf{x},\theta)\} \\ s.t. & \mathbf{x} \in X(\theta). \end{array} \tag{3.1}$$

For easy exposition, we use $\mathbf{f}(\mathbf{x},\theta)$ to denote the vector of objective functions

$$(f_1(\mathbf{x},\theta), f_2(\mathbf{x},\theta), \ldots, f_p(\mathbf{x},\theta))^T.$$

Table 2: Comparisons of the Problem Settings and Leaning Tasks for Different Models.

Paper	Linear case	Observations	Learn weights	Learn parameters
[64]	yes	multiple, noiseless	no	yes
[8]	yes	single, noisy	yes	no
[65]	no, general convex	single, noisy	yes	no
this dissertation	no, general convex	multiple, noisy	yes	yes

Also, the feasible set $X(\theta)$ is characterized as $X(\theta) = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{g}(\mathbf{x}, \theta) \leq \mathbf{0}\}$, where $\mathbf{g}(\mathbf{x}, \theta) = (g_1(\mathbf{x}, \theta), \ldots, g_q(\mathbf{x}, \theta))^T$ is another vector-valued function.

Following the current mainstream of inverse optimization study [1, 13, 7, 10], we restrict our focus to a convex DMP. Formally, we make the following assumption throughout the paper.

Assumption 3.2.1. Θ is a convex set. For each $\theta \in \Theta$, $\mathbf{f}(\mathbf{x}, \theta)$ is convex in $\mathbf{x}$, i.e., $f_l(\mathbf{x})$ is convex on $X(\theta)$ for all $l \in [p]$. Here, $X(\theta)$ is also a convex set for each $\theta \in \Theta$.

Definition 3.2.1 (Pareto optimality)**.** A decision vector $\mathbf{x}^* \in X(\theta)$ is said to be *Pareto optimal* (or *efficient*, or *non-dominated*) if there exists no other decision vector $\mathbf{x} \in X(\theta)$ such that $f_i(\mathbf{x}, \theta) \leq f_i(\mathbf{x}^*, \theta)$ for all $i \in [p]$, and $f_k(\mathbf{x}, \theta) < f_k(\mathbf{x}^*, \theta)$ for some $k \in [p]$.

In the study of multiobjective optimization, the set of all Pareto optimal solutions is denoted by $X_P(\theta)$ and called the Pareto optimal set. A common way to derive a Pareto optimal solution is to solve a problem with a single objective function constructed by the weighted sum of original functions, i.e., to solve the following problem [66].

$$\begin{aligned} \min \quad & w^T \mathbf{f}(\mathbf{x}, \theta) \\ s.t. \quad & \mathbf{x} \in X(\theta) \end{aligned} \tag{3.2}$$

where $w = (w^1, \ldots, w^p)^T$ is the nonnegative weight vector in the $(p-1)$-simplex $\mathscr{W}_p \equiv \{w \in \mathbb{R}^p_+ : \mathbf{1}^T w = 1\}$. When all weight components are required to be positive, such set is denoted by $\mathscr{W}_p^+$. Denote $S(w, \theta)$ the set of optimal solutions of 3.2, i.e., $S(w, \theta) = \arg\min_{\mathbf{x}} \{w^T \mathbf{f}(\mathbf{x}, \theta) : \mathbf{x} \in X(\theta)\}$.

Then, we have a couple of theoretical results regarding 3.2 that directly follow Theorems 3.1.1 - 3.1.3 of [67].

Proposition 3.2.1. Let $\mathbf{x} \in S(w, \theta)$ be an optimal solution of 3.2. The following statements hold.

(a) If $w \in \mathscr{W}_p^+$, then $\mathbf{x} \in X_P(\theta)$.

(b) If $\mathbf{x}$ is the unique optimal solution of 3.2, then $\mathbf{x} \in X_P(\theta)$.

According to Proposition 3.10 of [68] and Theorem 3.1.4 of [67], all Pareto optimal solutions of a convex 3.1 can be found by solving 3.2.

Proposition 3.2.2. Given that 3.1 is convex and $\mathbf{x} \in X_P(\theta)$, there exists a weight vector $w \in \mathscr{W}_p$ such that $\mathbf{x}$ is an optimal solution to 3.2, i.e., $\mathbf{x} \in S(w, \theta)$.

Based on Propositions 3.2.1 and 3.2.2, the following inclusive relationships can be derived.

Corollary 3.2.2.1. For a convex 3.1,

$$\bigcup_{w \in \mathscr{W}_p^+} S(w, \theta) \subseteq X_P(\theta) \subseteq \bigcup_{w \in \mathscr{W}_p} S(w, \theta). \tag{3.3}$$

Remark: (i) Results in Corollary 3.2.2.1 provides us a theoretical basis to make use of the weighted sum method to derive all Pareto optimal solutions. Actually, when 3.1 is convex and the objective functions are strictly convex, we have $X_P(\theta) = \bigcup_{w \in \mathscr{W}_p} S(w, \theta)$. (ii) When 3.1 is convex and $X(\theta)$ is compact, one important property of $X_P(\theta)$ is that it is a connected set, which, however, might not be convex as stated in [69, 68]. We note that it is very different from the situation of a convex single objective optimization problem, whose optimal solution set is convex.

3.2.2 Models for IMOP as an Unsupervised Learning Task

In this section, we present the development of our inverse multiobjective optimization models for parameter learning. Specifically, given a set of observations that are noisy Pareto optimal solutions collected from the decision maker population under study, we construct models for IMOP to infer parameter θ of 3.1. In addition to its more sophisticated structure, it is worth pointing out that we must handle a new challenge that does not occur in any

inverse optimization with a single objective function. Different from the single objective case that typically employs observations consisting of clear signal-response pairs [1, 13, 7, 10], decision makers' decisions are often observed without any information on their trade-off among objective functions. Hence, this is an unsupervised learning problem. Under such a situation, as demonstrated in this section, an unconventional framework for IMOP shall be developed to address this challenge.

We consider a set of observations that are noisy Pareto optimal solutions collected with possible measurement errors or decision makers' bounded rationality. Let $\mathbf{y}$ denote one such observation that is distributed according to an unknown distribution $\mathbb{P}_{\mathbf{y}}$ and supported on $\mathcal{Y}$. As noted in [7, 10], noise might come from measurement error, and thus $\mathbf{y}$ does not necessarily belong to $X(\theta)$.

Next, we describe the construction of our loss function with respect to a hypothesis θ. When weights over objective functions, i.e., the weight vector w, are known, the conventional quadratic loss function can be directly applied with respect to $\mathbf{y}$ and $S(w, \theta)$. Nevertheless, as previously mentioned, w is often missing and the Pareto optimal set should be adopted instead.

$$l(\mathbf{y}, \theta) = \min_{\mathbf{x} \in X_P(\theta)} \|\mathbf{y} - \mathbf{x}\|_2^2, \tag{3.4}$$

where $X_P(\theta)$ is the Pareto optimal set of 3.1 for a given θ.

Remark 3.2.1. Note that data for the single objective IOP typically consists of clear signal-response pairs $\{(u_i, \mathbf{y}_i)\}_{i \in [N]}$ [1, 13, 7, 10], where the signal u_i can be seen as the input or predictor variables and the response $\mathbf{y}_i$ is the output or response variables. Consequently, all the loss functions adopted in the single objective IOP essentially belong to the supervised learning type of loss functions. In contrast, the (3.4) defined above is of unsupervised learning type because the data available to the learner is only the decision makers' decisions $\{\mathbf{y}_i\}_{i \in [N]}$, not including any information on their preferences $\{w_i\}_{i \in [N]}$ that generate these decisions. In other words, the weight information associated with $\mathbf{y}_i$ is a "hidden variable". Particularly, both our loss function and the one used in [7] characterize the squared distance between an observed response $\mathbf{y}_i$ and a set. However, the set in [7] is determined by the signal u_i while ours is independent of any signal. That said, our loss function is of unsupervised learning

type while theirs is not. This big difference also highlights the main reason why IMOP is much more complex than single objective IOP.

Using (3.4), our inverse multiobjective optimization problem can be formulated as follows

$$\min_{\theta\in\Theta} \quad M(\theta) \equiv \mathbb{E}\Big(l(\mathbf{y},\theta)\Big), \tag{3.5}$$

where $M(\theta)$ is also called the risk of the loss function $l(\mathbf{y},\theta)$ for the hypothesis θ.

Practically, θ can not be learned by directly solving 3.5 as $\mathbb{P}_{\mathbf{y}}$ is not known a priori. Given available observations $\{\mathbf{y}_i\}_{i\in[N]}$, it is often the case that θ will be inferred through solving the following empirical risk minimizing problem:

$$\min_{\theta\in\Theta} \quad M^N(\theta) \equiv \tfrac{1}{N}\sum_{i\in[N]} l(\mathbf{y}_i,\theta). \tag{3.6}$$

Nevertheless, one remaining challenge of using (3.4) is that there is no general approach to comprehensively and explicitly characterize the Pareto optimal set $X_P(\theta)$. One way is to introduce weight variable representing the appropriate weight and convert the (3.4) into

$$\min_{w\in\mathscr{W}_p,\mathbf{x}\in S(w,\theta)} \|\mathbf{y}-\mathbf{x}\|_2^2.$$

However, this approach might not be suitable for a data-driven study, since it results in a drastically complicated model, where every single observation requires one weight variable and the nonlinear term between it and θ is heavily involved. On the contrary, according to Corollary 3.2.2.1 and its following remarks, we adopt a sampling approach to generate $w_k \in \mathscr{W}_p$ for each $k \in [K]$ and approximate $X_P(\theta)$ as the union of their $S(w_k,\theta)$s. Then, by utilizing binary variables that select an appropriate Pareto optimal solution from this union, the loss function is converted into the following *sampling based loss problem.*

$$\begin{aligned} l_K(\mathbf{y},\theta) \quad = \quad & \min_{\mathbf{x}_k, z_k\in\{0,1\}} \ \|\mathbf{y}-\sum_{k\in[K]} z_k\mathbf{x}_k\|_2^2 \\ & \text{s.t.} \quad \sum_{k\in[K]} z_k = 1,\ \mathbf{x}_k \in S(w_k,\theta). \end{aligned} \tag{3.7}$$

where constraint $\sum_{k\in[K]} z_k = 1$ ensures that one and only one of Pareto optimal solutions will be selected to approximate the distance from $\mathbf{y}$ to $X_P(\theta)$. Hence, solving this problem

identifies some w_k with $k \in [K]$ such that one Pareto optimal solution from $S(w_k, \theta)$ is closest to $\mathbf{y}$.

Comparisons between (3.4) and (3.7) are illustrated in Figure 6. The convergence rate of $l_K(\mathbf{y}, \theta)$ to $l(\mathbf{y}, \theta)$ is of $\mathcal{O}(1/K^{\frac{1}{p-1}})$. Details are provided in section 3.4. As p increases, we might require (approximately) exponentially more weight samples $\{w_K\}_{k\in[K]}$ to achieve an approximation accuracy. In fact, this phenomenon is a reflection of *curse of dimensionality* [70], a principle that estimation becomes exponentially harder as the number of dimension increases. In particular, the dimension here is the number of objective functions p.

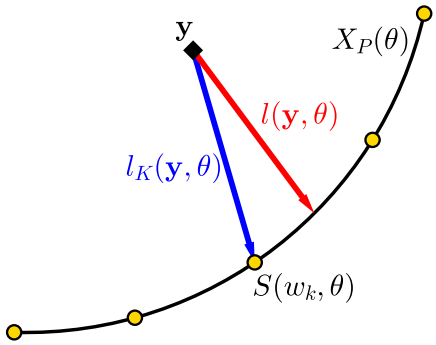


Figure 6: Illustration of the Loss Dunction and Surrogate Loss Function. Yellow Dots are the Pareto Optimal Solutions Sampled from $X_P(\theta)$. Red and Blue Arrows Indicate $l(\mathbf{y}, \theta)$ and $l_K(\mathbf{y}, \theta)$, Respectively.

Remark 3.2.2. (i) Similar to (3.4), the (3.7) still belongs to the unsupervised learning type because the observation $\mathbf{y}$ is independent from the sampled weights $\{w_k\}_{k\in[K]}$ and there is no external weight information associated with $\mathbf{y}$. Rather, the (3.7) would help reveal the hidden weight associated with $\mathbf{y}$.

(ii) As shown in Corollary 3.2.2.1, it is guaranteed that no Pareto optimal solution will be excluded if all weight vectors in $\mathscr{W}_p$ are enumerated. As it is practically infeasible, we can control the number of sampled weights K to achieve a desired tradeoff between the approximation accuracy and computational efficacy. Certainly, if the computational power is strong, we would suggest to draw a large number of weights evenly in $\mathscr{W}_p$ to avoid any bias. Although a set of binary variables $\{z_{ik}\}_{k\in[N]}$ is needed for each observation $\mathbf{y}_i$,

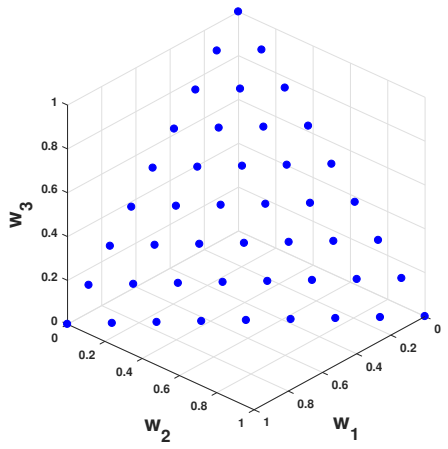


Figure 7: Blue Dots Indicate the Evenly Sampled Weights from the 3-dimensional Simplex $\mathscr{W}_3$. Here, $\mathbf{w}_1 + \mathbf{w}_2 + \mathbf{w}_3 = 1$.

the number of sampled weights K is independent from the number of observations N. As an example, we show the evenly sampled weights when $p = 3$ in Figure 7.

(iii) Indeed, as shown later in section 3.4.3, the large number of weight samples help recover the distribution of weights among decision makers under suitable conditions. As discussed earlier, such information should be very critical to manufacturers or service providers when dealing with many customers.

As previously mentioned, we indeed do not have the explicit representation of $X_P(\theta)$. Through the sampling approach described in the last subsection, variants of 3.5 using (3.7) can be easily defined. The following one is to reformulate 3.5 with weight samples, which helps us perform theoretical analysis of the reformulation of 3.6.

$$\min_{\theta \in \Theta} \quad M_K(\theta) \equiv \mathbb{E}\Big(l_K(\mathbf{y}, \theta) \Big). \tag{3.8}$$

Next, we provide the reformulation of 3.6 with the (3.7). As it serves as the primary model for analysis and computation, we present its comprehensive form to facilitate our discussion and understanding.

$$
\begin{aligned}
\min_{\theta\in\Theta} \quad & M_K^N(\theta) \equiv \tfrac{1}{N}\sum_{i\in[N]} \|\mathbf{y}_i - \sum_{k\in[K]} z_{ik}\mathbf{x}_k\|_2^2 \\
\text{s.t.} \quad & \mathbf{x}_k \in S(w_k,\theta), && \forall k\in[K], \\
& \sum_{k\in[K]} z_{ik} = 1, && \forall i\in[N], \\
& z_{ik}\in\{0,1\}, && \forall i\in[N],\ k\in[K].
\end{aligned}
\tag{3.9}
$$

Remark 3.2.3. 3.9 is the only model one can practically compute among all four models discussed previously. By making use of optimality conditions to represent $S(w_k,\theta)$, 3.9 can be solved numerically to derive an estimation of θ and two examples are provided in C.4.2 and C.4.3. According to [7, 10], existing data-driven inverse optimization models primarily differ from each other by using different loss functions. Our 3.9 model clearly has a more sophisticated structure with many new variables and constraints, which probably are necessary due to the learning context and task. To handle the incurred computational challenge, advanced algorithm developments are presented in section 3.5, which support our real applications with a greatly improved efficiency.

Before proceeding to next section, we summarize the proposed models for IMOP in Table 3, where **Empirical** and **Obj** mean that we use empirical risk and the specific objective function, respectively. Here, N is the number of observations, and K denotes the number of weight samples.

Table 3: Summary of Four IMOP Models

Model	Risk/Empirical	Loss function	Obj	Estimator	Computable
3.5	Risk	$l(\mathbf{y},\theta)$	$M(\theta)$	θ^*	✗
3.6	Empirical	$l(\mathbf{y},\theta)$	$M^N(\theta)$	$\hat{\theta}^N$	✗
3.8	Risk	$l_K(\mathbf{y},\theta)$	$M_K(\theta)$	$\hat{\theta}_K$	✗
3.9	Empirical	$l_K(\mathbf{y},\theta)$	$M_K^N(\theta)$	$\hat{\theta}_K^N$	✓

3.2.3 IMOP, Inverse Optimization, and Machine Learning

For simplicity and avoiding any confusion, we refer to data-driven inverse optimization problem with single objective function as IOP throughout the remainder of this dissertation.

Data-driven inverse optimization is the current mainstream and has been extensively investigated recently for the single objective case [1, 13, 7, 10, 6]. Similarly, our paper also follows the data-driven approach and takes a learning perspective to build an IMOP framework that directly considers many noisy observations. To clarify differences and connections, we propose a taxonomy that is applicable to both IOP and IMOP from the machine learning point of view.

As summarized in Table 1, IMOP transitions from a supervised learning task into an unsupervised learning task when less weight information for the decision is accompanied, while IOP is a supervised learnig task given that the data consists of clear signal-response pairs.

For the first class of IMOP, the weight-decision pair $(w_i, \mathbf{y}_i)$ is available to the learner for each $i \in [N]$. As a result, IMOP naturally becomes a supervised learning task [70] because w_i can be seen as the input or predictor variables and y_i is the output or response variables. In other words, $\{(w_i, \mathbf{y}_i)\}_{i\in[N]}$ is the set of training samples for the supervised learning problem of inferring the parameter θ in 3.1. Similar to any other supervised learning tasks, once obtained, 3.1 could be used to predict the decision maker's behavior given the preference w over different objective functions. Additionally, we note that IOP in essence is also a supervised learning task and IMOP degenerates into IOP when the weight-decision pair $(w_i, \mathbf{y}_i)$ is available to the learner for each $i \in [N]$. Here, the weight and decision correspond to the external signal (input) and response (output) in IOP, respectively. Therefore, all those methods for IOP are readily applicable to IMOP, making it the simplest among all three classes listed in Table 1.

For the second class of IMOP, some decisions are observed with the weight information while others are not. Through a similar analysis for the first class, IMOP is an unconventional semi-supervised learning task [71] where the supervision comes from the predictor variables, i.e., the additional weight information associated with the decisions. Specifically, it occurs

that some decisions are observed with knowledge on the range of weights over those objective functions. For example, some decision makers are risk-averse, indicating that their decisions are with large weights over the function representing risk. In fact, 3.9 can be easily extended to handle such situations without much efforts. More precisely, we merely need to replace the second set of constraints in 3.9 by the following constraints.

$$\begin{aligned} &\sum_{k\in\widetilde{K}_i} z_{ik} = 1, \quad \forall i \in [N'], \\ &\sum_{k\in[K]} z_{ik} = 1, \quad \forall i \in [N] \setminus [N'], \end{aligned} \tag{3.10}$$

where the first N' observations are with some information on weights captured in subset $\widetilde{K}_i \subseteq [K]$ for each $i \in [N']$. If we would like to emphasize the contribution of the observations in learning, the objective function of 3.9 can be modified as follows:

$$\min_{\theta\in\Theta} \tfrac{1}{N} \sum_{i\in[N]\setminus[N']} \|\mathbf{y}_i - \sum_{k\in[K]} z_{ik}\mathbf{x}_k\|_2^2 + \tfrac{\lambda}{N} \sum_{i\in[N']} \|\mathbf{y}_i - \sum_{k\in\widetilde{K}_i} z_{ik}\mathbf{x}_k\|_2^2 \tag{3.11}$$

where coefficient $\lambda \geq 1$ reflects the value of such more specific information.

For the third class of IMOP, the learner only has access to the decisions without any weight information. Thus, IMOP is an unsupervised learning task and the goal is to recover the structure of the Pareto optimal set from which these decisions are generated. However, this does not mean that the weight w disappears in our setting. In contrast, w appears in IMOP as a hidden variable and generates the decision $\mathbf{y}$ together with θ as shown in Figure 8. Just like any other machine learning tasks involving hidden variables [72], we need to learn w in order to infer the model parameter θ. Moreover, we can also see that the main difference between IOP and IMOP of this class is whether the predictors are hidden variables or not. As a consequence, the involvement of hidden information makes IMOP much more complex than IOP.

Throughout the remainder of the paper, we focus on discussing IMOP of the third class, and refer to it as IMOP for two main reasons. The first is that the weight information is typically very expensive to obtain in practice and what the learner can observe are only the decisions. Another reason is that the model for IMOP, i.e., 3.9, is quite flexible and analysis and algorithms for it can be readily extended to other two classes.

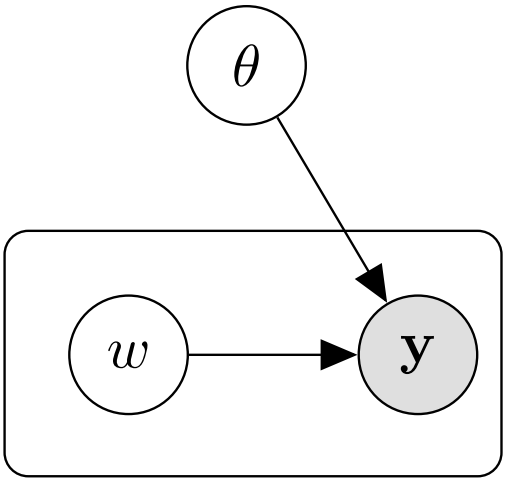


Figure 8: Graphical Plate Model of IMOP. The Unshaded Nodes are Hidden Variables and the Shaded Nodes Represent the Observed Variables. The Directed Links Indicate Dependencies between Variables.

3.3 Connections between IMOP, Clustering and Manifold Learning

In section 3.2.3, we show that IMOP is an unsupervised learning task. Subsequently, we study in this section connections between IMOP and two seemingly unrelated unsupervised learning tasks. The first one is the clustering problem, in particular the K-means clustering problem [18, 19]. The second one is the manifold learning problem, which seeks to construct low-dimensional manifolds from data points embedded in high-dimensional spaces [24, 25].

3.3.1 Connection bewteen IMOP and Clustering

We show that every K-means clustering problem can be transformed equivalently to an IMOP. Consequently, we prove the NP-hardness of IMOP from the reduction of K-means clustering problem. Conversely, we show that IMOP indeed can be interpreted as a Constrained K-means problem.

K-means clustering aims to partition the observations into K clusters such that the average squared distance between each observation and its closest cluster centroid is minimized.

Given observations $\{\mathbf{y}_i\}_{i\in[N]}$, a mathematical formulation of K-means clustering is presented in the following [73, 74].

$$\begin{aligned} \min_{\mathbf{x}_k, z_{ik}} \quad & \tfrac{1}{N}\sum_{i\in[N]} \|\mathbf{y}_i - \sum_{k\in[K]} z_{ik}\mathbf{x}_k\|_2^2 \\ \text{s.t.} \quad & \sum_{k\in[K]} z_{ik} = 1, && \forall i \in [N], \\ & \mathbf{x}_k \in \mathbb{R}^n, \;\; z_{ik} \in \{0,1\}, && \forall i \in [N],\; k \in [K], \end{aligned} \tag{3.12}$$

where K is the number of clusters, and $\{\mathbf{x}_k\}_{k\in[K]}$ are the centroids of the clusters.

Proposition 3.3.1. Given any 3.12 problem, we can construct an instance of 3.9, such that solving the K-means clustering problem is equivalent to solving the instance of 3.9.

The key step for the proof in Proposition 3.3.1 is to construct a 3.1 whose objective functions are quadratic and feasible region is a ball. Details of the proof are given in the supplementary material.

Lemma 3.3.2 ([21, 22])**.** 3.12 is NP-hard.

One should distinguish K-means clustering problem from K-means algorithm (a.k.a. Lloyd's algorithm) [19], where the later one is a fast heuristic to solve the former problem. Indeed, K-means clustering problem is NP-hard to solve even for instances in the plane [22], or $K = 2$ in general dimension [21].

Theorem 3.3.3 (NP-hardness of IMOP)**.** In general, 3.9 is NP-hard.

We conclude the proof by further noting that the construction is indeed polynomial. By Lemma 3.3.2, K-means clustering problem is NP-hard to solve even for instances in the plane, or with two clusters in the general dimension. This suggests that 3.9 is also difficult to solve even for instances in the plane, or $K = 2$ in general dimension.

Denote M_K the optimal value for 3.12. The following theorem depicts the relationship between M_K and $M_K^N(\hat{\theta}_K^N)$.

Theorem 3.3.4 (Constrained K-means clustering)**.** Given any observations $\{\mathbf{y}_i\}_{i\in[N]}$, and weight samples $\{w_k\}_{k\in[K]}$, we have $M_K \le M_K^N(\hat{\theta}_K^N)$. Moreover, $M_K = M_K^N(\hat{\theta}_K^N)$ if and only if there exists a 3.1, such that its Pareto optimal solutions $\{S(w_k, \theta)\}_{k\in[K]}$ are the centroids obtained by 3.12.

Now, we explain why one can interpret 3.9 as a Constrained K-means clustering problem. Here, the meaning of *Constrained* in our paper is slightly different from that of [23]. While both emphasize the incorporation of background knowledge into the clustering process, their *Constrained* means more about which observations should or should not be grouped together. Note that 3.9 has one more type of constraints than 3.12, i.e., $\mathbf{x}_k \in S(w_k, \theta), \forall k \in [K]$. These constraints require that the centroids of the clusters are restricted to be Pareto optimal solutions of the estimated 3.1. This also explains why 3.9 always has a larger optimal value than 3.12 as shown in Theorem 3.3.4.

3.3.2 Connection between IMOP and Manifold Learning

We show in this section that the Pareto optimal set is a piecewise continuous manifold with intrinsic dimension of $p-1$, where p is the number of objectives, regardless of the dimension of the decision space. Furthermore, we show IMOP could be interpreted as a manifold learning problem since solving IMOP in essence is to construct a 3.1 whose Pareto optimal closely matches observations.

Given a set of high-dimensional observations $\{\mathbf{y}_i\}_{i\in[N]}$ in $\mathbb{R}^n$, manifold learning attempts to find an embedding set $\{\mathbf{x}_i\}_{i\in[N]}$ in a low-dimensional space $\mathbb{R}^d$ ($d < n$), and the local manifold structure formed by $\{\mathbf{y}_i\}_{i\in[N]}$ is preserved in the embedded space [25, 24, 75]. Formally, given a set of data points $\{\mathbf{y}_i\}_{i\in[N]}$, we are required to find a mapping $f : \mathbb{R}^d \to \mathbb{R}^n$ and another set of points $\{\mathbf{x}_i\}_{i\in[N]}$ in $\mathbb{R}^d$ such that

$$\mathbf{y}_i = f(\mathbf{x}_i) + \epsilon_i, \quad i \in [N], \tag{3.13}$$

where ϵ_i represents random noise.

The central questions of manifold learning are: 1) Can we find a set of low-dimensional points $\{\mathbf{x}_i\}_{i\in[N]}$ such that the equation (3.13) holds? 2) What kind of regularity conditions should be imposed on f? 3) Is the model well defined [76, 77] ? These questions are the main focus of this section in the context of IMOP.

Theorem 3.3.5 (Pareto manifold)**.** Suppose Assumption 3.2.1 holds. For each $\theta \in \Theta$, the Pareto optimal set of 3.1 is a $(p-1)$-dimensional piecewise continuous manifold, where p is the number of objectives.

From this theorem, one can see that the Pareto optimal set of a 3.1 with two objectives is a piecewise continuous curve, and the Pareto optimal set of a 3.1 with three objectives is a piecewise continuous surface, etc.

Corollary 3.3.5.1. Suppose that both $\mathbf{f}(\mathbf{x}, \theta)$ and $\mathbf{g}(\mathbf{x}, \theta)$ are linear functions in $\mathbf{x}$ for all $\theta \in \Theta$. Then, $X_P(\theta)$ is a $(p-1)$-dimensional piecewise linear manifold for all $\theta \in \Theta$.

Remark 3.3.1. Note that the feasible set for a multiobjective linear program is a polyhedron. Thus, one way to interpret Corollary 3.3.5.1 is that the Pareto optimal set of such a program consists of Pareto optimal faces of the polyhedron that are arc-wise connected. Therefore, the Pareto optimal set naturally has a piecewise linear structure and forms a manifold.

Recall that $\{\mathbf{x}_k\}_{k\in[K]}$ are used in 3.9 to measure the distances between the observations $\{\mathbf{y}_i\}_{i\in[N]}$ and the underlying Pareto optimal set, which is a manifold by Theorem 3.3.5. Also, $\mathbf{x}_k$ is restricted to be an optimal solution of the 3.2 for all $k \in [K]$. That is, $\mathbf{x}_k \in S(w_k, \theta)$ for all $k \in [K]$. Then, following from Theorem 3.1.1 - 3.1.3 of [67], we have the following result regarding $\{\mathbf{x}_k\}_{k\in[K]}$.

Proposition 3.3.6. Suppose Assumption 3.2.1 holds. Then, $S(w_k, \theta) \subseteq X_P(\theta)$ for all $w \in \mathscr{W}_p^+$. If $\mathbf{f}(\mathbf{x}, \theta)$ is strictly convex in $\mathbf{x}$, then $S(w_k, \theta) \subseteq X_P(\theta)$ for all $w \in \mathscr{W}_p$.

Now, we explain why we can interpret IMOP as a manifold learning problem. Combining Proposition 3.3.6 and the way we sample $\{w_k\}_{k\in[K]}$ in Remark 3.2.2, one can show that $\{\mathbf{x}_k\}_{k\in[K]}$ in 3.9 are Pareto optimal points on the $X_P(\theta)$ to be estimated. Note that 3.9 is solved by minimizing the average distance between $\{\mathbf{y}_i\}_{i\in[N]}$ and $\{\mathbf{x}_k\}_{k\in[K]}$. Therefore, IMOP essentially seeks to find the 3.1 whose Pareto optimal set matches best the true Pareto optimal set where $\{\mathbf{y}_i\}_{i\in[N]}$ are sampled from.

Remark 3.3.2. Manifold learning methods typically returns a set of points in the dimension-reduced space [24, 25]. By solving 3.9, however, we obtain a set of representative points $\{\mathbf{x}_k\}_{k\in[K]}$ of a manifold in the decision space, instead of the $(p-1)$-dimensional space. Thus, the manifold recovered by solving IMOP is more like the Principal manifold introduced by [78] as lines or surfaces passing through "the middle" of the data distribution. Note that we also obtain the weights that generate the observations by solving 3.9. These weights are

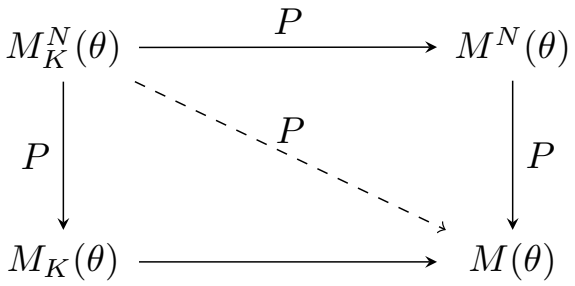


Figure 9: Uniform Convergence Diagram for Empirical Risks. $\xrightarrow{P}$ Means Convergence in Probability. $\longrightarrow$ Indicates the Convergence of a Sequence of Numbers. $\overset{P}{\dashrightarrow}$ Means Convergence in Probability for Double-index Random Variable.

lying on the $(p-1)$-dimensional manifold $\mathscr{W}_p$. Therefore, another way to interpret the result is that solving 3.9 yields a function $S(w, \theta)$ that maps a low dimensional point $w \in \mathscr{W}_p$ to a high dimensional point in the decision space. This answers the first question we ask in this section.

3.4 Consistency, Generalization Bound, and Identifiability Analysis

Our major task in this section is to show statistical properties of estimators constructed in section 3.2.2. More specifically, we show that these estimators asymptotically predict as well as the best possible result this type of inverse optimization model can achieve. In addition, we provide a generalization bound for the estimator constructed in 3.9. Subsequently, we propose the concept of identifiability in the context of decision making problems with multiple objectives, and show its correlation with the performance of our model for IMOP. We will begin by proving the uniform convergences of the empirical risks.

3.4.1 Risk Consistency of 3.9

Before proving the risk consistency of the estimators, we first need to prove the uniform convergence of the empirical risks as shown in Figure 9. Different from conventional learning tasks that consider convergence only in data size N, we need to show that the empirical risk $M_K^N(\theta)$ uniformly converges to the risk $M(\theta)$ in two directions, that is, in N and K simultaneously. As being common in the inverse optimization literature e.g., [7, 10], we now adopt the following assumptions.

Assumption 3.4.1. **(i)** The parameter set Θ is compact.

(ii) For each $\theta \in \Theta$, $X(\theta)$ is compact, and has a nonempty relatively interior. Also, $X(\theta)$ is uniformly bounded. Namely, there exists $B > 0$ such that $\|\mathbf{x}\|_2 \leq B$ for all $\mathbf{x} \in X(\theta)$ and $\theta \in \Theta$.

(iii) Functions $\mathbf{f}(\mathbf{x}, \theta)$ and $\mathbf{g}(\mathbf{x}, \theta)$ are continuous on $\mathbb{R}^n \times \Theta$.

(iv) $\mathbb{E}[\mathbf{y}^T\mathbf{y}] < +\infty$.

Assumptions (ii) and (iii) are important for the continuity of $X_P(\theta)$. Also, Assumption (iv), which is ensured once variance of the noise is finite, is fundamental to applying *the uniform law of large numbers* (ULLN) [79], one of the most used tools in performing consistency analysis.

Lemma 3.4.1. Suppose Assumptions 3.2.1 - 3.4.1 hold. $X(\theta)$ is continuous on Θ.

The continuity of $X(\theta)$ follows from its lower semicontinuity (l.s.c.) and upper semicontinuity (u.s.c.), both of which can be derived by using [80] under our assumptions.

Lemma 3.4.2. Suppose Assumptions 3.2.1 - 3.4.1 hold. If $\mathbf{f}(\mathbf{x}, \theta)$ is strictly convex in $\mathbf{x}$ for each $\theta \in \Theta$, then $X_P(\theta)$ is continuous on Θ.

Remark 3.4.1. Several things need to be emphasized when applying Theorem 7.1 of [81] to prove Lemma 3.4.2. (i) This theorem employs the condition that $X(\theta)$ is uniformly compact near θ, which guarantees that a sequence $\{\mathbf{x}_k\}$, generated from $X(\theta_k)$, contains a convergent subsequence. In Euclidean spaces, the uniform boundedness of $X(\theta)$, as stated in Assumption 3.4.1, is also adequate in the proof. (ii) This theorem gives the sufficient conditions for the l.s.c. of $X_P(\theta)$. All of these conditions are naturally satisfied under

Assumptions 3.2.1 - 3.4.1 except the one that requires $\mathbf{f}(\mathbf{x}, \theta)$ to be one-to-one, i.e., injective in $\mathbf{x}$. In fact, we can safely replace the one-to-one condition by the strict quasi-convexity of $\mathbf{f}(\mathbf{x}, \theta)$ in $\mathbf{x}$ without affecting the result. Since strict convexity implies strict quasi-convexity, the lower semicontinuity naturally follows.

Proposition 3.4.3 (ULLN for $M^N(\theta)$ in N)**.** Under the same conditions of Lemma 3.4.2, $M^N(\theta)$ uniformly converges to $M(\theta)$ in N. That is,

$$\sup_{\theta \in \Theta} |M^N(\theta) - M(\theta)| \xrightarrow{p} 0. \tag{3.14}$$

Proposition 3.4.4 (ULLN for $M_K^N(\theta)$ in N)**.** Under the same conditions of Lemma 3.4.2, $M_K^N(\theta)$ uniformly converges to $M_K(\theta)$ in N. That is, $\forall K$,

$$\sup_{\theta \in \Theta} |M_K^N(\theta) - M_K(\theta)| \xrightarrow{p} 0. \tag{3.15}$$

Throughout the paper, we use $K_2 \geq K_1$ to denote the set of weights $\{w_k\}_{k \in [K_1]} \subseteq \{w_k\}_{k \in [K_2]}$, and $K_2 > K_1$ to denote the set of weights $\{w_k\}_{k \in [K_1]} \subsetneq \{w_k\}_{k \in [K_2]}$. Then, we depict the monotonicity of $\{M_K(\theta)\}$ and $\{M_K^N(\theta)\}$ in K for each $\theta \in \Theta$ in the following lemma.

Lemma 3.4.5 (Monotonicity of $\{M_K(\theta)\}$ and $\{M_K^N(\theta)\}$ in K)**.** We have the following:

(a) The sequence $\{M_K(\theta)\}$ is monotone decreasing in K for all $\theta \in \Theta$. Moreover, $\{M_K(\hat{\theta}_K)\}$ is monotone decreasing in K. Specially, $M_K(\hat{\theta}_K) \geq M(\theta^*)$.

(b) Given any $\{\mathbf{y}_i\}_{i \in [N]}$, the sequence $\{M_K^N(\theta)\}$ is monotone decreasing in K for all $\theta \in \Theta$. Moreover, $\{M_K^N(\hat{\theta}_K^N)\}$ is monotone decreasing in K. Specially, $M_K^N(\hat{\theta}_K^N) \geq M^N(\hat{\theta}^N)$.

Lemma 3.4.6. Suppose Assumptions 3.2.1 - 3.4.1 hold. Suppose also that $\mathbf{f}(\mathbf{x}, \theta)$ is strongly convex in $\mathbf{x}$ for each $\theta \in \Theta$, that is, $\forall l \in [p]$, $\exists \lambda_l > 0$, $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n$,

$$f_l(\mathbf{y}, \theta) \geq f_l(\mathbf{x}, \theta) + \nabla f_l(\mathbf{x}, \theta)^T(\mathbf{y} - \mathbf{x}) + \frac{\lambda_l}{2}\|\mathbf{y} - \mathbf{x}\|_2^2. \tag{3.16}$$

Then, $\forall \theta \in \Theta$, $\forall w, w_0 \in \mathscr{W}_p$,

$$\|S(w, \theta) - S(w_0, \theta)\|_2 \leq \frac{2L}{\lambda}\|w - w_0\|_2, \tag{3.17}$$

where $L = \sqrt{p} \cdot \max_{l \in [p], \theta \in \Theta, \mathbf{x} \in X(\theta)} |f_l(\mathbf{x}, \theta)|$ is a finite number, and $\lambda = \min_{l \in [p]}\{\lambda_l\}$.

Lemma 3.4.7. Under Assumptions 3.2.1 - 3.4.1, we have that $\forall \mathbf{y} \in \mathcal{Y}, \forall \theta \in \Theta$,

$$0 \leq l_K(\mathbf{y}, \theta) - l(\mathbf{y}, \theta) \leq \frac{4(B+R)\zeta}{\lambda} \cdot \frac{\sqrt{2p}}{\Lambda - 1}, \tag{3.18}$$

where

$$K = \frac{(\Lambda + p - 2)!}{(\Lambda - 1)!(p-1)!}, \zeta = \max_{l \in [p], \mathbf{x} \in X(\theta), \theta \in \Theta} |f_l(\mathbf{x}, \theta)|. \tag{3.19}$$

Furthermore,

$$0 \leq l_K(\mathbf{y}, \theta) - l(\mathbf{y}, \theta) \leq \frac{16e(B+R)\zeta}{\lambda} \cdot \frac{1}{K^{\frac{1}{p-1}}}. \tag{3.20}$$

Thus, the surrogate loss function uniformly converges to the loss function at the rate of $\mathcal{O}(1/K^{\frac{1}{p-1}})$. Note that this rate exhibits a dependence on the number of objective functions p. As p increases, we might require (approximately) exponentially more weight samples $\{w_K\}_{k \in [K]}$ to achieve an approximation accuracy. In fact, this phenomenon is a reflection of *curse of dimensionality* [70], a principle that estimation becomes exponentially harder as the number of dimension increases. In particular, the dimension here is the number of objective functions p. Naturally, one way to deal with the curse of dimensionality is to employ dimension reduction techniques in statistics to find low-dimensional representation of the objective functions.

Example 3.4.1. When $p = 2$, 3.1 is a bi-objective decision making problem. Then, Lemma 3.4.7 shows that $l_K(\mathbf{y}, \theta) - l(\mathbf{y}, \theta)$ is of $\mathcal{O}(1/K)$. That is, $l_K(\mathbf{y}, \theta)$ asymptotically converges to $l(\mathbf{y}, \theta)$ sublinearly.

Proposition 3.4.8 (Uniform convergence of $M_K(\theta)$ in K)**.** Under the same conditions of Lemma 3.4.6, $M_K(\theta)$ uniformly converges to $M(\theta)$ in K for $\theta \in \Theta$. That is, $\sup_{\theta \in \Theta} |M_K(\theta) - M(\theta)| \longrightarrow 0$.

Next, we present a very mild assumption to bound random observations.

Assumption 3.4.2. The support $\mathcal{Y}$ of the distribution $\mathbf{y}$ is contained within a ball of radius R almost surely, where $R < \infty$. That is, $\mathbb{P}(\|\mathbf{y}\|_2 \leq R) = 1$.

Proposition 3.4.9 (Uniform convergence of $M_K^N(\theta)$ in K)**.** Suppose Assumptions 3.2.1 - 3.4.2 hold. If $\mathbf{f}(\mathbf{x}, \theta)$ is strongly convex in $\mathbf{x}$ for each $\theta \in \Theta$, then $M_K^N(\theta)$ uniformly converges to $M(\theta)$ in K for $\theta \in \Theta$ and N. That is, $\forall N$, $\sup_{\theta \in \Theta} |M_K^N(\theta) - M^N(\theta)| \xrightarrow{p} 0$.

We would like to point out that previous four convergence results are provided merely for theoretical understanding as neither the distribution of $\mathbf{y}$ or the Pareto optimal set $X_P(\theta)$ is available in practice. Nevertheless, they serve as the bridge to prove the uniform convergence of the numerically computable one of $M_K^N(\theta)$ to the abstract concept of $M(\theta)$. Before establishing the formal proof, we introduce one definition to support our convergence analysis with respect to both N and K.

Definition 3.4.1 (Double-index convergence)**.** Let $\{X_{mn}\}$ be an array of double-index random variables. Let X be a random variable. If $\forall \delta > 0, \forall \epsilon > 0$, $\exists N$, s.t. $\forall m, n \geq N$, $\mathbb{P}(|X_{mn} - X| > \epsilon) < \delta$. Then X_{mn} is said to converge in probability to X (denoted by $X_{mn} \xrightarrow{P} X$).

Proposition 3.4.10 (Uniform convergence of $M_K^N(\theta)$ in N and K)**.** Under the same conditions of Proposition 3.4.9, $M_K^N(\theta)$ uniformly converges to $M(\theta)$ in N and K for all $\theta \in \Theta$. That is,

$$\sup_{\theta \in \Theta} |M_K^N(\theta) - M(\theta)| \xrightarrow{P} 0. \tag{3.21}$$

We next show the risk consistency of the estimators. We denote Θ^* the set of parameters that minimizes the risk and refer to it as the optimal set. Namely, $\Theta^* = \{\theta^* \in \Theta : M(\theta^*) = \min_{\theta \in \Theta} M(\theta)\}$.

Theorem 3.4.11 (Consistency of 3.6)**.** Suppose Assumptions 3.2.1 - 3.4.1 hold. If $\mathbf{f}(\mathbf{x}, \theta)$ is strictly convex in $\mathbf{x}$ for each $\theta \in \Theta$, then $M(\hat{\theta}^N) \xrightarrow{p} M(\theta^*)$.

Theorem 3.4.11 states that $\hat{\theta}^N$ converges in probability to one point in the optimal set Θ^*.

Theorem 3.4.12 (Consistency of 3.8)**.** Suppose Assumptions 3.2.1 - 3.4.1 hold. If $\mathbf{f}(\mathbf{x}, \theta)$ is strongly convex in $\mathbf{x}$ for each $\theta \in \Theta$, then $M(\hat{\theta}_K) \xrightarrow{P} M(\theta^*)$.

Proof of Theorem 3.4.12 is essentially the same to that of Theorem 3.4.11, and is omitted.

Theorem 3.4.12 indicates that $\hat{\theta}_K$ also converges in probability to one point in the optimal set Θ^*.

Recall that 3.9 is the only one we can and will solve to infer the unknown parameters of a decision making problem among the four models listed in Table 3. Thus, the following theorem is the most important one from the perspective of computation.

Theorem 3.4.13 (Consistency of 3.9)**.** Suppose Assumptions 3.2.1 - 3.4.2 hold. If $\mathbf{f}(\mathbf{x}, \theta)$ is strongly convex in $\mathbf{x}$ for each $\theta \in \Theta$, then $M(\hat{\theta}_K^N) \xrightarrow{P} M(\theta^*)$.

Proof of Theorem 3.4.13 is essentially the same to those of Theorems 3.4.11 and 3.4.13, and is omitted.

Similar to Theorems 3.4.11 - 3.4.12, Theorem 3.4.13 indicates that $\hat{\theta}_K^N$ converges in probability to one point in the optimal set Θ^*. Actually, as we will see in EXAMPLE 3.4.2 and 3.4.3, if no information about decision makers' preference or partial understanding on θ is imposed, the optimal set Θ^* is often not a singleton even when the objective functions are strongly convex. This indicates one challenge of parameter inference through inverse multiobjective optimization. With such an observation, the risk consistency, or persistence in [82], is a more realistic standard for the estimator when learning parameters through solving IMOP.

3.4.2 Generalization Bound of 3.9

For fixed weight samples $\{w_k\}_{k\in[K]}$, we want to estimate the risk $M_K(\hat{\theta}_K^N)$ as it quantifies how well the performance of our estimator $\hat{\theta}_K^N$ generalizes to the unseen data. However, this quantity cannot be obtained since the distribution $\mathbb{P}_{\mathbf{y}}$ is unknown, and thus is a random variable (since it depends on the data). Hence, one way to make a statement about this quantity is to say how it relates to an estimate such as the empirical risk $M_K^N(\hat{\theta}_K^N)$. Before providing the main theorem, we first introduce some important definitions and lemmas.

Definition 3.4.2 (Rademacher random variables)**.** Random variables $\sigma_1, \ldots, \sigma_N$ are called *Rademacher random variables* if they are independent, identically distributed and $\mathbb{P}(\sigma_i = 1) = \mathbb{P}(\sigma_i = -1) = 1/2$ for $i \in [N]$.

Let $\mathcal{F}$ be a class of functions mapping from Z to $[a, b]$, and $Z_1, \ldots, Z_N$ be independent and identically distributed (i.i.d.) random variables on Z.

Definition 3.4.3. The Rademacher complexity of $\mathcal{F}$ is

$$Rad_N(\mathcal{F}) = \frac{1}{N}\mathbb{E}\bigg[\sup_{f\in\mathcal{F}}\sum_{i\in[N]}\sigma_i f(Z_i)\bigg], \tag{3.22}$$

where the expectation is taken over σ and $Z_1, \ldots, Z_N$.

Intuitively, $Rad_N(\mathcal{F})$ is large if one can find function $f \in \mathcal{F}$ that look like random noise, that is, these functions are highly correlated with Racemacher random variables $\sigma_1, \ldots, \sigma_N$.

Lemma 3.4.14. Let $\mathcal{F}$ be a class of functions mapping from Z to $[a, b]$. Let $Z_1, \cdots, Z_N$ be i.i.d. random variables on Z. Then, for any $0 < \delta < 1$, with probability at least $1 - \delta$, every $f \in \mathcal{F}$ satisfies

$$\mathbb{E}[f(Z)] \leq \frac{1}{N}\sum_{i\in[N]} f(Z_i) + 2Rad_N(\mathcal{F}) + (b-a)\sqrt{\frac{log(1/\delta)}{2N}}. \tag{3.23}$$

Remark 3.4.2. The last term of the inequality in Lemma 3.4.14 might not be tight. We are able to obtain tighter bounds using more complex methods such as the one in [83]. We refer the reader to [84, 85] for detailed introductions on how to characterize the generalization bound that the estimators may have in given situations.

Given K and θ, we define a function $f(\cdot, \theta)$ by $f(\mathbf{y}, \theta) = \min_{k\in[K]} \|\mathbf{y} - \mathbf{x}_k\|_2^2$, where $\mathbf{x}_k \in S(w_k, \theta)$ for all $k \in [K]$. Now consider the class of functions $\mathcal{F} = \{f(\cdot, \theta) : \theta \in \Theta\}$. To bound the risk $\mathbb{E}[f(\mathbf{y}, \theta)]$ using Lemma 3.4.14, we need to either compute the vaule of $Rad_N(\mathcal{F})$ or find an upper bound of it. Note that the computation of $Rad_N(\mathcal{F})$ involves solving a difficult optimization problem over $\mathcal{F}$. In contrast, obtaining a bound of $Rad_N(\mathcal{F})$ is relatively easier. Therefore, we seek to bound $Rad_N(\mathcal{F})$ in the following lemma.

Lemma 3.4.15. The Rademacher complexity of $\mathcal{F}$ is bounded by a function of sample size N,

$$Rad_N\big(\mathcal{F}\big) \leq \frac{K}{\sqrt{N}}\bigg(B^2 + 2BR\bigg). \tag{3.24}$$

Theorem 3.4.16 (Generalization bound)**.** Suppose Assumptions 3.2.1 - 3.4.2 hold. For any $0 < \delta < 1$, with probability at least $1 - \delta$ with respect to the observations,

$$M_K(\hat{\theta}_K^N) \leq M_K^N(\hat{\theta}_K^N) + \frac{1}{\sqrt{N}}\left(2K(B^2 + 2BR) + (B+R)^2\sqrt{\log(1/\delta)/2}\right) \text{ for each } K. \tag{3.25}$$

Essentially, this theorem indicates that the risk of the estimator constructed by solving 3.9, which can be seen as the test error for fixed weight samples $\{w_k\}_{k\in[K]}$, is no worse than the empirical risk, which can be seen as the training error, by an additional term that is of $\mathcal{O}(1/\sqrt{N})$.

3.4.3 Identifiability Analysis for IMOP

In this section, we propose the concept of identifiability in the context of decision making problems with multiple objectives, and show its correlation with the performance of our model for IMOP.

Definition 3.4.4 (Hausdorff semi-distance)**.** Let X and Y be two nonempty set. We define their *Hausdorff semi-distance* by

$$d_{sH}(X, Y) = \sup_{x\in X} \inf_{y\in Y} d(x, y). \tag{3.26}$$

Clearly, $d_{sH}(X, Y) = 0$ if $X = Y$. Nevertheless, $d_{sH}(X, Y) = 0$ does not always lead to $X = Y$.

Lemma 3.4.17. $d_{sH}(X, Y) = 0$ if and only if $X \subseteq Y$.

We are now ready to state our definition of Identifiability in the context of 3.1.

Definition 3.4.5 (Identifiability)**.** A 3.1 is said to be identifiable at $\theta \in \Theta$, if for all $\theta' \in \Theta\backslash\theta$,

$$d_{sH}(X_P(\theta), X_P(\theta')) > 0. \tag{3.27}$$

Intuitively, a 3.1 is identifiable if its Pareto optimal set can not be covered by that of any other DMP with parameter in Θ. More precisely, $X_P(\theta)$ is not a subset of $X_P(\theta')$ for any $\theta' \in \Theta \setminus \theta$.

3.4.3.1 Estimation Consistency of IMOP under Identifiability Let θ_0 be the underlying parameter of the 3.1 that generates the data. If 3.1 is identifiable at θ_0, and the data is not corrupted by noise, then $M(\theta)$ achieves its minimum uniquely at θ_0. We are now ready to state our result regarding the estimation consistency of $\hat{\theta}_K^N$.

Theorem 3.4.18 (Consistency of $\hat{\theta}_K^N$)**.** Suppose Assumptions 3.2.1 - 3.4.1 hold. Suppose also that $\mathbf{f}(\mathbf{x}, \theta)$ is strongly convex in $\mathbf{x}$ for each $\theta \in \Theta$, and that $\forall \mathbf{y} \in \mathcal{Y}, \mathbf{y} \in X_P(\theta_0)$. That is, there is no noise in the data. If 3.1 is identifiable at $\theta_0 \in \Theta$, then $\hat{\theta}_K^N \xrightarrow{P} \theta_0$.

On top of the ability of inferring parameters in 3.1, we would like to point out that our model for IMOP has an additional benefit of learning the distribution of decision makers' preferences.

By solving 3.9, we obtain not only an estimation of θ and $\{\mathbf{x}_k\}_{k\in[K]}$, but also the value of z_{ik} for each $i \in [N]$ and $k \in [K]$. We group all those noisy decisions with $z_{ik} = 1$ among $\{\mathbf{y}_i\}_{i\in[N]}$ to the cluster C_k for each $k \in [K]$. For the cluster C_k, all the noisy decisions share the same preference over objective functions. More precisely, we let w_k, the kth weight sample, represent the preference of the decision makers in C_k over multiple objective functions. Here, one latent assumption we make is that decision makers in the same cluster are homogeneous in their preferences for different objectives. Next, we propose the concept of the bijectivity of a 3.1 to support the performance analysis of the inferred preference.

Definition 3.4.6 (Bijectivity)**.** A 3.1 is said to be bijective at $\theta \in \Theta$ if $X_P(\theta) = \bigcup_{w\in\mathscr{W}_p} S(w, \theta)$, $S(w, \theta)$ is single valued for w almost surely, and $\forall w_1, w_2 \in \mathscr{W}_p$, $w_1 \neq w_2$ implies $S(w_1, \theta) \neq S(w_2, \theta)$.

With a slight abuse of notation, we let $w_{\mathbf{y}}$ be the true weight for $\mathbf{y}$, and $w_{\mathbf{y}}^{NK}$ be the estimated weight for $\mathbf{y}$ given $\hat{\theta}_K^N$. More precisely, $w_{\mathbf{y}}^{NK} = \arg\min_{w_k:k\in[K]}\{l_K(\mathbf{y}, \hat{\theta}_K^N)\}$. The following theorem shows that the inferred preference converges in probability to the true preference if the 3.1 we investigate enjoys the identifiability and the bijectivity defined above.

Theorem 3.4.19 (Consistency of $w_{\mathbf{y}}^{NK}$)**.** Suppose the same conditions of Theorem 3.4.18 hold. If 3.1 is bijective at θ_0, then $\|w_{\mathbf{y}} - w_{\mathbf{y}}^{NK}\|_2 \xrightarrow{P} 0$ for $\mathbf{y} \in \mathcal{Y}$ almost surely.

3.4.3.2 Non-identifiability of a Decision Making Problem A 3.1 might be non-identifiable in various ways. One trivial non-identifiability occurs due to scaling or permuting the component functions in $\mathbf{f}(\mathbf{x}, \theta)$ or $\mathbf{g}(\mathbf{x}, \theta)$. Nevertheless, this is not a serious problem in practice because some components of $\mathbf{f}(\mathbf{x}, \theta)$ or $\mathbf{g}(\mathbf{x}, \theta)$ might be known a priori, which helps avoid the occurrence of such non-identifiability. Otherwise, such non-identifiability could be prevented by normalizing certain components of the parameter before solving 3.9.

A more subtle non-identifiability issue occurs when different 3.1s have the same Pareto optimal set, as shown by the following two examples. Under such circumstances, one could not tell which program generates the observed decisions provided that no extra information is available.

Example 3.4.2.

$$\begin{aligned} \min \quad & \begin{pmatrix} x_1^2 + 2x_2^2 + 6x_1 + 2x_2 \\ 2x_1^2 + x_2^2 - 12x_1 - 10x_2 \end{pmatrix} \\ \text{s.t.} \quad & 3x_1 - x_2 \leq 6, \\ & x_2 \leq 3, \\ & x_1, x_2 \geq 0. \end{aligned} \tag{3.28}$$

Example 3.4.3.

$$\begin{aligned} \min \quad & \begin{pmatrix} 7x_1^2 + 11x_2^2 + 19x_1 \\ 12x_1^2 + 6x_2^2 - 72x_1 - 60x_2 \end{pmatrix} \\ \text{s.t.} \quad & 3x_1 - x_2 \leq 6, \\ & x_2 \leq 3, \\ & x_1, x_2 \geq 0. \end{aligned} \tag{3.29}$$

Proposition 3.4.20. EXAMPLE 3.4.2 and EXAMPLE 3.4.3 have the same Pareto optimal set.

We plot the two Pareto optimal sets in Figure 10. One can see that the two examples share the same Pareto optimal set. Suppose no restrictions on the variables x_1 and x_2, we obtain a set of points that consists of the optimal solution of 3.2 for each $w \in [0, 1]$. We call it the solution path for 3.1. To further illustrate why these two examples share the same Pareto optimal set, we plot the solution paths for both of them in Figure 10. It shows that solution path 2 is covered by solution path 1. Note that both solution paths have points lying outside of the feasible region. These points are rendered to become the same Pareto optimal solutions on the boundary of the feasible region, which explains why two 3.1s with different solution paths have the same Pareto optimal set.

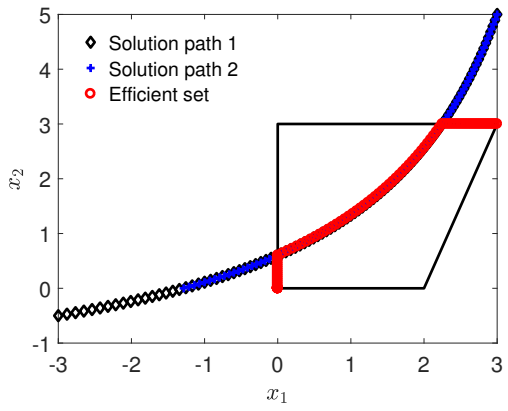


Figure 10: The Black Diamond Dots Represent the Solution Path of Example 3.4.2. The Blue "+" Dots Show the Solution Path of Example 3.4.3. The Red Circle Dots Indicate the Pareto Optimal Set for Both Examples.

3.4.3.3 Test Non-identifiability of a Decision Making Problem As shown in previous section, non-identifiability of a 3.1 occurs in various ways. It occurs even when 3.1 is strongly convex, which would bring serious problems to the inference of parameters. Therefore, it is necessary for us to provide a systematic procedure to test whether a 3.1 is identifiable or not. To achieve this goal, we first introduce the test problem in the following.

$$\begin{aligned} \max_{\theta\in\Theta} \quad & \|\theta - \hat{\theta}_K^N\|_1 \\ \text{s.t.} \quad & \mathbf{x}_i \in \bigcup_{k\in[K']} S(w_k, \theta) \quad \forall i \in [N'], \end{aligned} \tag{3.30}$$

where $\hat{\theta}_K^N$ is an optimal solution of 3.9, and $\{\mathbf{x}_i\}_{i\in[N]'}$ are the Pareto optimal points on $X_P(\hat{\theta}_K^N)$ which could be obtained a priori by solving 3.2 with a set of weights $\{w_i\}_{i\in[N']}$. Indeed, 3.30 seeks to find the furthest θ to $\hat{\theta}_K^N$ that still keeps $X_P(\hat{\theta}_K^N)$ Pareto optimal. Thus, the test statistic could be the optimal value z_{test} of 3.30, where $z_{test} > 0$ suggests that there might exist multiple parameters keeping $X_P(\hat{\theta}_K^N)$ Pareto optimal, and that 3.1 is non-identifiable.

We need three sets of weight samples to solve 3.30. The first set of weight samples $\{w_k\}_{k\in[K]}$ is used in 3.9. Once obtaining $\hat{\theta}_K^N$, we use the second set of weight samples $\{w_i\}_{i\in[N']}$ to generate the Pareto optimal points on $X_P(\hat{\theta}_K^N)$. The third set of weight samples

$\{w_k\}_{k\in[K']}$ is used to find the furthest θ to $\hat{\theta}_K^N$ that keeps $\{\mathbf{x}_i\}_{i\in[N']}$ Pareto optimal. These three sets of weights do not necessarily be the same. Since the weighting problem 3.2 is a convex program and thus is the easiest one among the three problems, $\{w_i\}_{i\in[N']}$ should be the largest set. In addition, 3.9 is the most difficult one to solve, and thus $\{w_k\}_{k\in[K]}$ should be the smallest set.

Suppose $\mathbf{f}(\mathbf{x},\theta)$ and $\mathbf{g}(\mathbf{x},\theta)$ are smooth in $\mathbf{x}$, we can reformulate the 3.30 by replacing the optimal set $S(w_k,\theta)$ with strong duality or its KKT conditions and using binary variables to indicate the inclusion relationship between $\mathbf{x}_i$ and $S(w_k,\theta)$. The reformulation is given in APPENDIX C.4.1. The test process is formally presented in Algorithm 2. We emphasize that this procedure becomes more accurate when more data are available.

Algorithm 2 Test Non-identifiability of a Decision Making Problem

1: Choose weight samples $\{w_k\}_{k\in[K]}$. Solve 3.9. Denote $\hat{\theta}_K^N$ the optimal solution.
2: Choose a new set of weight samples $\{w_i\}_{i\in[N']}$. Generate $|N'|$ Pareto optimal points on $X_P(\hat{\theta}_K^N)$ by solving 3.2. Namely, $\mathbf{x}_i \in S(w_i,\hat{\theta}_K^N)$ for each $i\in[N']$.
3: Choose another set of weight samples $\{w_k\}_{k\in[K']}$. Solve 3.30. Let the test statistic be the optimal value z_{test}.
4: If $z_{test}\neq 0$, we believe 3.1 is non-identifiable based on the data.

3.5 Solution Approaches to 3.9

The most natural approach to solve 3.9 is to transform it into a single level optimization problem by replacing the constraints $\mathbf{x}_k \in S(w_k,\theta)$ with optimality conditions [86]. In general, there are at least three ways to achieve this. One way is to replace $\mathbf{x}_k \in S(w_k,\theta)$ by the variational inequalities, the second way is to employ the strong duality theorem of convex optimization, and the third way is to replace $\mathbf{x}_k \in S(w_k,\theta)$ by the KKT conditions. Note that the first and second ways will introduce product terms of the upper level decision variables (i.e., θ) and lower level decision variables (i.e., $\mathbf{x}_k$), making the reformulated problems extremely difficult to solve by State-of-Art solvers. Nevertheless, the third approach would

avoid such a situation since the complementary constraints in KKT conditions can be linearized, and the resulting single level reformulation can be solved efficiently by State-of-Art mixed integer nonlinear programming solvers as demonstrated later in section 3.6. Hence, we will present our solution approaches based on the reformulations using KKT conditions.

Due to the NP-hardness shown in Theorem 3.3.3, solving 3.9 exactly with large size or high dimensional data set is practically infeasible. To tackle this challenge, we develop an expectation-maximization (EM)-style clustering-type algorithm that guarantees to converge to a (local) optimal solution. Moreover, this algorithm can be enhanced through incorporating manifold learning. Finally, we present a fast heuristic algorithm based on alternating direction method of multipliers (ADMM) to solve the Maximization step of the clustering-type algorithm. Although ADMM requires solving a much smaller problem many times, experimental results in section 3.6 indeed shows that it provides a tractable approach for solving 3.9. Since ADMM is not the main contribution of our paper, we put it in Appendix C.3.1. The general framework of IMOP and these algorithms is presented in Figure 11.

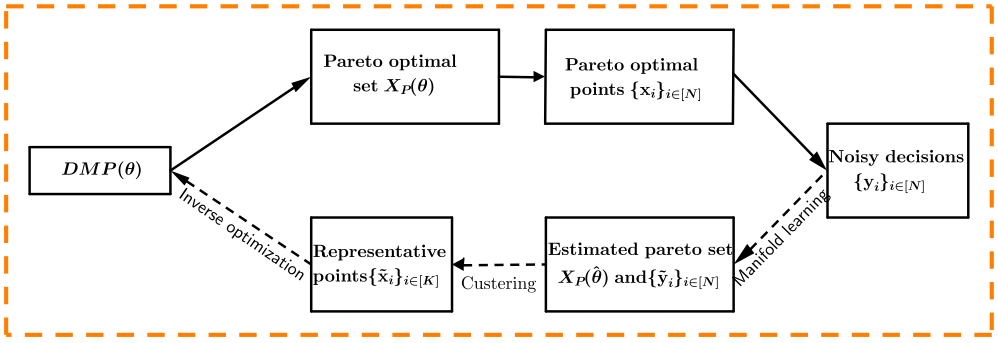


Figure 11: The Framework of Solving IMOP through Manifold Learning and Clustering.

3.5.1 Solving IMOP through a Clustering-type Approach

We provide in section 3.3.1 deep insights on the connections between 3.9 and the K-means clustering problem. Leveraging these insights, we develop an efficient clustering-type algorithm to solve 3.9. Clearly, in both 3.9 and 3.12, one needs to assign $\{\mathbf{y}_i\}_{i\in[N]}$ to certain clusters in such a way that the average squared distance between $\mathbf{y}_i$ and its closest $\mathbf{x}_k$ is

minimized. The difference is whether $\mathbf{x}_k$ has restriction or not. In 3.9, each $\mathbf{x}_k$ is restricted to belong to $S(w_k, \theta)$, while there is no restriction for $\mathbf{x}_k$ in 3.12. As such, each $\mathbf{x}_k$ in 3.12 is the centroid of the observations in the kth cluster. Nevertheless, we will show in the following that the centroid of cluster k is closely related to $\mathbf{x}_k$ in 3.9 for each $k \in [K]$. More precisely, we are able to obtain $\mathbf{x}_k$ given only the centroid and the number of observations in each cluster.

For each $k \in [K]$, we denote C_k the set of noisy decisions with $z_{ik} = 1$ after solving 3.9 to optimal. That is, observations in C_k are closest to $\mathbf{x}_k$. Consequently, we partition $\{\mathbf{y}_i\}_{i\in[N]}$ into K clusters $\{C_k\}_{k\in[K]}$. Let $\overline{\mathbf{y}}_k = \frac{1}{|C_k|}\sum_{\mathbf{y}_i \in C_k} \mathbf{y}_i$ be the centroid of cluster C_k, and denote $Var(C_k)$ the variance of C_k. Through an algebraic calculation, we get

$$M_K^N(\theta) = \frac{1}{N}\sum_{i\in[N]} \|\mathbf{y}_i - \sum_{k\in[K]} z_{ik}\mathbf{x}_k\|_2^2 = \frac{1}{N}\sum_{k\in[K]} |C_k|\bigg(\|\overline{\mathbf{y}}_k - \mathbf{x}_k\|_2^2 + Var(C_k)\bigg). \tag{3.31}$$

Note that $\{Var(C_k)\}_{k\in[K]}$ is a set of fixed values when clusters $\{C_k\}_{k\in[K]}$ are given. If we know the clusters $\{C_k\}_{k\in[K]}$ beforehand, we see in (3.31) that K centroids $\{\overline{\mathbf{y}}_k\}_{k\in[K]}$ and $\{|C_k|\}_{k\in[K]}$ are enough to solve 3.9. This is the key insight we leverage to solve 3.9. However, similar to K-means clustering, $\{C_k\}_{k\in[K]}$ are not known a priori. In K-means clustering algorithm [19], this problem is solved by initializing the clusters, and then iteratively updating the clusters and centroids until convergence. Similarly, we propose a procedure that alternately clusters the noisy decisions (assignment step) and find θ and $\{\mathbf{x}_k\}_{k\in[K]}$ (update step) until convergence. Given θ and $\{\mathbf{x}_k\}_{k\in[K]}$, the assignment step can be done easily as we discussed previously. Moreover, the update step can be established by solving the problem as follows.

$$\begin{aligned}
\min_{\theta, \mathbf{x}_{k'}} \quad & \tfrac{1}{N}\sum_{k\in[K]} |C_k| \|\overline{\mathbf{y}}_k - \textstyle\sum_{k'\in[K]} z_{kk'}\mathbf{x}_{k'}\|_2^2 & \\
\text{s.t.} \quad & \mathbf{x}_{k'} \in S(w_{k'}, \theta), & \forall k' \in [K], \\
& \sum_{k'\in[K]} z_{kk'} = 1, & \forall k \in [K], \\
& z_{kk'} \in \{0, 1\}, & \forall k \in [K],\ k' \in [K].
\end{aligned} \tag{3.32}$$

The expectation-maximization (EM)-style algorithm is formally presented in the following.

Algorithm 3 Solving 3.9 through a Clustering-based Approach

1: **Input:** Noisy decisions $\{\mathbf{y}_i\}_{i\in[N]}$, weight samples $\{w_k\}_{k\in[K]}$.

2: **Initialization**: Partition $\{\mathbf{y}_i\}_{i\in[N]}$ into K clusters using K-means clustering. Calculate $\{\overline{\mathbf{y}}_k\}_{k\in[K]}$. Solve 3.32 and get an initial estimation of θ and $\{\mathbf{x}_k\}_{k\in[K]}$.

3: **while** stopping criterion is not satisfied **do**

4: **Assignment step**: Assign each $\mathbf{y}_i$ to the closest $\mathbf{x}_k$ to form new clusters. Calculate their centroids $\{\overline{\mathbf{y}}_k\}_{k\in[K]}$.

5: **Update step**: Update θ and $\{\mathbf{x}_k\}_{k\in[K]}$ by solving 3.32.

6: **end while**

7: **Output:** An estimate of the parameter of 3.1. Denote it by $\hat{\theta}_C$.

Remark 3.5.1. (i) In practice, we would apply one of the following as the stopping criterion: cluster assignments do not change; or, the maximum number of iterations is reached. (ii) In **Initialization** step, we take K-means++ algorithm [20] as the default clustering method, run it multiple times and select the centroids of the best clustering results to further solve 3.32. (iii) In the **Assignment step**, note that we only handle non-empty clusters and break ties consistently, e.g., by assigning an observation $\mathbf{y}_i$ to the cluster with the lowest index if there are several equidistant $\mathbf{x}_k$. Otherwise, the algorithm can cycle forever in a loop of clusters that have the same cost. (iv) In the **Update step**, 3.32 can be solved either by directly computing the KKT based single level reformulation or by applying the ADMM approach shown in Appendix C.3.1 We note that 3.32 indeed can be solved efficiently by State-of-Art mixed integer nonlinear programming solvers as demonstrated in the experiments.

Since 3.9 is non-convex, there may exist multiple local optimal solutions. Nevertheless, we will show that Algorithm 3 indeed converges to a (local) optimal solution in finite steps. The key step of the proof is established in the following lemma.

Lemma 3.5.1. Both the **Assignment step** and the **Update step** in Algorithm 3 decrease $M_K^N(\theta)$.

Theorem 3.5.2 (Finite convergence)**.** Suppose there is an oracle to solve 3.32. Algorithm 3 converges to a (local) optimal solution of 3.9 in a finite number of iterations.

Proof. Since there is at most K^N ways to partition $\{\mathbf{y}_i\}_{i\in[N]}$ into K clusters, the monotonically decreasing Algorithm 3 will eventually arrive at a (local) optimal solution in finite steps. □

Remark 3.5.2. (i) In practice, Algorithm 3 converges pretty fast, typically within several iterations. The main reason is that the **Initialization** step often provides a good estimation of the true parameter, since the K centroids returned by K-means clustering represent the observations well in general, especially when K is large. (ii) Algorithm 3 is extremely Pareto optimal in computation especially when $N \gg K$. The reason is that in each iteration only K representative points (i.e., the centroids of clusters) are used to update θ, instead of the whole batch of observations.

3.5.2 An Enhanced Algorithm for Solving IMOP with Manifold Learning

We provide another algorithm leveraging the connection shown in section 3.3.2 that the Pareto optimal set is a piecewise continuous manifold with intrinsic dimension of $p-1$, where p is the number of objectives, regardless of the dimension of the decision space.

Algorithm 4 Solving 3.9 with manifold learning and clustering

1: **Input:** Noisy decision $\{\mathbf{y}_i\}_{i\in[N]}$, evenly sampled weights $\{w_k\}_{k\in[K]}$.
2: Apply nonlinear manifold learning algorithm for $\{\mathbf{y}_i\}_{i\in[N]}$. Get low dimensional points $\{\mathbf{x}_i\}_{i\in[N]}$, where $\mathbf{x}_i \in \mathbb{R}^{p-1}$.
3: Group $\{\mathbf{x}_i\}_{i\in[N]}$ into K clusters using 3.12.. Denote I_K the set of labels of $\{\mathbf{x}_i\}_{i\in[N]}$. Find K centroids of $\{\mathbf{y}_i\}_{i\in[N]}$ according to I_K. Denote $\{C_k\}_{k\in[K]}$ these centroids.
4: Run Algorithm 3 with $\{C_k\}_{k\in[K]}$ and $\{w_k\}_{k\in[K]}$.
5: **Output:** An estimate of the parameter θ of 3.1.

Remark 3.5.3. (i) Linear manifold learning methods, such as Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA), perform well when there exists a linear structure in the data. However, applying them in step 2 might not be appropriate since our data has a non-linear structure by Theorem 3.3.5, even for the simplest linear case stated in Corollary 3.3.5.1. Hence, we focus on nonlinear manifold learning methods. (ii) Similar to

Algorithm 3, we run K-means++ algorithm multiple times and select the centroids of the best clustering results in step 3.

3.6 Computational Experiments

In this section, we illustrate the performances of the proposed algorithms on a multi-objective linear program (MLP), two multiobjective quadratic programs (MQP) and one general multiobjective nonlinear program. Our experiments have been run on Bridges system at the Pittsburgh Supercomputing Center (PSC) [61]. The mixed integer second order conic problems (MISOCP) are solved with Gurobi [87]. All the algorithms are programmed with Julia [62] unless otherwise specified. All the single level reformulations of the IMOP are given in Appendix C.4. Throughout this section we use **SR** to refer that we solve these single level reformulations to optimality using Gurobi.

3.6.1 Learning the Objective Functions of an MLP

Consider the following Tri-objective linear programming problem:

$$\begin{aligned} \min \quad & \{-x_1, -x_2, -x_3\} \\ s.t. \quad & x_1 + x_2 + x_3 \leq 5, \\ & x_1 + x_2 + 3x_3 \leq 9, \\ & x_1, x_2, x_3 \geq 0. \end{aligned} \tag{3.33}$$

In this example, there are two efficient faces, one is the triangle defined by vertices $(2, 4, 5)$, the other one is the tetragon defined by vertices $(1, 3, 5, 4)$ as shown by Figure 12.

We generate the data as follows. First, $N = 10000$ Pareto optimal points $\{\mathbf{x}_i\}_{i\in[N]}$ are uniformly sampled on faces $(2, 4, 5)$ and $(1, 3, 5, 4)$. Next, the observations $\{\mathbf{y}_i\}_{i\in[N]}$ are obtained by adding noise to each Pareto optimal point, where the noise has a jointly normal distribution with zero mean and 0.5^2 units identity covariance. Namely, $\mathbf{y}_i = \mathbf{x}_i + \epsilon_i$, where $\epsilon_i \sim \mathcal{N}(\mathbf{0}_3, 0.5^2\mathbf{I}_3)$ for each $i \in [N]$. We assume that the parameters to be learned are

non-positive. In addition, we add the normalization constraints $\mathbf{1}^T\mathbf{c}_1 = -1, \mathbf{1}^T\mathbf{c}_2 = -1$ and $\mathbf{1}^T\mathbf{c}_3 = -1$ to prevent the arise of trivial solutions, such as $\mathbf{c}_1 = \mathbf{c}_2 = \mathbf{c}_3 = [0, 0, 0]^T$. Then, we uniformly choose the weights $\{w_k\}_{k\in[K]}$ such that $w_k \in \mathscr{W}_3$ for each $k \in [K]$. Here, we set $K = 81$.

Algorithms 3 - 4 are used to solve 3.9. In Algorithm 3, we run K-means++ algorithm 10 times to find the best clustering results. Centroids of the $K = 81$ clusters are plotted in Figure 12b. In Algorithm 4, we use Kernel PCA [88] to project the data into a 2-dimension space, and then apply K-means++ clustering algorithm to find $K = 81$ clusters. Centroids of the $K = 81$ clusters are plotted in Figure 12c. As shown in Figures 12b - 12c, Algorithm 4 provides the better estimation of the manifold before solving 3.9 than Algorithm 3. Nevertheless, both solve 3.9 as they all recover the true Pareto optimal set even with the initial estimation of the parameter in the Initialization step. Thus, we won't run the later steps in Algorithm 3. The estimating results using Algorithm 3 are $\hat{\mathbf{c}}_1 = [0, 0, -1]^T$, $\hat{\mathbf{c}}_2 = [-0.3333, -0.3333, -0.3333]^T$ and $\hat{\mathbf{c}}_3 = [-0.2871, -0.2871, -0.4258]^T$ and $\hat{\mathbf{c}}_3 = [-0.2871, -0.2871, -0.4258]^T$. The estimating results using Algorithm 4 are $\hat{\mathbf{c}}_1 = [-0.4, -0.4, -0.2]^T$, $\hat{\mathbf{c}}_2 = [-0.2, -0.2, -0.6]^T$ and $\hat{\mathbf{c}}_3 = [-0.3333, -0.3333, -0.3333]^T$.

Given the estimation $\hat{\mathbf{c}}_1 = [0, 0, -1]^T$, $\hat{\mathbf{c}}_2 = [-0.3333, -0.3333, -0.3333]^T$ and $\hat{\mathbf{c}}_3 = [-0.2871, -0.2871, -0.4258]^T$, we apply Algorithm 2 to test whether this example is identifiable or not. Step 1 is omitted since it has been completed in the previous experiment. In Step 2, we randomly sample $|N'| = 200$ points from the Pareto optimal. In Step 3, we uniformly generate $|K'| = 200$ weights. In Step 4, we replace the optimal set $S(w_k, \theta)$ by KKT conditions and solve the 3.30 , and it achieves the maximum value when $\mathbf{c}_1 = [-0.4778, -0.4994, -0.0228]^T, \mathbf{c}_2 = [0.0, -0.0217 - 0.9783]^T, \mathbf{c}_3 = [-0.9556, -0.0444, 0.0]^T$. The test statistic $z_{test} = 4.5813$, which is greater than 0. Thus, we claim that this example is non-identifiable.

3.6.2 Learning the Preferences and Constraints of an MQP

We consider the following multiobjective quadratic programming problem.

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^2_+} \quad & \begin{pmatrix} f_1(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T Q_1 \mathbf{x} + \mathbf{c}_1^T \mathbf{x} \\ f_2(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T Q_2 \mathbf{x} + \mathbf{c}_2^T \mathbf{x} \end{pmatrix} \\ s.t. \quad & A\mathbf{x} \geq \mathbf{b}, \end{aligned} \tag{3.34}$$

where parameters of the objective functions and the constraints are

$$Q_1 = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}, \mathbf{c}_1 = \begin{bmatrix} 3 \\ 1 \end{bmatrix}, Q_2 = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}, \mathbf{c}_2 = \begin{bmatrix} -6 \\ -5 \end{bmatrix}, A = \begin{bmatrix} -3 & 1 \\ 0 & -1 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} -6 \\ -3 \end{bmatrix}. \tag{3.35}$$

3.6.2.1 Learning the Right-hand Side of Constraints In the first set of experiments, suppose the right-hand side $\mathbf{b}$ is unknown, and the learner seeks to learn $\mathbf{b}$ given the noisy decisions she observes. Assume that $\mathbf{b}$ is within the range $[-8,-1]^2$. We generate the data as follows. We first compute Pareto optimal solutions $\{\mathbf{x}_i\}_{i\in[N]}$ by solving 3.2 with weight samples $\{w_i\}_{i\in[N]}$ that are uniformly chosen from $\mathscr{W}_2$. Next, the noisy decision $\mathbf{y}_i$ is obtained by adding noise to $\mathbf{x}_i$ for each $i \in [N]$. More precisely, $\mathbf{y}_i = \mathbf{x}_i + \epsilon_i$, where each element of ϵ_i has a truncated normal distribution supported on $[-1,1]$ with mean 0 and standard deviation 0.1 for all $i \in [N]$.

Both the SR approach and the ADMM approach (Algorithm 6) are applied to solve for $\mathbf{b}$ with different N and K. The basic parameters for the implementation of the ADMM approach are given in the following. The observations are equally partitioned into $T = N/2$ groups. We pick the penalty parameter $\rho = 0.5$ as the best out of a few trials. We use the initialization $\mathbf{b}^0 = \mathbf{v}^{t,0} = \mathbf{0}_2$ for the iterations. The tolerances of the primal and dual residuals are set to be $\epsilon^{pri} = \epsilon^{dual} = 10^{-3}$. We find that Algorithm 6 converges in 100 iterations in general, thus the termination criterion is set to be either the norms of the primal and dual residuals are smaller than 10^{-3} or the iteration number k reaches 100.

In Figure 13 we summarize the computational results averaged over 10 repetitions of the experiments for each N and K using Algorithm 6. Note that $\mathbf{b}_{true} = [-3,-6]^T$. The smaller the estimator error is, the closer is $\hat{\mathbf{b}}$ to $\mathbf{b}_{true}$. Note that 3.9 is prediction consistent by Theorem 3.4.13 for this example. The results in Figure 13 show the estimation consistency

of the 3.9 as the estimation error decreases to zero with the increase of the data size N and weight sample size K, although it does not satisfy the conditions for Theorem 3.4.18. Note that estimation consistency implies risk consistency. Thus, this result illustrates Theorem 3.4.13. Also, we see that the estimation error becomes more stable when using more weight samples, i.e., K becomes larger. In Tables 4- 5, we summarize the computational time that averages over 10 repetitions of the experiments for each algorithm, N and K. Here p-ADMM means that we implement the θ^t-update step of ADMM in parallel with 28 cores. $*$ means that we can not get reasonable estimation of the parameter within three hours. As shown in these tables, both ADMM and p-ADMM approaches dramatically improve the computational efficacy over the SR approach when N and K are large. On average, p-ADMM is two times faster than ADMM. Moreover, the SR approach could handle only small size problems with roughly $N \leq 20$ and $K \leq 11$. To further illustrate the performance of the ADMM algorithm, we plot the primal and dual residuals versus the iteration number in each of the 100 repetitions for $N = 20, K = 21$, and the estimation error versus the iteration number in Figures 14a and 14b, respectively. The two figures show that the ADMM approach converges within 100 iterations under the above setting.

3.6.2.2 Learning the Objective Functions In the second set of experiments, suppose $\mathbf{c}_1$ and $\mathbf{c}_2$ are unknown, and the learner seeks to learn them given the noisy decisions. Assume that $\mathbf{c}_1$ and $\mathbf{c}_2$ are within range $[-10, 10]^2$. We generate the data in a way similar to the first set of experiments. The only difference is that each element of the noise has a uniform distribution supporting on $[-0.25, 0.25]$ with mean 0 for all $i \in [N]$.

We would like to use Algorithm 3 to solve large-scale 3.9. We note that the SR approach can not handle cases when $N \geq 10$ and $K \geq 11$ in the **Update step**. Hence, the ADMM approach (Algorithm 6) is applied to solve 3.32. The stopping criterion for Algorithm 3 is that the maximum iteration number reaches five. In the **Initialization step**, we run K-means++ algorithm 50 times to find the best clustering results. When solving 3.32 using ADMM, we partition the observations in such a way that each group has only one observation. We pick the penalty parameter $\rho = 0.5$ as the best out of a few trials. We use the initialization $\mathbf{c}_1^0 = \mathbf{c}_2^0 = \mathbf{v}_1^{t,0} = \mathbf{v}_2^{t,0} = \mathbf{0}_2$ for the iterations. The tolerances of the primal and dual residuals

are set to be $\epsilon^{pri} = \epsilon^{dual} = 10^{-3}$. The termination criterion is that either the norms of the primal and dual residuals are smaller than 10^{-3} or the iteration number k reaches 50.

In Figure 15a, we report the prediction errors averaged over 10 repetitions of the experiments for different N and K. Here, we use an independent validation set that consists of 10^5 noisy decisions generated in the same way as the training data to compute the prediction error. We also calculate the prediction error using the true parameter and $M(\theta_{true}) = 0.022742$. More precisely, we evenly generate $K = 10^4$ weight samples and calculate the associated Pareto optimal solutions on the true Pareto optimal set. These Pareto optimal solutions are then used to find the prediction error of the true parameter. We observe that the prediction error has the trend to decrease to $M(\theta_{true})$ with the increase of the data size N and weight sample size K. This makes lots of sense because 3.9 is risk consistent by Theorem 3.4.13 for this example. To further illustrate the performance of the algorithm, we plot the change of assignments versus iteration in the **Assignment step** over 10 repetitions of the experiments with $N = 5 \times 10^4, K = 21$ in Figure 15b. One can see the assignments become stable in 5 iterations, indicating the fast convergence of our algorithm. Also, we plot the estimated Pareto optimal set with $N = 5 \times 10^4, K = 21$ in the first repetition in Figure 15c. Here, $\hat{\mathbf{c}}_1 = [2.0023, 0.0454]^T$ and $\hat{\mathbf{c}}_2 = [-5.7197, -4.6949]^T$. They are not equal to the true parameters as this MQP is non-identifiable. However, our method still recovers the unknown parameters quite well as the estimated Pareto optimal set almost coincides with the true one.

We also plot our prediction of the distribution for the preferences of $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$. Since there are only two objective functions, it is sufficient to draw the distribution of the weight for $f_1(\mathbf{x})$ (given that weights of $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ summing up to 1). As shown in Figure 15d, except in the two endpoint areas, the number of noisy decisions assigned to each weight follows roughly uniformly distribution, which matches our uniformly sampled weights. Indeed, we would like to point out that a *boundary effect* probably occurs in these two endpoint areas. Although different weights are imposed on component functions, the noiseless optimal solutions, as well as observed decisions, do likely to merge together due to the limited feasible space in those areas. We believe that it reflects an essential challenge in learning multiple objective functions in practice and definitely deserves a further study.

We next compare the performance of Algorithm 3 and Algorithm 4. We find that the manifold learning based method generally performs better when the data has lots of noise. Specifically, in the third set of experiments, suppose $\mathbf{c}_1$ and $\mathbf{c}_2$ are unknown, and the learner seeks to learn them given the noisy decisions. Assume that $\mathbf{c}_1$ and $\mathbf{c}_2$ are within range $[-10, 10]^2$. We generate the data in a way similar to the previous two sets of experiments. The difference is that each element of the noise has a uniform distribution supporting on $[-1, 1]$ with mean 0 for all $i \in [N]$.

We report the estimation results in Table 6. Here, Laplacian means that we use Laplacian eigenmaps [89] to do the manifold learning. Similarly, KernelPCA stands for Kernel PCA [90]. Autoencoder stands for [91]. ManifoldChart stands for Manifold Charting [92]. NCA stands for Neighborhood Components Analysis [93]. For Algorithm 4, we try 5 common manifold leaning methods. We report the results for K up to 11 because both algorithms would recover the true Pareto optimal set quite well when $K = 21$. We find that Algorithm 4 with Laplacian, Autoencoder and ManifoldChart perform better than Algorithm 3 when $K = 6$ or 11, while Algorithm 4 using KernelPCA and NCA cannot beat Algorithm 3. This suggests that one should try different manifold learning algorithms when using Algorithm 4 In addition, noting that Algorithm 3 involes multiple iterations of solving 3.32. Thus, Algorithm 4 would take much less time than Algorithm 3.

To further illustrate the performance of the two algorithms, we plot the centroids obtained in two algorithms when $K = 11$ in Figure 16a and 16b , respectively. Figure 16b shows clearly that the principal points (centroids) in Algorithm 4 almost lie on and recover the true Pareto optimal set, while centroids in Algorithm 3 lie around the true Pareto optimal set. This explains why Algorithm 4 would give us better estimation results. Also, we can see in Figure 16b that the estimated Pareto optimal set almost coincides with the true Pareto optimal set.

3.6.3 Learning the Expected Returns in Portfolio Optimization

In this example, we consider various noisy decisions arising from different investors in a stock market. More precisely, we consider a portfolio selection problem, where investors need

to determine the fraction of their wealth to invest in each security in order to maximize the total return and minimize the total risk. The portfolio selection process typically involves the cooperation between an investor and a portfolio analyst, where the analyst provides an efficient frontier on a certain set of securities to the investor and then the investor selects a portfolio according to her preference to the returns and risks. The classical Markovitz mean-variance portfolio selection [17] in the following is often used by analysts.

$$\begin{aligned} \min_{\mathbf{x}} \quad & \begin{pmatrix} f_1(\mathbf{x}) & = -\mathbf{r}^T\mathbf{x} \\ f_2(\mathbf{x}) & = \mathbf{x}^T Q\mathbf{x} \end{pmatrix} \\ s.t. \quad & 0 \le x_i \le b_i, \qquad\qquad \forall i \in [n], \\ & \sum_{i=1}^{n} x_i = 1, \end{aligned} \tag{3.36}$$

where $\mathbf{r} \in \mathbb{R}^n_+$ is a vector of individual security expected returns, $Q \in \mathbb{R}^{n\times n}$ is the covariance matrix of securities returns, $\mathbf{x}$ is a portfolio specifying the proportions of capital to be invested in the different securities, and b_i is an upper bound put on the proportion of security $i \in [n]$.

In portfolio optimization, the forecast of security expected returns $\mathbf{r}$ is essential within the portfolio selection process. Note that different analysts might use different $\mathbf{r}$, which are due to different information sources and insights, to make recommendations. Consider a scenario that A observes that customers of B often make more revenues. Then, A might want to use our model to infer the $\mathbf{r}$ that B really uses.

We use the Portfolio data *BlueChipStockMoments* derived from real data in the Matlab Financial Toolbox. The true expected returns and true return covariances matrix for the first 8 securities are given in Appendix. W.L.O.G, we suppose that the expected returns for the last three securities are known. The data is generated as follows. We set the upper bounds for the proportion of the 8 securities to $b_i = 1.0, \forall i \in [8]$. We first generate optimal portfolios on the efficient frontier in Figure 17a by solving 3.2 with weight samples $\{w_i\}_{i\in[N]}$ chosen from $\mathscr{W}_2$. The first element of w_i, ranging from 0 to 1, follows a truncated normal distribution derived from a normal distribution with mean 0.5 and standard deviation 0.1. In what follows, we will not distinguish truncated normal distribution from normal distribution because their difference is negligible. Subsequently, each component of these portfolios is rounded to the nearest thousandth, which can be seen as measurement error.

Algorithm 3 is applied in this experiment. For a reason similar to the previous experiment, we use the ADMM approach (Algorithm 6) to solve 3.32. The stopping criterion for Algorithm 3 is that the maximum iteration number reaches five. In the **Initialization step**, we run K-means++ algorithm 50 times to find the best clustering result. When solving 3.32 using ADMM, we partition the observations in such a way that each group has only one observation. We pick the penalty parameter $\rho = 1$ as the best out of a few trials. We initialize $\mathbf{r}^0 = \mathbf{v}^{t,0} = \mathbf{0}_8$ for the iterations. The tolerances of the primal and dual residuals are set to be $\epsilon^{pri} = \epsilon^{dual} = 10^{-4}$. The termination criterion is that either the norms of the primal and dual residuals are smaller than 10^{-4} or the iteration number k reaches 10.

In Table 7, we list the estimation error averaged over 10 repetitions of the experiments for each N and K using Algorithm 6. The estimation error has the trend to becomes smaller when N and K increase, indicating the estimation consistency and thus risk consistency of the method we propose. To further illustrate the performance of our method, we plot the estimated efficient frontier which is very close to the real one as shown in Figure 17a. We also plot our estimation on the distribution of the weight of $f_1(\mathbf{x})$ among the noisy decisions. As shown in Figure 17b, the number of noisy decisions assigned to each weight follows a normal distribution with mean 0.5012 and standard deviation 0.1013. The 0.95 confidence intervals for the mean and standard deviation are $[0.4992, 0.5032]$ and $[0.0999, 0.1027]$, respectively. It is reasonable as we generate the portfolios by solving 3.2 with normally sampled weights and the feasible set of $\mathbf{x}$ is of a much weaker boundary effect, comparing to that in Section 3.6.2.2.

3.6.4 Learning the O-D Matrix

Let $G = (N, A)$ be a directed transportation network defined by a set N of nodes and a set A of directed links. Each link $a \in A$ has an associated flow-dependent travel time $t_a(v_a)$ that denotes the average travel time on each link. The travel time function $t_a(v_a)$ is assumed to be differentiable, convex, and monotonically increasing with the amount of flow v_a. Each link $a \in A$ also has an associated flow-dependent traffic emissions $e_a(v_a)$ that denotes the average traffic emissions on each link. Let W denote the set of O-D pairs, R_w denote the

set of all routes between the O-D pair $w \in W$, d_w represents the travel demand of O-D pair w, and f_r^w denote the traffic flow on the route r connecting the O-D pair w. $\delta_{ar}^w = 1$ if route $r \in R^w$ uses link a, and 0 otherwise.

We consider the following Bi-criteria traffic network system optimization problem of minimizing congestion and traffic emissions simultaneously [94]:

$$\begin{aligned} \min \quad & \begin{pmatrix} \sum_{a \in A} t_a(v_a) v_a \\ \sum_{a \in A} e_a(v_a) v_a \end{pmatrix} \\ s.t. \quad & d_w = \sum_{r \in R_w} f_r^w, && \forall w \in W, \\ & v_a = \sum_{w \in W} \sum_{r \in R_w} f_r^w \delta_{ar}^w, && \forall a \in A, \\ & v_a, f_r^w \geq 0, && \forall r \in R_w, w \in W. \end{aligned} \tag{3.37}$$

Note that the problem becomes a minimization of a weighted combination of congestion and traffic emissions if the external costs of congestion and emissions can be obtained. These costs change from time to time, which will lead to different link flows. We seek to learn the O-D matrix given the link flows under different values of time and monetary valuation of traffic emissions. In addition, the presence of measurement errors in the observed link flows are explicitly considered.

Fig 18 shows a road network with six nodes and seven links used in [95, 94]. The network has two O-D pairs $(1, 3)$ and $(2, 4)$, where $(1, 3)$ has the demand of 2500 vehicles per hour and $(2, 4)$ has the demand of 3500 vehicles per hour. We use the US Bureau of Public Road link travel time function to determine the travel time on each link. The function is of the form $t_a(v_a) = t_a^0(1 + 0.15 \cdot (v_a/C_a)^4)$, where t_a^0 and C_a are parameters representing the free-flow travel time (in minutes) and capacity (vehicles per hour) of link $a \in A$.

We follow the work [96] and assume the total emissions generated by the vehicles on link a is $e_a(v_a) = h_a v_a$, where h_a denotes the emission factor associated with link a. The key part in the estimation of vehicle emissions is that the volume of emissions equals to the product of emission factors times the link flow. The values of the parameters are listed in Table 8.

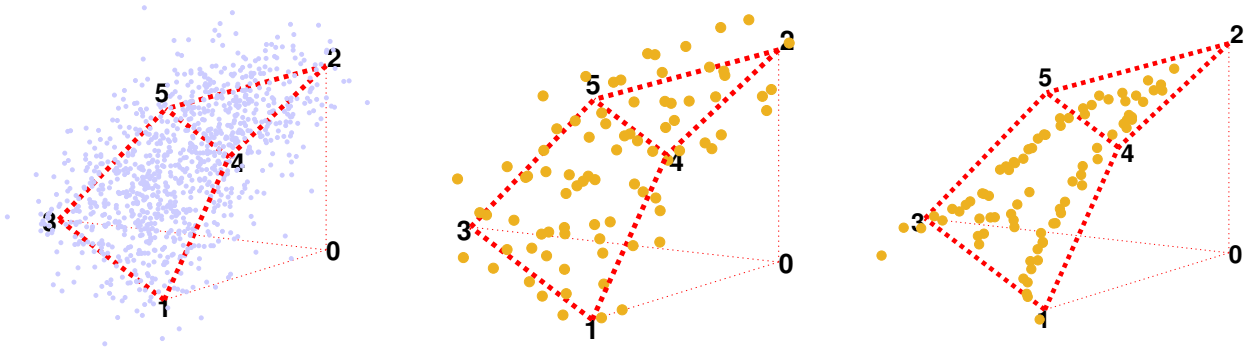
We generate the data as follows. We start by computing the efficient solutions $\{\mathbf{y}_i\}_{i \in [N]}$ using the weighted sum approach. The weights $\{w_i\}_{i \in [N]}$ are uniformly sampled such that

$w_i \in \mathscr{W}_2$ for each $i \in [N]$, where $N = 10$. Since we do not want to over emphasize either the congestion or the traffic emission in the bi-criteria traffic network system, we concentrate the weights and set $w_i \in [0.3, 0.7]^2$ for each $i \in [N]$. Subsequently, each component of the efficient solutions is rounded to the nearest ten, which can be treated as measurement error. We assume the demand of O-D pairs $(1, 3)$ and $(2, 4)$ are bigger than 1000 and smaller than 10000 vehicles per hour. Then, we evenly sample the weights $\{w_k\}_{k\in[K]}$ such that $w_k \in \mathscr{W}_2$ for each $k \in [K]$.

We implement the SRe approach using the solver FilMINT. The solutions returned by FilMINT are not guaranteed to be optimal since the inference of the O-D matrix involves solving a mixed integer nonconvex program. FilMINT can handle instances with $K \leq 100$ quite efficiently. In Table 9 we summarize the computational results for different K. The table lists for each K the estimations for the demands of O-D pairs $(1, 3)$ and $(2, 4)$, and also the estimation error, which is given by $\|\text{estimation - true O-D}\|_2/\|\text{true O-D}\|_2$. The table shows that the estimation error becomes smaller and smaller when K increases, indicating that our method still works for general convex MOP.

3.7 Conclusions

We study in this dissertation the problem of learning the objective functions and constraints of a multiobjective decision making problem, based on observations of efficient solutions which might carry noise. Specifically, we formulate such a learning task as an inverse multiobjective optimization problem, and provide a deep analysis to establish the statistical significance of the inference results from the presented model. Moreover, we discuss the strong correlation between the identifiability of the decision making problem and the performance of our inverse optimization model. We then develop two numerical algorithms to handle the computational challenge from the large number of observations. We confirm by extensive numerical experiments that the proposed algorithms can learn the parameters with great accuracy while dramatically improve the computational efficacy.



(a) Data (b) Algorithm 3 (c) Algorithm 4

Figure 12: Learning the Objective Functions of a Tri-objective Linear Program Using $N = 10000$ Observations. (a) The Light Blue Dots Indicate the 1000 Observations Randomly Selected From the Data Set. Two Pareto Optimal Faces are the Triangle $(2, 4, 5)$, and the Tetragon $(1, 3, 5, 4)$. (b) Orange Dots Indicate the Centroids After Using K-means Clustering. (C) Orange Dots Indicate the Centroids after Using Kernel PCA and K-means Clustering.

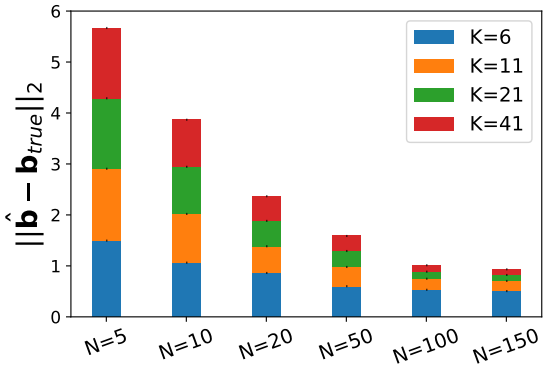


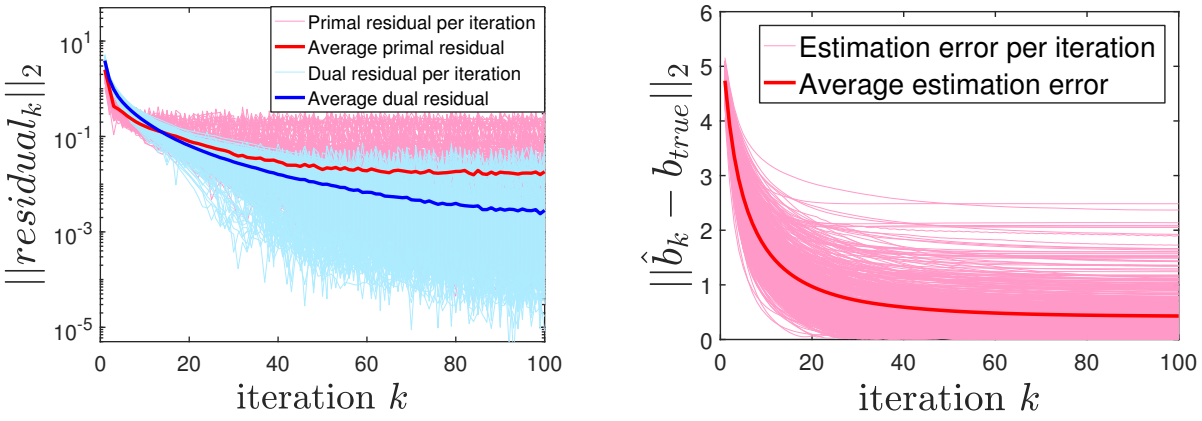
Figure 13: Estimation Error $\|\hat{\mathbf{b}} - \mathbf{b}_{true}\|_2$ for Different N and K

Table 4: Average Running Time over 10 Repetitions (In Seconds)

	$N = 5$			$N = 10$			$N = 20$		
	SR	ADMM	p-ADMM	SR	ADMM	p-ADMM	SR	ADMM	p-ADMM
$K = 6$	0.31	14.92	11.72	0.78	23.13	15.10	4.07	43.95	20.73
$K = 11$	0.42	20.93	12.83	3.10	33.88	19.43	705.36	66.91	28.95
$K = 21$	3.83	33.23	17.74	391.18	61.99	36.79	*	122.98	55.93
$K = 41$	38.42	59.67	31.69	*	156.78	107.48	*	343.72	205.98

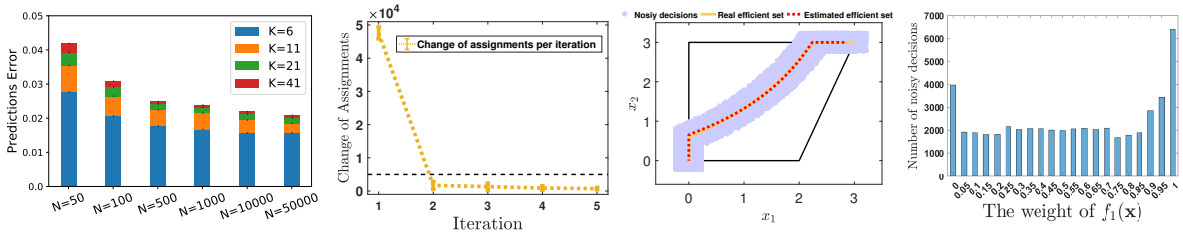
Table 5: Average Running Time over 10 Repetitions (In Seconds)

	$N = 50$			$N = 100$			$N = 150$		
	SR	ADMM	p-ADMM	SR	ADMM	p-ADMM	SR	ADMM	p-ADMM
$K = 6$	119.58	110.42	44.69	5423.19	222.19	87.45	*	335.90	131.73
$K = 11$	*	166.39	69.25	*	336.82	138.80	*	508.30	208.95
$K = 21$	*	306.91	141.22	*	613.08	278.28	*	923.58	418.27
$K = 41$	*	819.94	501.40	*	1705.70	1058.20	*	2536.29	1572.20



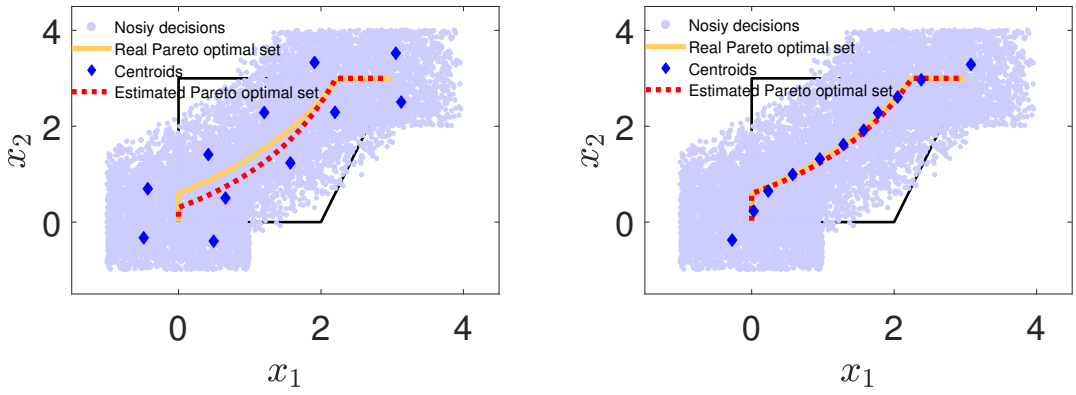
(a) (b)

Figure 14: Learning the Right-Hand Side of an MQP. We Run 500 Repetitions of the Experiments with $N = 20$, $K = 21$. (a) Norms of Primal Residuals and Dual Residuals Versus Iteration Number. (b) Norms of Estimation Error Versus Iteration Number.



(a) (b) (c) (d)

Figure 15: Learning the Objective Functions of an MQP with $N = 5 \times 10^4$ and $K = 21$. (a) Prediction Error $M(\hat{\theta}_K^N)$ for Different N and K. (b) The Dotted Yellow Line is the Error Bar Plot of the Change of the Assignments in Five Iterations over 10 Repetitions. (c) We Pick the First Repetition of the Experiments. Purple Dots Indicate the Data. The Estimated Pareto Optimal Set is Indicated by the Red Dotted Line. The Real Pareto Optimal Set is Shown by the Yellow Line. (d) Each Bar Represents the Number of Noisy Decisions that Have the Corresponding Weights for $F_1(\mathbf{x})$.



(a) Algorithm 3 (b) Algorithm 4

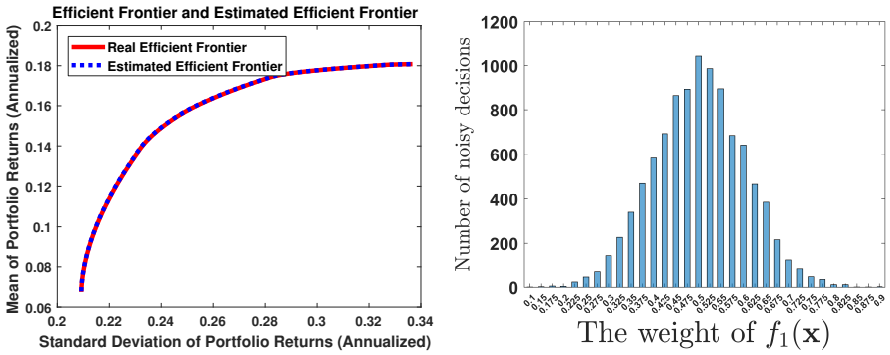
Figure 16: (a) Estimation Result for Algorithm 3. Purple Dots Indicate the Noisy Decisions. The Real Pareto Optimal Set is Shown by the Yellow Line. Blue Diamonds are the Centroids Obtained in the Initialization Step. The Estimated Pareto Optimal Set is Indicated by the Red Dotted Line. (b) Estimation Result for Algorithm 4 Using Laplacian. Blue Diamonds are the Centroids Obtained through Manifold Learning and Clustering in Step 3.

Table 6: Prediction Errors of Different Agorithms with 10000 Observations.

Algorithm 3		Algorithm 4				
		Laplacian	KernelPCA	Autoencoder	ManifoldChart	NCA
$K = 6$	0.0667	0.0436	0.0775	0.0390	0.0331	0.0323
$K = 11$	0.0262	0.0238	0.0581	0.0249	0.0244	0.0319

Table 7: Estimation Error $\|\hat{\mathbf{r}} - \mathbf{r}_{true}\|_2$ for Different N and K

	$N = 100$	$N = 1000$	$N = 2500$	$N = 5000$	$N = 7500$	$N = 10000$
$K = 11$	0.0337	0.0513	0.0406	0.0264	0.0227	0.0194
$K = 21$	0.0164	0.0154	0.0077	0.0055	0.0042	0.0043
$K = 41$	0.0220	0.0054	0.0030	0.0022	0.0018	0.0016
$K = 81$	0.0215	0.0028	0.0017	0.0008	0.0008	0.0008



(a) (b)

Figure 17: Learning the expected return of a Portfolio optimization problem with $N = 10000$ noisy portfolios and $K = 41$ weight samples. (a) The red line indicates the real efficient frontier. The blue dots indicates the estimated efficient frontier using the estimated expected return. (b) Each bar represents the number of the noisy portfolios that have the corresponding weights for $f_1(\mathbf{x})$.

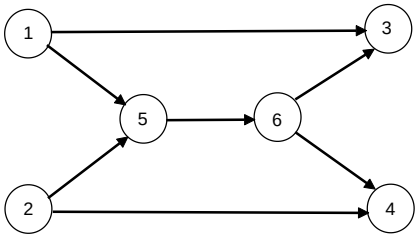


Figure 18: A Six-node Network

Table 8: Data for the Six-node Network

Link a	(1,3)	(2,4)	(1,5)	(5,6)	(2,5)	(6,3)	(6,4)
t_a^0	8.0	9.0	2.0	6.0	3.0	3.0	4.0
C_a	2000	2000	2000	4000	2000	2500	2500
h_a	8.0	9.0	2.0	6.0	3.0	3.0	4.0

Table 9: Estimation Results for Different K

K	6	11	21	41	81
O-D $(1,3)$	2056.79	2218.64	2218.64	2218.64	2288.95
O-D $(2,4)$	2185.46	3259.60	3259.60	3259.60	3576.67
Estimation error	0.3225	0.0860	0.0860	0.0860	0.0522

4.0 Wasserstein Distributionally Robust Inverse Multiobjective Optimization

4.1 Literature Review

Our work belongs to the emerging field of inverse multiobjective optimization. The goal is to infer the parameters of the multiobjective decision making problem that explains the observed decisions well. This field actually carries the data-driven concept and becomes more applicable as large amounts of data are generated and become readily available, especially those from digital devices and online transactions. There are several recent studies related to the presented research. Papers [8, 65] consider a single observation that is assumed to be an exact solution for a multiobjective linear program. Then, given a set of well-defined linear functions, an inverse optimization is formulated to learn the corresponding weight for each objective function to explain the observed decision. The most recent one is [11], which proposes the general framework to infer the objective functions or constraints from multiple noisy decisions through inverse multiobjective optimization. This work takes the framework of empirical risk minimization and generally works well when there are fewer uncertainties in the model, data or hypothetical parameter space. In contrast, we believe that those uncertainties root inherently in the field of inverse multiobjective optimization, and we aim to hedge against their influences by adopting the distributionally robust optimization paradigm based on Wasserstein metric.

Our work draws inspirations from [10], which develops a distributionally robust approach for inverse optimization to infer the utility function from sequentially arrived observations. They propose the suboptimality loss function to quantify the degree of suboptimality of an observed decision under a given candidate objective function. They show that the associated distributionally robust inverse optimization approach offers out-of samples performance guarantees. However, their approach is specifically designed for the simpler case where the decision making problem has only one objective function. Differently, our approach considers a more complex situation and is suitable when the decision making problem has multiple objectives. Moreover, instead of using the suboptimality loss function, we consider another

one that would better capture the learner's purpose to predict the decision maker's decisions. Due to the nonconvex nature of our loss function, extensive efforts are made to develop the algorithm for solving the resulting nonconvex minmax program.

4.2 Problem Setting

To help the reader better understand the core concepts in IMOP, we repeat several pieces of necessary information in Chapter 3.

4.2.1 Multiobjective Decision Making Problem

We consider a family of parametrized multiobjective decision making problems of the form

$$\begin{array}{ll} \min\limits_{\mathbf{x}\in\mathbb{R}^n} & \big\{f_1(\mathbf{x},\theta), f_2(\mathbf{x},\theta),\ldots,f_p(\mathbf{x},\theta)\big\} \\ s.t. & \mathbf{x}\in X(\theta) \end{array} \tag{4.1}$$

where $p \geq 2$ and $f_l : \mathbb{R}^n \times \mathbb{R}^{n_\theta} \mapsto \mathbb{R}$ for each $l \in [p]$. Assume parameter $\theta \in \Theta \subseteq \mathbb{R}^{n_\theta}$. We denote the vector of objective functions by $\mathbf{f}(\mathbf{x},\theta) = (f_1(\mathbf{x},\theta), f_2(\mathbf{x},\theta),\ldots,f_p(\mathbf{x},\theta))^T$. Assume $X(\theta) = \{\mathbf{x}\in\mathbb{R}^n : \mathbf{g}(\mathbf{x},\theta) \leq \mathbf{0}, \mathbf{x}\in\mathbb{R}^n_+\}$, where $\mathbf{g}(\mathbf{x},\theta) = (g_1(\mathbf{x},\theta),\ldots,g_q(\mathbf{x},\theta))^T$ is another vector-valued function with $g_k : \mathbb{R}^n \times \mathbb{R}^{n_\theta} \mapsto \mathbb{R}$ for each $k \in [q]$.

Definition 4.2.1 (Pareto optimality)**.** For fixed θ, a decision vector $\mathbf{x}^* \in X(\theta)$ is said to be Pareto optimal if there exists no other decision vector $\mathbf{x} \in X$ such that $f_i(\mathbf{x},\theta) \leq f_i(\mathbf{x}^*,\theta)$ for all $i \in [p]$, and $f_k(\mathbf{x},\theta) < f_k(\mathbf{x}^*,\theta)$ for at least one $k \in [p]$.

In the study of multiobjective optimization, the set of all Pareto optimal solutions is denoted by $X_P(\theta)$ and called the Pareto optimal set. The weighting method is commonly used to obtain a Pareto optimal solution through computing the problem of weighted sum (PWS) [66] as follows.

$$\begin{array}{ll} \min & w^T\mathbf{f}(\mathbf{x},\theta) \\ s.t. & \mathbf{x}\in X(\theta) \end{array} \tag{4.2}$$

where $w = (w^1, \ldots, w^p)^T$. Without loss of generality, all possible weights are restricted to a simplex, which is denoted by $\mathscr{W}_p = \{w \in \mathbb{R}^p_+ : \ \mathbf{1}^T w = 1\}$. Next, we denote the set of optimal solutions for the 4.2 by

$$S(w, \theta) = \arg\min_{\mathbf{x}} \left\{ w^T \mathbf{f}(\mathbf{x}, \theta) : \mathbf{x} \in X(\theta) \right\}.$$

In the following, we make a few assumptions to simplify our understanding, which are actually mild and appear frequently in the literature.

Assumption 4.2.1. Set Θ is a convex compact set in $\mathbb{R}^{n_\theta}$. There exists $D > 0$ such that $\sup_{\theta \in \Theta} \|\theta\|_2 \leq D$. In addition, $\mathbf{f}(\mathbf{x}, \theta)$ and $\mathbf{g}(\mathbf{x}, \theta)$ are convex in $\mathbf{x}$ for each $\theta \in \Theta$.

4.2.2 Inverse Multiobjective Optimization

We begin with a discussion on the construction of an appropriate loss function for the inverse multiobjective optimization problem as discussed in Chapter 3. We consider the following loss function and surrogate loss function.

Given a noisy decision $\mathbf{y}$ and a hypothesis θ, the loss function is defined as the minimum (squared) distance between $\mathbf{y}$ and the efficient set $X_P(\theta)$:

$$l(\mathbf{y}, \theta) = \min_{\mathbf{x} \in X_P(\theta)} \|\mathbf{y} - \mathbf{x}\|_2^2. \tag{4.3}$$

For a general 4.1, however, there might exist no explicit way to characterize the efficient set $X_P(\theta)$. Hence, an approximation approach to practically describe this set can be adopted. Then, a sampling approach can be adopted to generate $w_k \in \mathscr{W}_p$ for each $k \in [K]$ and approximate $X_P(\theta)$ as $\bigcup_{k \in [K]} S(w_k, \theta)$. Then, the *surrogate loss function* is defined as

$$l_K(\mathbf{y}, \theta) = \min_{\mathbf{x} \in \bigcup\limits_{k \in [K]} S(w_k, \theta)} \|\mathbf{y} - \mathbf{x}\|_2^2. \tag{4.4}$$

By using binary variables, this surrogate loss function can be converted into the *surrogate loss problem.*

$$\begin{aligned} l_K(\mathbf{y}, \theta) \ = \ & \min_{z_j \in \{0,1\}} \ \|\mathbf{y} - \sum_{k \in [K]} z_k \mathbf{x}_k\|_2^2 \\ & \text{s.t.} \ \ \sum_{k \in [K]} z_k = 1, \ \mathbf{x}_k \in S(w_k, \theta) \end{aligned} \tag{4.5}$$

We make the following assumptions as that in [11], which are common in inverse optimization.

Assumption 4.2.2. **(a)** $X(\theta)$ is closed, and has a nonempty relative interior. $X(\theta)$ is also bounded. Namely, there exists $B > 0$ such that $\|\mathbf{x}\|_2 \leq B$ for all $\mathbf{x} \in X(\theta)$. The support $\mathcal{Y}$ of the noisy decisions $\mathbf{y}$ is contained within a ball of radius R almost surely, where $R < \infty$. In other words, $\mathbb{P}(\|\mathbf{y}\|_2 \leq R) = 1$.

(b) Each function in $\mathbf{f}$ is strongly convex on $\mathbb{R}^n$, that is for each $l \in [p]$, $\exists \lambda_l > 0$, $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n$

$$\Big(\nabla f_l(\mathbf{y}, \theta_l) - \nabla f_l(\mathbf{x}, \theta_l)\Big)^T (\mathbf{y} - \mathbf{x}) \geq \lambda_l \|\mathbf{x} - \mathbf{y}\|_2^2.$$

Given observations $\{\mathbf{y}_i\}_{i\in[N]}$ drawn i.i.d. according to the distribution $\mathbb{P}_{\mathbf{y}}$, the inverse multiobjective optimization program is given in the following.

$$\min_{\theta\in\Theta} \quad \tfrac{1}{N} \sum_{i\in[N]} l_K(\mathbf{y}_i, \theta). \tag{4.6}$$

4.2.3 Wasserstein Ambiguity Set

Let $\mathcal{Y} \subseteq \mathbb{R}^n$ be the observation space where the observed noisy decisions take values. Denote $\mathscr{P}(\mathcal{Y})$ be the set of all probability distributions on $\mathcal{Y}$. From now on, we let the Wasserstein ambiguity set $\mathcal{P}$ be the 1-Wasserstein ball of radius ϵ centered at P_0:

$$\mathcal{P} = \mathbb{B}_\epsilon(P_0) := \{Q \in \mathscr{P}(\mathcal{Y}) : \mathcal{W}(Q, P_0) \leq \epsilon\}, \tag{4.7}$$

where P_0 is the nominal distribution on $\mathcal{Y}$, $\epsilon > 0$ is the radius of the set, and $\mathcal{W}(Q, P_0)$ is the wasserstein distance metric of order 1 defined as [31, 30, 28]

$$\mathcal{W}(Q, P_0) = \inf_{\pi\in\Pi(Q,P_0)} \int_{\mathcal{Y}\times\mathcal{Y}} \|z_1 - z_2\|_2 \pi(dz_1, dz_2),$$

where $\Pi(Q, P_0)$ is the set of probability distributions on $\mathcal{Y} \times \mathcal{Y}$ with marginals Q and P_0.

4.3 Wasserstein Distributionally Robust IMOP

In this section, we propose the Wasserstein distributionally robust IMOP, and show its equivalence to a semi-infinite program. Subsequently, we present an algorithm to handle the resulting reformulations, and show its convergence in finite steps. Finally, we establish the statistical performance guarantees for the distributionally robust IMOP.

Given observations $\{\mathbf{y}_i\}_{i\in[N]}$ drawn i.i.d. according to the distribution $\mathbb{P}_{\mathbf{y}}$, the corresponding distributionally robust program of (4.6) equipped with the Wasserstein ambiguity set is constructed as follows

$$\min_{\theta\in\Theta} \sup_{Q\in\mathbb{B}_\epsilon(\widehat{P}_N)} \mathbb{E}_{\mathbf{y}\sim Q}\left[l_K(\mathbf{y},\theta)\right], \tag{4.8}$$

which minimizes the worst case expected loss over all the distributions in the Wasserstein ambiguity set. Here $\mathbb{B}_\epsilon(\widehat{P}_N)$ is defined in (4.7), and $\widehat{P}_N$ is the empirical distribution satisfying: $\widehat{P}_N(\mathbf{y}_i) = 1/N, \forall i \in [N]$.

4.3.1 Semi-infinite Reformulations

Problem (4.8) involves minimizing a supremum over infinitely many distributions, which makes it difficult to solve. In this section, we establish the reformulation of (4.8) into a semi-infinite program.

The performance of (4.8) depends on how the change of θ affects the objective values. For $\forall w \in \mathscr{W}_p, \theta_1 \in \Theta, \theta_2 \in \Theta$, we consider the following function

$$h(\mathbf{x}, w, \theta_1, \theta_2) = w^T\mathbf{f}(\mathbf{x},\theta_1) - w^T\mathbf{f}(\mathbf{x},\theta_2).$$

Assumption 4.3.1. $\exists\kappa > 0$, $\forall w \in \mathscr{W}_p$, $\forall\theta_1 \neq \theta_2 \in \Theta$, $h(\cdot, w, \theta_1, \theta_2)$ is Lipschitz continuous on $\mathcal{Y}$:$\forall\mathbf{x},\mathbf{y}\in\mathcal{Y}$,

$$|h(\mathbf{x}, w, \theta_1, \theta_2) - h(\mathbf{y}, w, \theta_1, \theta_2)| \leq \kappa\|\theta_1 - \theta_2\|_2\|\mathbf{x} - \mathbf{y}\|_2.$$

Basically, this assumption requires that the objective functions will not change much when either the parameter θ or the variable $\mathbf{x}$ is perturbed. It actually holds in many common situations, including the multiobjective linear program (MLP) and multiobjective quadratic program (MQP). As a motivating example, we give the κ for an MQP.

Example 4.3.1. Suppose that $\mathbf{f}(\mathbf{x}, \theta) = \begin{pmatrix} \frac{1}{2}\mathbf{x}^T Q_1 \mathbf{x} + \mathbf{c}_1^T \mathbf{x} \\ \frac{1}{2}\mathbf{x}^T Q_2 \mathbf{x} + \mathbf{c}_2^T \mathbf{x} \end{pmatrix}$, where $\theta = (Q_1, Q_2, \mathbf{c}_1, \mathbf{c}_2)$. Under Assumption 4.2.2, we know that $\|\mathbf{y}\|_2 \leq R$. Then, $h(\cdot, w, \theta_1, \theta_2)$ is $2R\|\theta_1 - \theta_2\|_2$-Lipschitz continuous on $\mathcal{Y}$. That is, we can set $\kappa = 2R$.

Under the previous assumptions, we will establish several properties of the loss function $l_K(\mathbf{y}, \theta)$, which are ensential for our reformulation for (4.8).

Lemma 4.3.1. Under Assumptions 4.2.1 - 4.3.1, the loss function $l_K(\mathbf{y}, \theta)$ has the following properties:

(a) $\forall \mathbf{y} \in \mathcal{Y}, \theta \in \Theta, 0 \leq l_K(\mathbf{y}, \theta) \leq (B + R)^2$.

(b) $l_K(\mathbf{y}, \theta)$ is uniformly $2(B + R)$-Lipschitz continuous in $\mathbf{y}$. That is, $\forall \theta \in \Theta, \forall \mathbf{y}_1, \mathbf{y}_2 \in \mathcal{Y}$, we have

$$|l_K(\mathbf{y}_1, \theta) - l_K(\mathbf{y}_2, \theta)| \leq 2(B + R)\|\mathbf{y}_1 - \mathbf{y}_2\|_2.$$

(c) $l_K(\mathbf{y}, \theta)$ is uniformly $\frac{4(B+R)\kappa}{\lambda}$-Lipschitz continuous in θ. That is, $\forall \mathbf{y} \in \mathcal{Y}, \forall \theta_1, \theta_2 \in \Theta$, we have

$$|l_K(\mathbf{y}, \theta_1) - l_K(\mathbf{y}, \theta_2)| \leq \frac{4(B + R)\kappa}{\lambda}\|\theta_1 - \theta_2\|_2.$$

(a) and (b) of this lemma are built upon direct analyses of the loss function $l_K(\mathbf{y}, \theta)$. Proof of (c) is much more involved and needs the key observation that the perturbation of $S(w, \theta)$ due to θ is bounded by the perturbation of θ by applying Proposition 6.1 in [57]. Proof details are given in the supplementary material.

Let

$$\mathcal{V} := \Big\{ \mathbf{v} \in \mathbb{R}^{N+1} : V_1 \leq v_i \leq (m+1)V_2 - mV_1, \forall i \in [N], \\ 0 \leq v_{N+1} \leq (V_2 - V_1)/\epsilon \Big\}.$$

where V_1 and V_2 are the lower and upper bounds for the loss function $l_K(\mathbf{y}, \theta)$, respectively. By part (a) of Lemma 4.3.1, we will set $V_1 = 0$, and $V_2 = (B+R)^2$ throughout the remainder of the paper.

We are now ready to state the main result in this paper.

Theorem 4.3.2 (Semi-infinite Reformulation)**.** Under Assumptions 4.2.1 - 4.3.1, (4.8) is equivalent to the following semi-infinite program:

$$\begin{aligned} \min_{\theta, \mathbf{v}} \quad & \epsilon \cdot v_{N+1} + \tfrac{1}{N} \sum_{i \in [N]} v_i \\ s.t. \quad & \sup_{\widetilde{\mathbf{y}} \in \mathcal{Y}} \left(l_K(\widetilde{\mathbf{y}}, \theta) - v_{N+1} \cdot \|\widetilde{\mathbf{y}} - \mathbf{y}_i\|_2 \right) \leq v_i, \quad \forall i \in [N], \\ & \theta \in \Theta, \mathbf{v} \in \mathcal{V} \end{aligned} \tag{4.9}$$

The establishment of Theorem 4.3.2 relies on those properties of the loss function $l_K(\mathbf{y}, \theta)$ stated in Lemma 4.3.1. Although $l_K(\mathbf{y}, \theta)$ might not be convex in θ or $\mathbf{y}$, these properties ensure that strong (Kantorovich) duality holds for the inner problem of (4.8). Details of the proof are given in the supplementary material.

Next, we will discuss how to incorporate the explicit form of $l_K(\mathbf{y}, \theta)$ into the constraints of (4.9). For each $i \in [N]$, constraints in (4.9) is equivalent to: $\forall \widetilde{\mathbf{y}} \in \mathcal{Y}$,

$$\begin{aligned} & \|\widetilde{\mathbf{y}} - \mathbf{x}_k\|_2^2 - v_{N+1} \cdot \|\widetilde{\mathbf{y}} - \mathbf{y}_i\|_2 - v_i \leq M z_{ik}, \\ & \sum_{k \in [K]} z_{ik} = K - 1, \end{aligned} \tag{4.10}$$

where the additional constraint $\sum_{k \in [K]} z_{ik} = K-1$, is imposed to ensure that $\|\widetilde{\mathbf{y}} - \mathbf{x}_k\|_2^2 - v_i - v_{N+1} \cdot \|\widetilde{\mathbf{y}} - \mathbf{y}_i\|_2 \leq 0$ for at least one $k \in [K]$. M is an uniform upper bound for the left-hand side of the first constraint in (4.10). An appropriate M could be $(B+R)^2$, since $\forall i \in [N], k \in [K]$,

$$\begin{aligned} & \|\widetilde{\mathbf{y}} - \mathbf{x}_k\|_2^2 - v_{N+1} \cdot \|\widetilde{\mathbf{y}} - \mathbf{y}_i\|_2 - v_i \leq \|\widetilde{\mathbf{y}} - \mathbf{x}_k\|_2^2 \\ & \leq (B+R)^2. \end{aligned}$$

One can verify that (4.10) is indeed equivalent to the first set of constraints in (4.9) without much effort.

Remark 4.3.1. We admit that the semi-infinite reformulation in Theorem 4.3.2 might still be valid if some assumption is not satisfied. Consider, for example, one of the objective functions is known to be strongly convex and the decision makers always has a positive preference for it.

4.3.2 Algorithm and Analysis of Convergence

Theorem 4.3.2 shows that the Wasserstein distributionally inverse multiobjective program (4.8) is equivalent to the semi-infinite program (4.9). Now, any existing method for solving the general semi-infinite program can be employed to solve (4.9). In particular, we are interested in using exchange methods [97], since our proposed algorithm inherits the spirit of these methods when applied to solving the minmax problem. The basic idea is to approximate the infinite set of constraints in (4.9) with a sequence of finite sets of constraints. Iteratively, new constraints are added to the previous set of constraints by solving a maximum constraint violation problem. This is repeated until certain stopping criterion is satisfied.

Next, we will discuss how to construct the finite problem (i.e., the master problem).

Let $\widetilde{\mathcal{Y}}_i = \{\widetilde{\mathbf{y}}_{i1}, \cdots, \widetilde{\mathbf{y}}_{iJ_i}\} \subseteq \mathcal{Y}, \forall i \in [N]$ be a collection of finite subsets of $\mathcal{Y}$, where each subset has J_i samples. Then, the associated finite problem of (4.9) is

$$\begin{aligned} &\min_{\theta,\mathbf{v}} \ \epsilon \cdot v_{N+1} + \tfrac{1}{N} \sum_{i\in[N]} v_i, \\ &s.t.\ l_K(\widetilde{\mathbf{y}}_{ij}, \theta) - v_{N+1} \cdot \|\widetilde{\mathbf{y}}_{ij} - \mathbf{y}_i\|_2 \leq v_i, \forall j \in [J_i], i \in [N], \\ &\qquad \theta \in \Theta, v \in \mathcal{V}. \end{aligned} \tag{4.11}$$

By the same arguments for the transformation from constraints in (4.9) to (4.10), constraints in (4.11) are equivalent to

$$\begin{aligned} &\|\widetilde{\mathbf{y}}_{ij} - \mathbf{x}_k\|_2^2 - v_{N+1} \cdot \|\widetilde{\mathbf{y}}_{ij} - \mathbf{y}_i\|_2 - v_i \leq M z_{ijk}, \\ &\sum_{k\in[K]} z_{ijk} = K - 1, \ \ \forall i \in [N], j \in [J_i]. \end{aligned}$$

Using the above transformation, (4.11) can be further cast into the following finite problem with finitely many constraints:

$$\begin{aligned}
&\min_{\theta,\mathbf{v},\mathbf{x}_k,z_{ijk}} \quad \epsilon \cdot v_{N+1} + \tfrac{1}{N}\sum_{i\in[N]} v_i, \\
&s.t. \quad \|\widetilde{\mathbf{y}}_{ij} - \mathbf{x}_k\|_2^2 - v_{N+1}\cdot\|\widetilde{\mathbf{y}}_{ij} - \mathbf{y}_i\|_2 - v_i \le M z_{ijk}, \\
&\qquad \mathbf{x}_k \in S(w_k,\theta), \\
&\qquad \sum_{k\in[K]} z_{ijk} = K-1, \\
&\qquad \theta\in\Theta, v\in\mathcal{V}, z_{ijk}\in\{0,1\}, \forall i\in[N], j\in[J_i], k\in[K].
\end{aligned} \tag{4.12}$$

At each iteration, new constraints are determined to add to the previous set of constraints in (4.12) by solving the following **Maximum constraint violation problem** (i.e., the subproblem): $\forall i \in [N]$,

$$CV_i = \max_{\widetilde{\mathbf{y}}\in\mathcal{Y}} l_K(\widetilde{\mathbf{y}},\widehat{\theta}) - \widehat{v}_{N+1}\cdot\|\widetilde{\mathbf{y}} - \mathbf{y}_i\|_2 - \widehat{v}_i. \tag{4.13}$$

Denote $\widetilde{\mathbf{y}}_i$ the optimal solution of (4.13) for each $i \in [N]$. Whenever we find that $CV_i > 0$, we append $\widetilde{\mathbf{y}}_i$ to $\widetilde{\mathcal{Y}}_i$. As a result, we tighten our approximation for the infinite set of constraints in (4.9) by imposing the additional constraint $l_K(\widetilde{\mathbf{y}}_i,\widehat{\theta}) - \widehat{v}_{N+1}\cdot\|\widetilde{\mathbf{y}}_i - \mathbf{y}_i\|_2 - \widehat{v}_i \le 0$ in the next iteration.

With the above assumptions and analyses, we now present our method to solve (4.8) in Algorithm 5. We also illustrate the general scheme of Algorithm 5 in Figure 19.

Remark 4.3.2. In Step 6, the maximum constraint violation problem can be solved exactly and efficiently by invoking solver such as Baron [98]. Nevertheless, it can also be solved approximately by decomposing into K subproblems, each of which is a possibly nonconvex program when $\widehat{v}_{N+1} < 1$. Nevertheless, this nonconvex problem is a quadratically constrained quadratic program (QCQP) with a single constraint. Thus, it can be solved exactly and efficiently through the S-procedure [99, 100]. Additionally, K different subproblems can be solved independently and in parallel, allowing a linear speedup of Step 6.

For completeness, we provide the convergence proof of Algorithm 5 in the following theorem.

Algorithm 5 Wasserstein Distributionally Robust IMOP

Input: noisy decisions $\{\mathbf{y}_i\}_{i\in[N]}$, weights $\{w_k\}_{k\in K}$, radius ϵ of Wasserstein ball, and stopping tolerance δ

Initialize $\widetilde{\mathcal{Y}}_i \leftarrow \emptyset, \forall i \in [N]$

repeat

  solve the master problem in (4.12) with $\widetilde{\mathcal{Y}}_i, \forall i \in [N]$, and return an optimal solution $(\widehat{\theta}, \widehat{\mathbf{v}})$

  for $i = 1, \ldots, N$ **do**

    solve the subproblem, i.e., the maximum constraint violation problem in (4.13)

    if $CV_i > 0$ **then** let $\widetilde{\mathcal{Y}}_i \leftarrow \widetilde{\mathcal{Y}}_i \cup \{\widetilde{\mathbf{y}}_i\}$ **end if**

  end for

until $\max_{i\in[N]} CV_i \leq \delta$

Output: a δ-optimal solution $\widehat{\theta}_N$ of (4.9)

Theorem 4.3.3. Under Assumptions 4.2.1 - 4.3.1, Algorithm 5 converges within $(\frac{GR_0}{\delta} + 1)^{n_\theta+N+1}$ iterations. Here,

$$G = (1 + 2R + \frac{4(B+R)\kappa}{\lambda}),$$
$$R_0 = \sqrt{D^2 + N\big((m+1)V_2 - mV_1\big)^2 + \left(\frac{V_2 - V_1}{\epsilon}\right)^2}.$$

Remark 4.3.3. The proof of convergence is in spirit similar to that of the cutting plane methods for robust optimization and distributionally robust optimization [101, 29]. In practice, we mention that the actual number of iterations typically required is much smaller than $(\frac{GR_0}{\delta} + 1)^{n_\theta+N+1}$.

Variants of Algorithm 5 Note that we add $\widetilde{\mathbf{y}}_i$ to $\widetilde{\mathcal{Y}}_i$ whenever $V_i > 0$ for each $i \in [N]$. Nevertheless, from the convergence proof, it suffices to add only one $\widetilde{\mathbf{y}}_i$ corresponding to the biggest V_i that are positive. Consequently, we dramatically ease the computational burden in each iteration.

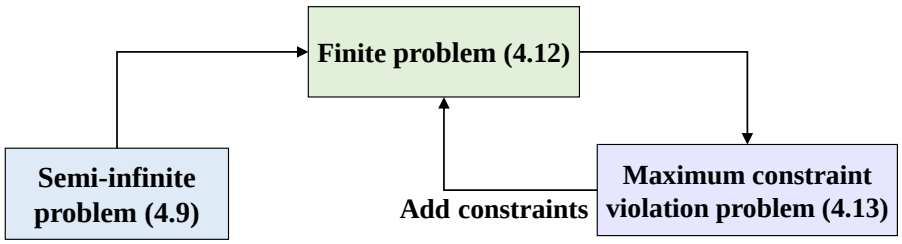


Figure 19: General scheme of Algorithm 5.

4.3.3 Performance Guarantees

One of the main goals of statistical analysis of learning algorithms is to understand how the excess risk of a data dependent decision rule output by the empirical risk minimization depends on the sample size of the observations and on the "complexity" of the class Θ. Next, we provide a performance guarantee for (4.8) by showing below that the excess risk of the estimator obtained by solving (4.8) would converge sub-linearly to zero.

Theorem 4.3.4 (Excess risk bound)**.** Define the minimax risk estimator

$$\theta^* \in \arg\min_{\theta\in\Theta} \left\{ \sup_{Q\in\mathbb{B}_\epsilon(P)} \mathbb{E}_{\mathbf{y}\sim Q} \left[l_K(\mathbf{y},\theta) \right] \right\},$$

where P is the distribution from which the observations $\{\mathbf{y}_i\}_{i\in[N]}$ are drawn, and the minimax empirical risk estimator

$$\widehat{\theta}_N \in \arg\min_{\theta\in\Theta} \left\{ \sup_{Q\in\mathbb{B}_\epsilon(\widehat{P}_N)} \mathbb{E}_{\mathbf{y}\sim Q} \left[l_K(\mathbf{y},\theta) \right] \right\}.$$

and $\widehat{P}_N$ is the empirical distribution of the observations $\{\mathbf{y}_i\}_{i\in[N]}$.

Under Assumptions 4.2.1 - 4.3.1, $\forall 0 < \delta < 1$, the following holds with probability at least $1-\delta$:

$$\sup_{Q\in\mathbb{B}_\epsilon(P)} \mathbb{E}_{\mathbf{y}\sim Q} \left[l_K(\mathbf{y},\widehat{\theta}_N) \right] - \sup_{Q\in\mathbb{B}_\epsilon(P)} \mathbb{E}_{\mathbf{y}\sim Q} \left[l_K(\mathbf{y},\theta^*) \right] \leq \tfrac{H}{\sqrt{N}} + \tfrac{3(B+R)^2\sqrt{\log(2/\delta)}}{\sqrt{2N}}$$

where H is a constant depending only on $D, B, R, n_\theta, \kappa$:

$$H = 96 \left(\frac{3D\sqrt{n_\theta}}{\kappa} + 2R \right) (B+R).$$

Remark 4.3.4. Analogous to the convergence rate of empirical risk minimization when $\epsilon = 0$, we get an $\mathcal{O}(1/\sqrt{N})$ excess risk bound. However, the obtained excess risk bound does not depend on the radius ϵ of the Wasserstein ambiguity set. Similar to [35], this phenomenon is due to the fact that we are using the Lipschitz continuity of the loss function $l_K(\mathbf{y}, \theta)$. Moreover, the right terms in the excess risk bound inequality increase as either D, B, R, n_θ grow or κ shrinks, indicating that the learnability of the decision making model decreases. This is consistent with our observation that uncertainties in the model, data, and parameter space will enhance the difficulty for learning the parameters trhough inverse multiobjective optimization in general.

4.4 Experiments

In this section, we will provide a multiobjective quadratic program (MQP) and a portfolio optimization problem to illustrate the performance of the proposed algorithm 5. The mixed integer second order conic problems are solved by Gurobi [87]. All the algorithms are programmed with Julia [62].

4.4.1 Learning the Objective Functions of an MQP

Consider the following multiobjective quadratic optimization problem.

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^2_+} \quad & \begin{pmatrix} f_1(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T Q_1 \mathbf{x} + \mathbf{c}_1^T \mathbf{x} \\ f_2(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T Q_2 \mathbf{x} + \mathbf{c}_2^T \mathbf{x} \end{pmatrix} \\ s.t. \quad & A\mathbf{x} \leq \mathbf{b} \end{aligned} \tag{4.14}$$

where the parameters of the two objective functions are

$$Q_1 = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}, \mathbf{c}_1 = \begin{bmatrix} -0.5 \\ -1 \end{bmatrix}, Q_2 = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}, \mathbf{c}_2 = \begin{bmatrix} -5 \\ -2.5 \end{bmatrix},$$

and the parameters for the feasible region are

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}.$$

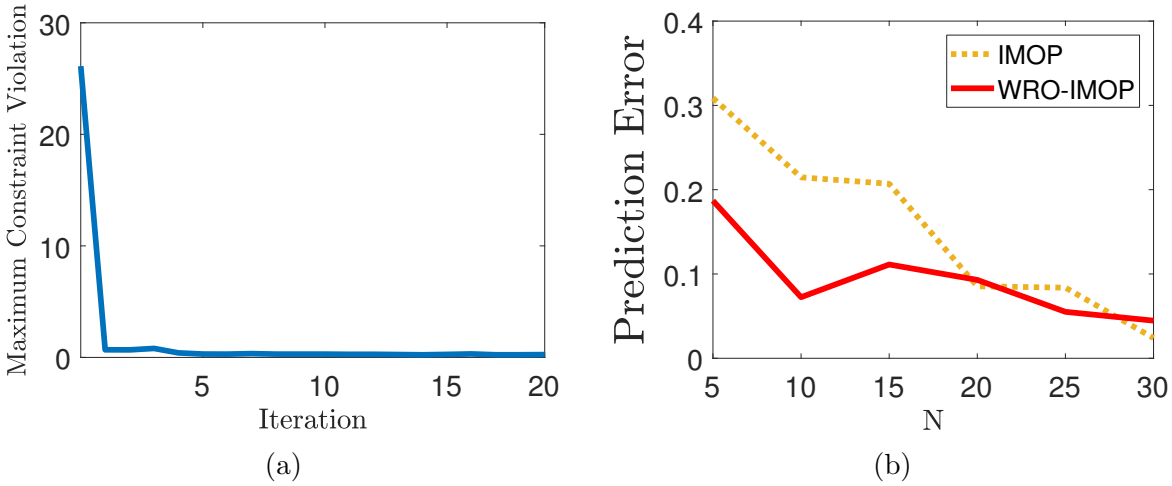


Figure 20: Learning the objective functions of a Multiobjective quadratic program. (a) Maximum constraint violation versus iteration for $N = 15$. (b) Prediction errors for two methods with different N. Results are averaged over 10 repetitions.

We seek to learn $\mathbf{c}_1$ and $\mathbf{c}_2$ in this experiment. The data is generated as follows. We first compute Pareto optimal solutions $\{\mathbf{x}_i\}_{i\in[N]}$ by solving 4.2 with weight samples $\{w_i\}_{i\in[N]}$ that are uniformly chosen from $\mathscr{W}_2$. Next, the noisy decision $\mathbf{y}_i$ is obtained by adding noise to $\mathbf{x}_i$ for each $i \in [N]$. More precisely, $\mathbf{y}_i = \mathbf{x}_i + \epsilon_i$, where each element of ϵ_i has a uniform distribution supporting on $[-0.25, 0.25]$ with mean 0 for all $i \in [N]$.

We assume that $\mathbf{c}_1$ and $\mathbf{c}_2$ are within $[-6, 0]^2$, and the first elements for them are given. $K = 6$ weights from $\mathscr{W}_2$ are evenly sampled. The radius ϵ of the Wasserstein ambiguity set is selected from the set $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1\}$. We report below the results with lowest

prediction error across all candidate radii. The stopping criteria δ is set to be 0.1. Then, we implement Algorithm 5 with different N.

To illustrate the performance of the algorithm in a statistical way, we run 10 repetitions of the experiments. Figure 20a shows the maximum constraint violation $\max_{i\in[N]} V_i$ versus iteration for one repetition when $N = 10$. As can be seen in the figure, the algorithm converges very fast. In Figure 20b, we report the prediction errors averaged over 10 repetitions with both the robust and non-robust approaches for different N. Here, we use an independent validation set that consists of 10^5 noisy decisions generated in the same way as the training data to compute the prediction error. The experiments suggest that the Wasserstein distributionally robust approach can significnatly reduce the prediction error, especially when N is small, i.e., we have very limited number of observations.

4.4.2 Learning the Expected Returns

In this example, we consider various noisy decisions arising from different investors in a stock market. More precisely, we consider a portfolio selection problem, where investors need to determine the fraction of their wealth to invest in each security in order to maximize the total return and minimize the total risk. The classical Markovitz mean-variance portfolio selection [17] in the following is frequently employed by analysts.

$$\begin{aligned} \min \quad & \begin{pmatrix} f_1(\mathbf{x}) & = -\mathbf{r}^T\mathbf{x} \\ f_2(\mathbf{x}) & = \mathbf{x}^T Q\mathbf{x} \end{pmatrix} \\ s.t. \quad & 0 \le x_i \le b_i \qquad\qquad \forall i \in [n] \\ & \sum_{i=1}^{n} x_i = 1 \end{aligned} \tag{4.15}$$

where $\mathbf{r} \in \mathbb{R}^n_+$ is a vector of individual security expected returns, $Q \in \mathbb{R}^{n\times n}$ is the covariance matrix of securities returns, $\mathbf{x}$ is a portfolio specifying the proportions of capital to be invested in the different securities, and b_i is an upper bound on the proportion of security i, $\forall i \in [n]$.

We use the portfolio data *BlueChipStockMoments* derived from real data in the Matlab Financial Toolbox. The true expected returns and true return covariance matrix for the first 8 securities are given in the supplementary material. Suppose a learner seeks to learn the

expected return for the first four securities that an analyst uses based on 20 noisy decisions from investors that the analyst serves.

The noisy decision for each investor $i \in [20]$ is generated as follows. We set each upper bound for the proportion of the 8 securities to $b_i = 1.0, \forall i \in [8]$. Then, we uniformly sample 20 weights and use them to generate optimal portfolios on the efficient frontier that is plot in Figure 22. Subsequently, each component of these portfolios is rounded to the nearest thousandth, which can be seen as measurement error. The radius ϵ of the Wasserstein ambiguity set is selected from the set $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1\}$. The stopping criteria δ is set to be 0.1.

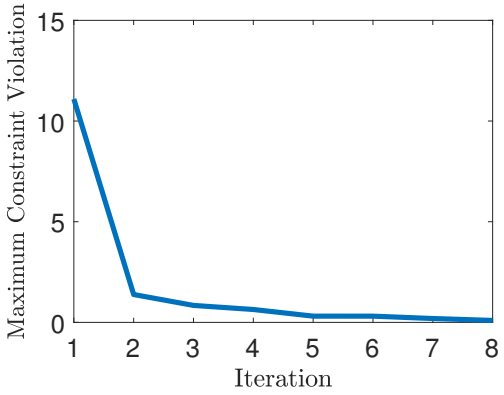


Figure 21: Maximum constraint violation versus iteration.

Figure 21 shows that our algorithm converges in 8 iterations. We also plot the estimated efficient frontiers using both the robust and non-robust approaches with $K = 6$ in Figure 22. We can see that the estimated efficient frontier of the Wasserstein distributionally robust approach is closer to the real one than the non robust approach, showing that our method in this paper allows for a lower prediction error when only limited number of decisions observed are accessible. Note that the first function is not strongly convex. The experiment results suggest that our reformulation is generalizable to a broader class of problems.

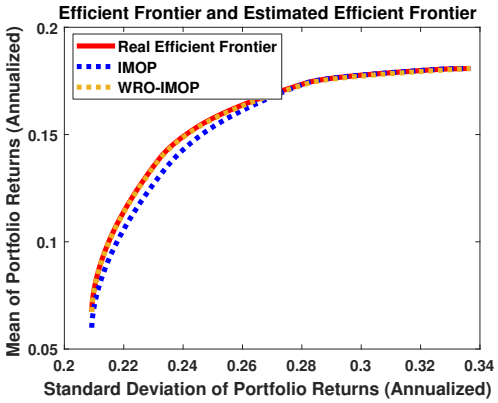


Figure 22: The red line indicates the real efficient frontier. The yellow dots indicates the estimated efficient frontier using the distributionally robust approach. The blue dots indicates the estimated efficient frontier using the non-robust approach.

5.0 Conclusion and Future Work

In this dissertation, we develop a set of methodologies in the field of inverse optimization based learning. First, we propose an online learning method to infer preferences or restrictions from noisy observations is developed and implemented. We prove a regret bound for the implicit online learning algorithm under certain regularity conditions, and show the algorithm is statistically consistent, which guarantees that our algorithm will asymptotically achieves the best prediction error permitted by the inverse model. Finally, we illustrate the performance of our learning method on both a consumer behavior problem and a transshipment problem. Results show that our algorithm can learn the parameters with great accuracy and is very robust to noises, and achieves drastic improvement in computational efficacy over the batch learning approach.

We also study the problem of learning the objective functions and constraints of a multiobjective decision making problem, based on observations of efficient solutions which might carry noise. Specifically, we formulate such a learning task as an inverse multiobjective optimization problem, and provide a deep analysis to establish the statistical significance of the inference results from the presented model. Moreover, we discuss the strong correlation between the identifiability of the decision making problem and the performance of our inverse optimization model. We then develop two numerical algorithms to handle the computational challenge from the large number of observations. We confirm by extensive numerical experiments that the proposed algorithms can learn the parameters with great accuracy while dramatically improve the computational efficacy.

In addition, we propose a distributionally robust approach to inverse multiobjective optimization. Specifically, we study the problem of learning the objective functions or constraints of a multiobjective decision making model, based on a set of observed decisions. In particular, these decisions might not be exact and possibly carry measurement noises or are generated with the bounded rationality of decision makers. We use the Wasserstein metric to construct the uncertainty set centered at the empirical distribution of these decisions. We show that this framework has several nice statistical performance guarantees. We also develop an

efficient algorithm to solve the resulting minmax problem and prove its finite convergence. Numerical experiments demonstrate the effectiveness of the proposed algorithm.

Future work for the inverse optimization and the inverse multiobjective optimization will mainly focus on the application of the proposed methods, exploring new models and algorithms, and developing strong theories. For example, we seek to apply the online learning methods to real world machine learning problems, such as those occurred in designing recommender systems. We will also investigate the robustness of the proposed inverse optimization model, and apply it to infer the preferences of the decision makers such as those investors in portfolio optimization. Furthermore, we will explore incorporating deep learning into inverse optimization and inverse multiobjective optimization to scale up their capabilities in the modern machine learning framework.

Appendix A

A.1 Omitted Mathematical Reformulations

A.1.1 Single Level Reformulation for the Inverse LP

When the objective function is linear, namely, the optimization problem has the following form

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n_+} \quad & \mathbf{c}^T\mathbf{x} \\ s.t. \quad & A\mathbf{x} \geq \mathbf{b} \end{aligned} \tag{A.1}$$

Suppose that the right hand side $\mathbf{b}$ changes over time t. That is, $\mathbf{b} = \mathbf{b}_t$ at time t. When trying to learn $\mathbf{c}$, the single level reformulation the inverse problem is

$$\begin{aligned} \min_{\mathbf{c}\in\Theta} \quad & \tfrac{1}{2}\|\mathbf{c}-\mathbf{c}_t\|_2^2 + \eta_t\|\mathbf{y}_t - \mathbf{x}\|_2^2 \\ s.t. \quad & \mathbf{A}\mathbf{x} \geq \mathbf{b}_t,\ \mathbf{x} \geq \mathbf{0} \\ & \mathbf{A}^T\mathbf{u} \leq \mathbf{c}, \\ & \mathbf{x} \leq M_1\mathbf{z}_1 \\ & \mathbf{c} - \mathbf{A}^T\mathbf{u} \leq M_1(1-\mathbf{z}_1) \\ & \mathbf{u} \leq M_2\mathbf{z}_2 \\ & \mathbf{A}\mathbf{x} - \mathbf{b}_t \leq M_2(1-\mathbf{z}_2) \\ & \mathbf{x}\in\mathbb{R}^n_+,\ \ \mathbf{u}\in\mathbb{R}^m_+,\ \ \mathbf{z}_1\in\{0,1\}^n,\ \ \mathbf{z}_2\in\{0,1\}^m \end{aligned} \tag{A.2}$$

where M_1 and M_2 are appropriate numbers used to bound $\mathbf{x}$ and $\mathbf{c}-\mathbf{A}^T\mathbf{u}$, $\mathbf{u}$ and $\mathbf{A}\mathbf{x}-\mathbf{b}_t$ respectively.

We have a similar single level reformulation when learning the Right-hand side $\mathbf{b}$. Clearly, this is a Mixed Integer Second Order Cone program(MISOCP) when learning either $\mathbf{c}$ or $\mathbf{b}$.

A.1.2 Single Level Reformulation for the Inverse QP

When the objective functions are quadratic, namely, the optimization problem has the following form

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & \tfrac{1}{2}\mathbf{x}^T Q\mathbf{x} + \mathbf{c}^T\mathbf{x} \\ s.t. \quad & A\mathbf{x} \geq \mathbf{b} \end{aligned} \tag{A.3}$$

Suppose that $\mathbf{c}$ changes over time t. That is, $\mathbf{c} = \mathbf{c}_t$ at time t. When trying to learn $\mathbf{b}$, the single level reformulation for the inverse problem is

$$\begin{aligned} \min_{\mathbf{b}\in\Theta} \quad & \tfrac{1}{2}\|\mathbf{b} - \mathbf{b}_t\|_2^2 + \eta_t\|\mathbf{y}_t - \mathbf{x}\|_2^2 \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{b}, \\ & \mathbf{u} \leq M\mathbf{z} \\ & \mathbf{A}\mathbf{x} - \mathbf{b} \leq M(1-\mathbf{z}) \\ & Q\mathbf{x} + \mathbf{c}_t - \mathbf{A}^T\mathbf{u} = 0 \\ & \mathbf{b} \in \mathbb{R}^m, \;\; \mathbf{x} \in \mathbb{R}^n, \;\; \mathbf{u} \in \mathbb{R}^m_+, \;\; \mathbf{z} \in \{0,1\}^m \end{aligned} \tag{A.4}$$

where M is an appropriate number used to bound $\mathbf{u}$ and $\mathbf{A}\mathbf{x} - \mathbf{b}$.

We have a similar single level reformulation when learning the objective $\mathbf{c}$. Clearly, this is a Mixed Integer Second Order Cone program(MISOCP) when learning either $\mathbf{c}$ or $\mathbf{b}$.

A.2 Omitted Proofs

A.2.1 Proof of Lemma 2.3.1

Proof. By Assumption 2.3.1(b), we know that $S(u,\theta)$ is a single-valued set for each $u \in \mathcal{U}$.

$\forall \mathbf{y} \in \mathcal{Y}$, $\forall u \in \mathcal{U}$, $\forall \theta_1, \theta_2 \in \Theta$, without of loss of generality, let $l(\mathbf{y}, u, \theta_1) \geq l(\mathbf{y}, u, \theta_2)$. Then,

$$\begin{aligned} |l(\mathbf{y}, u, \theta_1) - l(\mathbf{y}, u, \theta_2)| &= l(\mathbf{y}, u, \theta_1) - l(\mathbf{y}, u, \theta_2) \\ &= \|\mathbf{y} - S(u,\theta_1)\|_2^2 - \|\mathbf{y} - S(u,\theta_2)\|_2^2 \\ &= \langle S(u,\theta_2) - S(u,\theta_1), 2\mathbf{y} - S(u,\theta_1) - S(u,\theta_2)\rangle \\ &\leq 2(B+R)\|S(u,\theta_2) - S(u,\theta_1)\|_2 \end{aligned} \tag{A.5}$$

The last inequality is due to Cauchy-Schwartz inequality and the Assumptions 2.3.1(a), that is

$$\|2\mathbf{y} - S(u,\theta_1) - S(u,\theta_2)\|_2 \leq 2(B+R) \tag{A.6}$$

Next, we will apply Proposition 6.1 in [57] to bound $\|S(u,\theta_2) - S(u,\theta_1)\|_2$.

Under Assumptions 2.2.1 - 2.3.2, the conditions of Proposition 6.1 in [57] are satisfied. Therefore,

$$\|S(u,\theta_2) - S(u,\theta_1)\|_2 \leq \frac{2\kappa}{\lambda}\|\theta_1 - \theta_2\|_2 \tag{A.7}$$

Plugging (A.6) and (A.7) in (A.5) yields the claim. □

A.2.2 Proof of Theorem 2.3.2

Proof. we will use Theorem 3.2 in [56] to prove our theorem.

Let $G_t(\theta) = \frac{1}{2}\|\theta - \theta_t\|_2^2 + \eta_t l(\mathbf{y}_t, u_t, \theta)$.

We will now show the loss function is convex. The first step is to show that if Assumption 2.3.3 holds, then the loss function $l(\mathbf{y}, u, \theta)$ is convex in θ. $\forall \mathbf{y} \in \mathcal{Y}$, $\forall u \in \mathcal{U}$, $\forall \theta_1, \theta_2 \in \Theta$, we have

$$
\begin{aligned}
& \alpha l(\mathbf{y}, u, \theta_1) + \beta l(\mathbf{y}, u, \theta_2) - l(\mathbf{y}, u, \alpha\theta_1 + \beta\theta_2) \\
= \;& \alpha\|\mathbf{y} - S(u, \theta_1)\|_2^2 + \beta\|\mathbf{y} - S(u, \theta_2)\|_2^2 - \|\mathbf{y} - S(u, \alpha\theta_1 + \beta\theta_2)\|_2^2 \\
= \;& \alpha\|\mathbf{y} - S(u, \theta_1)\|_2^2 + \beta\|\mathbf{y} - S(u, \theta_2)\|_2^2 - \|\mathbf{y} - \alpha S(u, \theta_1) - \beta S(u, \theta_2)\|_2^2 \\
& + \|\mathbf{y} - \alpha S(u, \theta_1) - \beta S(u, \theta_2)\|_2^2 - \|\mathbf{y} - S(u, \alpha\theta_1 + \beta\theta_2)\|_2^2 \\
= \;& \alpha\beta\|S(u, \theta_1) - S(u, \theta_2)\|_2^2 + \|\mathbf{y} - \alpha S(u, \theta_1) - \beta S(u, \theta_2)\|_2^2 - \|\mathbf{y} - S(u, \alpha\theta_1 + \beta\theta_2)\|_2^2 \\
= \;& \alpha\beta\|S(u, \theta_1) - S(u, \theta_2)\|_2^2 \\
& - \langle \alpha S(u, \theta_1) + \beta S(u, \theta_2) - S(u, \alpha\theta_1 + \beta\theta_2), \\
& 2\mathbf{y} - S(u, \alpha\theta_1 + \beta\theta_2) - \alpha S(u, \theta_1) - \beta S(u, \theta_2)\rangle \\
\geq \;& \alpha\beta\|S(u, \theta_1) - S(u, \theta_2)\|_2^2 - \|\alpha S(u, \theta_1) + \beta S(u, \theta_2) \\
& - S(u, \alpha\theta_1 + \beta\theta_2)\|_2 \|2\mathbf{y} - S(u, \alpha\theta_1 \\
& + \beta\theta_2) - \alpha S(u, \theta_1) - \beta S(u, \theta_2)\|_2
\end{aligned}
\tag{A.8}
$$

The last inequality is by Cauchy-Schwartz inequality. Note that

$$
\begin{aligned}
& \|\alpha S(u, \theta_1) + \beta S(u, \theta_2) - S(u, \alpha\theta_1 + \beta\theta_2)\|_2 \\
& \|2\mathbf{y} - S(u, \alpha\theta_1 + \beta\theta_2) - \alpha S(u, \theta_1) - \beta S(u, \theta_2)\|_2 \\
& \leq 2(B + R)\|\alpha S(u, \theta_1) + \beta S(u, \theta_2) - S(u, \alpha\theta_1 + \beta\theta_2)\|_2 \\
& \leq \alpha\beta\|S(u, \theta_1) - S(u, \theta_2)\|_2
\end{aligned}
\tag{A.9}
$$

Plugging (A.9) in (A.8) yields the result.

Using Theorem 3.2 in [56], for $\alpha_t \leq \frac{G_t(\theta_{t+1})}{G_t(\theta_t)}$, we have

$$
R_T \leq \; \sum_{t=1}^{T} \frac{1}{\eta_t}(1 - \alpha_t)\eta_t l(\mathbf{y}_t, u_t, \theta_t) + \frac{1}{2\eta_t}(\|\theta_t - \theta^*\|_2^2 - \|\theta_{t+1} - \theta^*\|_2^2) \tag{A.10}
$$

Notice that

$$\begin{aligned} G_t(\theta_t) - G_t(\theta_{t+1}) &= \eta_t(l(\mathbf{y}_t, u_t, \theta_t) - l(\mathbf{y}_t, u_t, \theta_{t+1})) - \tfrac{1}{2}\|\theta_t - \theta_{t+1}\|_2^2 \\ &\leq \tfrac{4(B+R)\kappa\eta_t}{\lambda}\|\theta_t - \theta_{t+1}\|_2 - \tfrac{1}{2}\|\theta_t - \theta_{t+1}\|_2^2 \\ &\leq \tfrac{8(B+R)^2\kappa^2\eta_t^2}{\lambda^2} \end{aligned} \tag{A.11}$$

The first inequality follows by applying Lemma 2.3.1.

Let $\alpha_t = \frac{R_t(\theta_{t+1})}{R_t(\theta_t)}$. Using (A.11), we have

$$(1-\alpha_t)\eta_t l(\mathbf{y}_t, u_t, \theta_t) = (1-\alpha_t)G_t(\theta_t) = G_t(\theta_t) - G_t(\theta_{t+1}) \leq \tfrac{8(B+R)^2\kappa^2\eta_t^2}{\lambda^2} \tag{A.12}$$

Plug (A.12) in (A.10), and note the telescoping sum,

$$R_T \leq \sum_{t=1}^{T} \frac{8(B+R)^2\kappa^2\eta_t}{\lambda^2} + \sum_{t=1}^{T} \frac{1}{2\eta_t}(\|\theta_t - \theta^*\|_2^2 - \|\theta_{t+1} - \theta^*\|_2^2)$$

Setting $\eta_t = \frac{D\lambda}{2(B+R)\kappa\sqrt{2t}}$, we can simplify the second summation to $\frac{D(B+R)\kappa\sqrt{2}}{\lambda}$ since the sum telescopes and $\theta_1 = \mathbf{0}, \|\theta^*\|_2 \leq D$. The first sum simplifies using $\sum_{t=1}^{T} \frac{1}{\sqrt{t}} \leq 2\sqrt{T} - 1$ to obtain the result

$$R_T \leq \frac{4\sqrt{2}(B+R)D\kappa}{\lambda}\sqrt{T}.$$

□

A.2.3 Proof of Theorem 2.3.3

Proof. Since $f(\mathbf{x}, u, \theta)$ is strongly convex in $\mathbf{x}$ on $\mathbb{R}^n$ by Assumption 2.3.1, it is also strictly convex in $\mathbf{x}$ on $\mathbb{R}^n$. Then, all the conditions required in Theorem 3. of [7] are naturally satisfied under our assumptions. Applying that theorem yields

$$\frac{1}{T}\sum_{t\in[T]} l(\mathbf{y}_t, u_t, \theta^T) \xrightarrow{p} \mathbb{E}\left[l(\mathbf{y}, u, \theta^*)\right] \tag{A.13}$$

where $\theta^T = \arg\min_{\theta\in\Theta}\{\sum_{t\in[T]} l(\mathbf{y}_t, u_t, \theta)\}$ is the estimation of the parameter in batch setting.

From Theorem 3.2 we have

$$\frac{1}{T}\sum_{t\in[T]} l(\mathbf{y}_t, u_t, \theta_t) - \frac{1}{T}\sum_{t\in[T]} l(\mathbf{y}_t, u_t, \theta^T) \leq \frac{4\sqrt{2}(B+R)D\kappa}{\lambda\sqrt{T}} \xrightarrow{p} 0 \tag{A.14}$$

Adding (A.13) and (A.14) up, we have the risk consistency result

$$\frac{1}{T}\sum_{t\in[T]} l(\mathbf{y}_t, u_t, \theta_t) \xrightarrow{p} \mathbb{E}\left[l(\mathbf{y}, u, \theta^*)\right]$$

□

A.2.4 Proof of Corollary 2.3.3.1

Proof. Note that $\forall\theta \in \Theta$,

$$\mathbb{E}\left[l(\mathbf{y}, u, \theta)\right] = \mathbb{E}\left[\min_{\tilde{\mathbf{x}}\in S(u,\theta)} \|\mathbf{x} + \epsilon - \tilde{\mathbf{x}}\|_2^2\right] = \mathbb{E}\left[\min_{\tilde{\mathbf{x}}\in S(u,\theta)} \|\mathbf{x} - \tilde{\mathbf{x}}\|_2^2\right] + \mathbb{E}[\epsilon^T\epsilon] \geq \mathbb{E}[\epsilon^T\epsilon]$$

We further notice that $\mathbb{E}\left[\min_{\tilde{\mathbf{x}}\in S(u,\theta_0)} \|\mathbf{x} - \tilde{\mathbf{x}}\|_2^2\right] = 0$, since $\mathbf{x} \in S(u, \theta_0)$. Therefore, we have

$$\mathbb{E}\left[l(\mathbf{y}, u, \theta^*)\right] = \mathbb{E}\left[l(\mathbf{y}, u, \theta_0)\right] = \mathbb{E}[\epsilon^T\epsilon]$$

Then, applying Theorem 2.3.3 yields the result, since we have shown $\mathbb{E}\left[l(\mathbf{y}, u, \theta^*)\right] = \mathbb{E}[\epsilon^T\epsilon]$. □

A.3 Omitted Examples

A.3.1 Examples for which Assumption 3.3 holds

Consider for example the following quadratic program

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & \tfrac{1}{2}\mathbf{x}^T Q\mathbf{x} + (\mathbf{c}+u)^T\mathbf{x} \\ s.t. \quad & A\mathbf{x} \geq \mathbf{b} \end{aligned} \tag{A.15}$$

where Q is a positive semidefinite matrix, and u is the external signal.

Suppose that the parameter we seek to learn is $\mathbf{c}$, all the others are given. If for each $u \in \mathcal{U}$, the optimal solution for the above program is in the interior of the feasible region. Then,

$$S(u,\mathbf{c}_1) = -Q^{-1}(\mathbf{c}_1+u); \;\; S(u,\mathbf{c}_2) = -Q^{-1}(\mathbf{c}_2+u);$$
$$S(u,\alpha\mathbf{c}_1+\beta\mathbf{c}_2) = -Q^{-1}(\alpha\mathbf{c}_1+\beta\mathbf{c}_2+u).$$

Then, we have

$$0 = \|\alpha S(u,\mathbf{c}_1)+\beta S(u,\mathbf{c}_2) - S(u,\alpha\mathbf{c}_1+\beta\mathbf{c}_2)\|_2 \leq \alpha\beta\|S(u,\theta_1)-S(u,\theta_2)\|_2/(2(B+R)).$$

A.4 Data for the Applications

A.4.1 Data for Learning the Consumer Behavior

Table 10: True $\mathbf{r}$

1.180	1.733	1.564	0.040	2.443	1.055	4.760	5.000	1.258	4.933

Table 11: True Q

2.360	0	0	0	0	0	0	0	0	0
0	3.465	0	0	0	0	0	0	0	0
0	0	3.127	0	0	0	0	0	0	0
0	0	0	0.0791	0	0	0	0	0	0
0	0	0	0	4.886	0	0	0	0	0
0	0	0	0	0	2.110	0	0	0	0
0	0	0	0	0	0	9.519	0	0	0
0	0	0	0	0	0	0	9.999	0	0
0	0	0	0	0	0	0	0	2.517	0
0	0	0	0	0	0	0	0	0	9.867

A.4.2 Data for Learning the Transportation Cost

We let $\lambda_1 = 2$, $\lambda_2 = 10$, $u_e = 1.3$ for all $e \in E$, $y_1 = 3$ and $y_2 = 1.5$.

Table 12: True transportation cost for each edge

c_{13}	c_{14}	c_{23}	c_{25}	c_{34}	c_{35}
3.124	4.119	3.814	1.071	5.398	2.899

Appendix B

Nomenclature

$\hat{\theta}^N$ The estimator of θ constructed by solving 3.6

$\hat{\theta}_K$ The estimator of θ constructed by solving 3.8

$\hat{\theta}_K^N$ The estimator of θ constructed by solving 3.9

λ_l f_l in 3.1 is strongly convex with parameter λ_l

$\mathscr{W}_p$ The standard $(p-1)$-simplex given by $\{w \in \mathbb{R}_+^p : \mathbf{1}^T w = 1\}$

$\mathscr{W}_p^+$ The interior of the standard $(p-1)$-simplex

$\mathbf{x_y}$ The nearest point to $\mathbf{y}$ in $X_P(\theta)$

Θ The parameter space for θ

θ The parameter that determines 3.1

θ^* The estimator of θ constructed by solving 3.5

$\{\mathbf{y}_i\}_{i\in[N]}$ The observed noisy decisions

B The upper bound for the radius of the feasible region $X(\theta)$

$d_{sH}(X,Y)$ The Hausdorff semi-distance between two sets X and Y

K The number of weight samples in (3.7)

$l(\mathbf{y},\theta)$ The unsupervized learning (3.4)

$l_K(\mathbf{y},\theta)$ The unsupervized learning (3.7)

N The number of observations

p The number of objective functions in 3.1

q The number of constraints in 3.1

R The upper bound for the radius of the support $\mathcal{Y}$ of the observations

$Rad_N(\mathcal{F})$ The Rademacher complexity of the function class $\mathcal{F}$

$S(w,\theta)$ The set of optimal solutions for 3.2

$w_{\mathbf{y}}^K$ The nearest weight to $w_{\mathbf{y}}$ among $\{w_k\}_{k\in[K]}$

$w_{\mathbf{y}}^{NK}$ The estimated weight for $\mathbf{y}$ using $\hat{\theta}_K^N$

$w_{\mathbf{y}}$ The weight such that $\mathbf{x}_{\mathbf{y}} = S(w_{\mathbf{y}}, \theta)$

$w_K^{\mathbf{y}}$ The weight among $\{w_k\}_{k\in[K]}$ such that $l_K(\mathbf{y}, \theta) = \|\mathbf{y} - S(w_{\mathbf{y}}^K, \theta)\|_2^2$

$X(\theta)$ Feasible region for 3.1

$X_P(\theta)$ The set of all Pareto optimal solutions for 3.1

Appendix C

C.1 Why IMOP instead of IOP?

Consider a scenario where decision makers are subject to same restrictions but need to make their individualized *optimal* decisions considering two objective functions, as in the following bi-objective linear programming problem with $a > b > 0$ and $c > 0$. Figure 23 displays the feasible region of an instance with $a = 6, b = 1, c = 1$, i.e., the triangle AOB.

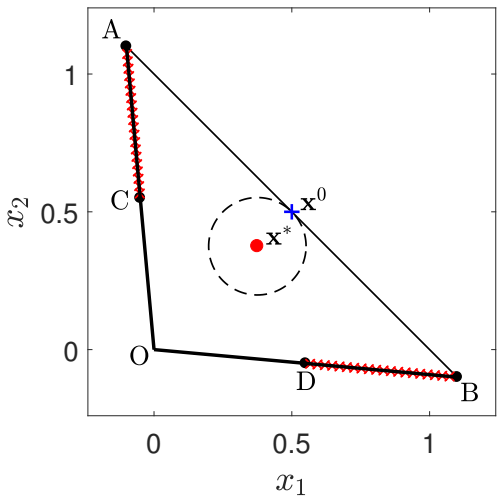


Figure 23: $O(0,0)$, $A(-0.2, 1.2)$, and $B(1.2, -0.2)$ are the Vertices of the Feasible Region. $C(-0.1, 0.6)$, $D(0.6, -0.1)$ and $\mathbf{x}^0(0.5, 0.5)$ are the Midpoints of OA, OB, and AB, Respectively. The Red Dot $\mathbf{x}^*(0.375, 0.375)$ is the Geometric Mean of All the Points in Segments AC and BD. The Bold Segments OA and OB are the Pareto Optimal (solution) Set for the Bi-objective Linear Programming Problem.

$$\min\ x_1 \tag{C.1a}$$

$$\min\ x_2 \tag{C.1b}$$

$$s.t.\ ax_1 + bx_2 \geq 0, \tag{C.1c}$$

$$bx_1 + ax_2 \geq 0, \tag{C.1d}$$

$$x_1 + x_2 \leq c. \tag{C.1e}$$

With multiple objectives, rational decision makers seek Pareto optimal solutions, which are those that cannot be improved without sacrificing performances in one or more criteria (see section 3.2.1). In Figure 23, it is straightforward to see that points on edges OA and OB are Pareto optimal solutions that could be selected by rational decision makers. Assume that many observed decisions evenly occur in segments AC and BD. If they are treated as noisy observations of a pristine solution to $\min\{\mathbf{c}^T\mathbf{x} : (\text{ C.1c}) - (\text{ C.1e})\}$, we can infer the coefficient $\mathbf{c}$ and obtain a denoised solution $\mathbf{x}^*$ through computing the IOP model with the quadratic loss function. Actually, noting that optimal $\mathbf{x}^*$ minimizes the averaged distance to those observations, we can derive its analytical characterization.

Specifically, the sum of squares of the Euclidean distance between $\mathbf{x}^*$ and evenly distributed observations on AC and BD can be represented as the following integral:

$$\int_{\frac{bc}{2(a-b)}}^{\frac{bc}{a-b}} \|\begin{pmatrix} x_1^* \\ x_2^* \end{pmatrix} - \begin{pmatrix} -v \\ \frac{a}{b}v \end{pmatrix}\|_2^2 \, d\, v + \int_{\frac{bc}{2(a-b)}}^{\frac{bc}{a-b}} \|\begin{pmatrix} x_1^* \\ x_2^* \end{pmatrix} - \begin{pmatrix} \frac{a}{b}v \\ -v \end{pmatrix}\|_2^2 \, d\, v \tag{C.2}$$

$$= \frac{bc}{a-b}(x_1^* - \frac{3}{8}c)^2 + \frac{bc}{a-b}(x_2^* - \frac{3}{8}c)^2 + \Delta, \tag{C.3}$$

where Δ depends on a, b and c only. Thus, $\mathbf{x}^* = (\frac{3}{8}c, \frac{3}{8}c)$, the arithmetic mean of observations, minimizes this integration. As $\mathbf{x}^*$ is an interior point, the only $\mathbf{c}$ that renders $\mathbf{x}^*$ optimal is the trivial one, i.e., $(c_1, c_2) = (0, 0)$, which does not have any relevance to the actual objective functions.

Indeed, we still cannot obtain reasonable explanation of the data, even if taking an additional consideration by restricting $\mathbf{x}^*$ to be on the boundary of the feasible region, which helps to avoid the previous trivial estimation. Note from Figure 23 $\mathbf{x}^0$, i.e., the projection of $\mathbf{x}^*$ on AB, is the optimal boundary point to that integration. Because $\mathbf{x}^0$ is in the interior

of AB, the unique $\mathbf{c}$ that renders $\mathbf{x}^0$ optimal is $(c_1, c_2) = (-1, -1)$. This inference basically reflects opposite information regarding decision makers' intentions or desires.

C.2 Omitted Proofs

C.2.1 Proof of Proposition 3.3.1

Proof. Denote $\{\mathbf{y}_i\}_{i\in[N]}$ these points, denote K the number of clusters, and denote $\{\mathbf{x}_k\}_{k\in[K]}$ the set of optimal centroids. We then construct an equivalent instance of 3.9 as follows. Note that an IMOP is determined by a 3.1, the parameter space of θ, and a set of weight samples $\{w_k\}_{k\in[K]}$.

First, let us consider the 3.1 whose objective functions are quadratic and feasible region is a ball in $\mathbb{R}^n$ that has the following form

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & \begin{bmatrix} \frac{1}{2}\mathbf{x}^T\mathbf{x} + \mathbf{c}_1^T\mathbf{x} \\ \vdots \\ \frac{1}{2}\mathbf{x}^T\mathbf{x} + \mathbf{c}_K^T\mathbf{x} \end{bmatrix} \\ s.t. \quad & \|\mathbf{x}\|_2 \leq 2\max_{i\in[N]}\|\mathbf{y}_i\|_2, \end{aligned} \tag{C.4}$$

where $\mathbf{c}_l \in \mathbb{R}^n$ for $l \in [K]$. The task of IMOP for C.4 is to learn $\{\mathbf{c}_l\}_{l\in[K]}$ given $\{\mathbf{y}_i\}_{i\in[N]}$. Since the objective functions and the constraint are convex, C.4 is a convex 3.1.

Second, we let Θ be the parameter space that consists of $\mathbf{c}_l \in \mathbb{R}^n$ such that $\|\mathbf{c}_l\|_2 \leq \max_{i\in[N]}\|\mathbf{y}_i\|_2$ for each $l \in [K]$. One can readily check Θ defined in such a way is a convex and compact set. Assume $\{\mathbf{x}_k\}_{k\in[K]}$ are the optimal centroids for $\{\mathbf{y}_i\}_{i\in[N]}$ in K-means clustering. Since each of the optimal centroid is the mean of the points in that cluster, we have $\mathbf{x}_k \leq \max_{i\in[N]}\|\mathbf{y}_i\|_2$. Now, let $\mathbf{c}_k = -\mathbf{x}_k$, which is achievable because $\|\mathbf{c}_k\|_2 \leq \max_{i\in[N]}\|\mathbf{y}_i\|_2$ for each $k \in [K]$.

Third, we select $\{w_k\}_{k\in[K]}$ in the following way: the K weights are the K vertices of $\mathscr{W}_K$.

With a slight abuse of notation, let $\mathbf{x}_k = S(w_k, \mathbf{c}_1, \ldots, \mathbf{c}_K)$ be the optimal solution of 3.2 for C.4 for each $k \in [K]$. This mild abuse of notation allows us to express our results in

a unified manner. It will be clear from context whether we are treating K-means clustering problem or 3.9, and consequently be clear which definition of $\mathbf{x}_k$ we mean. Since the objective functions in C.4 are strictly convex, each $\mathbf{x}_k$ is a Pareto optimal point by Proposition 3.2.1.

Now, we are ready to show the equivalence between 3.12 and the constructed IMOP. Note that the only difference between the two problems is that $\{\mathbf{x}_k\}_{k\in[K]}$ for IMOP are restricted to be Pareto optimal points for some 3.1, while no restriction is put on $\{\mathbf{x}_k\}_{k\in[K]}$ for K-means clustering. Thus, the optimal value of the K-means clustering provides a lower bound for the optimal value of the constructed IMOP. Then, it suffices to show that they have the same optimal value, which can be done by proving that the previously defined $\{\mathbf{c}_k\}_{k\in[K]}$ and the optimal centroids $\{\mathbf{x}_k\}_{k\in[K]}$ solve 3.9.

Since the K weights are vertices of the simplex $\mathscr{W}_K$, each Pareto optimal point $\mathbf{x}_k$ corresponds to the unique optimal solution for one single objective optimization problem. More specifically,

$$\mathbf{x}_k = \arg\min_{\mathbf{x}}\{\frac{1}{2}\mathbf{x}^T\mathbf{x} + \mathbf{c}_k^T\mathbf{x} : \|\mathbf{x}\|_2 \leq 2\max_{i\in[N]}\|\mathbf{y}_i\|_2\}. \tag{C.5}$$

One can readily check that the previously defined $\mathbf{c}_k = -\mathbf{x}_k$ indeed make the optimal centroid $\mathbf{x}_k$ of the K-means clustering problem also an optimal solution in (C.5). It shows that the construed 3.9 is solved bt $\{\mathbf{c}_k\}_{k\in[K]}$ and the optimal centroids $\{\mathbf{x}_k\}_{k\in[K]}$.

To this end, we have shown that the optimal values of 3.9 we constructed and 3.12 are indeed equal. Therefore, solving the 3.9 we constructed provides an optimal partition of $\{\mathbf{y}_i\}_{i\in[N]}$ for the K-means clustering. □

C.2.2 Proof of Theorem 3.3.5

Proof. Under our assumption, for each Pareto optimal point $\mathbf{y}$, $\exists w \in \mathscr{W}_p$, s.t. $\mathbf{y} \in S(w,\theta)$ by PROPOSITION 3.2.2. Recall that $w \in \mathbb{R}^p_+, \mathbf{1}^T w = 1$, thus w lies in a $(p-1)$-d manifold. We show that the mapping $S(w,\theta) : w \to \mathbf{y}$ for fixed θ is continuous in w in Lemma 3.4.6. By the definition of manifold, the Pareto optimal set is a $(p-1)$-d manifold. □

C.2.3 Proof of Lemma 3.4.1

Proof. Since $\mathbf{g}(\mathbf{x}, \theta)$ is continuous and thus l.s.c. on $\mathbb{R}^n \times \Theta$ by ASSUMPTION 3.4.1, $X(\theta)$ is u.s.c. for each $\theta \in \Theta$ by Theorem 10 in [80]. From ASSUMPTION 2.2.1, we know that $\mathbf{g}(\mathbf{x}, \theta)$ is convex in $\mathbf{x}$ for each $\theta \in \Theta$. From ASSUMPTION 3.4.1, $X(\theta)$ has a nonempty relatively interior. Namely, there exists a $\bar{\mathbf{x}} \in \mathbb{R}^n$ such that $\mathbf{g}(\bar{\mathbf{x}}, \theta) < \mathbf{0}$. Then, $X(\theta)$ is l.s.c. for each $\theta \in \Theta$ by Theorem 12 in [80]. Hence, $X(\theta)$ is continuous on Θ. □

C.2.4 Proof of Lemma 3.4.2

Proof. First, we will show that $X_P(\theta)$ is u.s.c. on Θ. Since $\mathbf{f}(\mathbf{x}, \theta)$ is strictly convex in $\mathbf{x}$ for each $\theta \in \Theta$, the Pareto optimal set $X_P(\theta)$ coincides with the weakly Pareto optimal set. In addition, we know that $X(\theta)$ is continuous on Θ by Lemma 3.4.1. Also, note the pointed convex cone we use throughout this paper has the same meaning as the domination structure D in [81], and we set $D = \mathbb{R}^p_+$. To this end, we can readily verify that the sufficient conditions for upper semicontinuity in Theorem 7.1 of [81] are satisfied. Thus, $X_P(\theta)$ is u.s.c..

Next, we will show that $X_P(\theta)$ is l.s.c. on Θ. Theorem 7.2 of [81] provides the sufficient conditions for the lower semicontinuity of $X_P(\theta)$. All of these conditions are naturally satisfied under Assumptions 2.2.1 - 3.4.1 except the one that requires $\mathbf{f}(\mathbf{x}, \theta)$ to be one-to-one, i.e., injective in $\mathbf{x}$. Next, we will show that the one-to-one condition can be safely replaced by the strict quasi-convexity of $\mathbf{f}(\mathbf{x}, \theta)$ in $\mathbf{x}$.

Theorem 7.2 of [81] is a direct result of part (ii) in Lemma 7.2 of [81]. To complete our proof, we only need to sightly modify the last part of the proof in Lemma 7.2. In what follows we will use notations in that paper.

Since strict convexity implies strict quasi-convexity, f is strictly quasi-convex. Suppose that $f(\bar{x}, \hat{u}) = f(\hat{x}, \hat{u})$ does not imply $\bar{x} = \hat{x}$. Let $z = \frac{\bar{x}+\hat{x}}{2}$. By the strict quasi-convexity of f, we have

$$f(z, \hat{u}) = f(\frac{\bar{x} + \hat{x}}{2}, \hat{u}) < \max\{f(\bar{x}, \hat{u}), f(\hat{x}, \hat{u})\} = f(\hat{x}, \hat{u}). \tag{C.6}$$

This contradicts the fact that $\hat{x} \in M(\hat{u})$, where $M(\hat{u})$ is the Pareto optimal set given $\hat{u}$. Hence, $\bar{x}$ must be equal to $\hat{x}$. The remain part of the proof is the same as that of Lemma 7.2. □

C.2.5 Proof of Proposition 3.4.3

Proof. We apply Theorem 2 of [79] in our proof. We start by checking that the three conditions for using this theorem are satisfied. First, by Lemma 3.4.2, $X_P(\theta)$ is continuous. Then, applying Berge Maximum Theorem [102] to 3.6 implies that the empirical risk $M^N(\theta)$ is continuous. Second, by Assumption 3.4.1, Θ is a compact set. Third, $\forall \mathbf{y} \in \mathcal{Y}, \min_{\mathbf{x} \in X_P(\theta)} \|\mathbf{y} - \mathbf{x}\|_2^2 \leq \|\mathbf{y}\|_2^2 + B^2 + 2B\|\mathbf{y}\|_2$ and the right-hand side is integrable with respect to $\mathbf{y}$ under Assumption 3.4.1. Consequently, all three conditions are satisfied and the proof is concluded. □

C.2.6 Proof of Proposition 3.4.4

Proof. Similar to Proposition 3.4.3, the key step is to show the continuity of $M_K^N(\theta)$ in θ for each K. It suffices to show that $\bigcup_{k \in [K]} S(w_k, \theta)$ is continuous in θ for all K. First, let us establish the continuity of $S(w_k, \theta)$ in θ for each $k \in [K]$. Note that the feasible region $X(\theta)$ is irrelevant to w. Thus, applying the Berge Maximum Theorem [102] to (3.2) implies that $S(w_k, \theta)$ is upper semicontinuous in θ. Hence, $S(w_k, \theta)$ is continuous in θ as it is a single-valued set. Second, let us show the continuity of $\bigcup_{k \in [K]} S(w_k, \theta)$ in θ. By Propositions 2 and 4 of [80], we know that a finite union of continuous sets, i.e., $\bigcup_{k \in [K]} S(w_k, \theta)$, is continuous in θ. Finally, applying Theorem 2 of [79] yields the uniform convergence of $M_K^N(\theta)$ to $M_K(\theta)$ in N. □

C.2.7 Proof of Lemma 3.4.5

Proof. **(a)** Let $K_2 \geq K_1$. Under our setting, $K_2 \geq K_1$ implies $\{w_k\}_{k \in [K_1]} \subseteq \{w_k\}_{k \in [K_2]}$. By the definition of $l_K(\mathbf{y}, \theta)$, we have $l_{K_1}(\mathbf{y}, \theta) \geq l_{K_2}(\mathbf{y}, \theta)$ for all $\mathbf{y} \in \mathcal{Y}$, and thus $M_{K_1}(\theta) \geq M_{K_2}(\theta)$ for all $\theta \in \Theta$. Therefore, $\{M_K(\theta)\}$ is monotone decreasing in K.

Recall the definition of $\hat{\theta}_K$ in Table 3, we know $\hat{\theta}_{K_2}$ minimizes $M_{K_2}(\theta)$. Therefore, $M_{K_2}(\hat{\theta}_{K_1}) \geq M_{K_2}(\hat{\theta}_{K_2})$. In addition, $M_{K_1}(\hat{\theta}_{K_1}) \geq M_{K_2}(\hat{\theta}_{K_1})$ by the first part of **(a)**. Consequently,

$$M_{K_1}(\hat{\theta}_{K_1}) \geq M_{K_2}(\hat{\theta}_{K_1}) \geq M_{K_2}(\hat{\theta}_{K_2}).$$

Therefore, $M_{K_1}(\hat{\theta}_{K_1}) \geq M_{K_2}(\hat{\theta}_{K_2})$ for $K_2 \geq K_1$.

Similarly, we can readily show that $M_K(\hat{\theta}_K) \geq M(\theta^*)$ by noting that

$$M_K(\hat{\theta}_K) \geq M(\hat{\theta}_K) \geq M(\theta^*).$$

The first inequality is a direct result of the first part of **(a)**; the second inequality follows from the fact that θ^* minimizes $M(\theta)$ by definition.

(b) Let $K_2 \geq K_1$. By the definition of $l_K(\mathbf{y}, \theta)$, we have $l_{K_1}(\mathbf{y}_i, \theta) \geq l_{K_2}(\mathbf{y}_i, \theta)$ for all $i \in [N]$, and thus $M^N_{K_1}(\theta) \geq M^N_{K_2}(\theta)$ for all $\theta \in \Theta$. Therefore, $\{M^N_K(\theta)\}$ is monotone decreasing in K.

Recall the definition of $\hat{\theta}^N_K$ in Table 3, we know $\hat{\theta}^N_{K_2}$ minimizes $M^N_{K_2}(\theta)$. Therefore, $M^N_{K_2}(\hat{\theta}^N_{K_1}) \geq M^N_{K_2}(\hat{\theta}^N_{K_2})$. In addition, $M^N_{K_1}(\hat{\theta}^N_{K_1}) \geq M^N_{K_2}(\hat{\theta}^N_{K_1})$ by the first part of **(b)**. Consequently,

$$M^N_{K_1}(\hat{\theta}^N_{K_1}) \geq M^N_{K_2}(\hat{\theta}^N_{K_1}) \geq M^N_{K_2}(\hat{\theta}^N_{K_2}). \tag{C.7}$$

Hence, $M^N_{K_1}(\hat{\theta}^N_{K_1}) \geq M^N_{K_2}(\hat{\theta}^N_{K_2})$ for $K_2 \geq K_1$.

Finally, we can show $M^N_K(\hat{\theta}^N_K) \geq M^N(\hat{\theta}^N)$ by noting that $M^N_K(\hat{\theta}^N_K) \geq M^N(\hat{\theta}^N_K) \geq M^N(\hat{\theta}^N)$. □

C.2.8 Proof of Lemma 3.4.6

Proof. $\forall w \in \mathscr{W}_p$, one can readily check that $w^T\mathbf{f}(\cdot,\theta)$ is strongly convex for each θ and thus

$$w^T\mathbf{f}(\mathbf{y},\theta) \geq w^T\mathbf{f}(\mathbf{x},\theta) + \nabla w^T\mathbf{f}(\mathbf{x},\theta)^T(\mathbf{y}-\mathbf{x}) + \frac{\lambda}{2}\|\mathbf{y}-\mathbf{x}\|_2^2. \tag{C.8}$$

Thus, the second-order growth condition holds for $w^T\mathbf{f}(\cdot,\theta)$ for all $\theta \in \Theta$. That is,

$$w^T\mathbf{f}(\mathbf{x},\theta) \geq w^T\mathbf{f}(S(w,\theta),\theta) + \frac{\lambda}{2}\|(S(w,\theta)-\mathbf{x}\|_2^2. \tag{C.9}$$

In addition, $\forall w, w_0 \in \mathscr{W}_p$, we have

$$\begin{aligned} |w^T\mathbf{f}(\mathbf{x},\theta) - w_0^T\mathbf{f}(\mathbf{x},\theta)| &= |(w-w_0)^T\mathbf{f}(\mathbf{x},\theta)| \\ &\leq \|w-w_0\|_2\|\mathbf{f}(\mathbf{x},\theta)\|_2 \qquad \text{(Cauchy-Schwarz inequality)} \\ &\leq L\|w-w_0\|_2. \end{aligned} \tag{C.10}$$

□

C.2.9 Proof of Lemma 3.4.7

Proof. By definition,

$$l_K(\mathbf{y},\theta) - l(\mathbf{y},\theta) = \min_{\mathbf{x}\in\bigcup_{k\in[K]} S(w_k,\theta)} \|\mathbf{y}-\mathbf{x}\|_2^2 - \min_{\mathbf{x}\in X_P(\theta)} \|\mathbf{y}-\mathbf{x}\|_2^2 \geq 0. \tag{C.11}$$

Let $\|\mathbf{y} - S(w_K^{\mathbf{y}},\theta)\|_2^2 = \min_{\mathbf{x}\in\bigcup_{k\in[K]} S(w_k,\theta)} \|\mathbf{y}-\mathbf{x}\|_2^2$, and $\|\mathbf{y} - S(w_{\mathbf{y}},\theta)\|_2^2 = \min_{\mathbf{x}\in X_P(\theta)} \|\mathbf{y}-\mathbf{x}\|_2^2$. Let $w_{\mathbf{y}}^K$ be the closest weight sample among $\{w_k\}_{k\in[K]}$ to $w_{\mathbf{y}}$. Then,

$$\begin{aligned} l_K(\mathbf{y},\theta) - l(\mathbf{y},\theta) &= \|\mathbf{y} - S(w_K^{\mathbf{y}},\theta)\|_2^2 - \|\mathbf{y} - S(w_{\mathbf{y}},\theta)\|_2^2 \\ &\leq \|\mathbf{y} - S(w_{\mathbf{y}}^K,\theta)\|_2^2 - \|\mathbf{y} - S(w_{\mathbf{y}},\theta)\|_2^2 \\ &= \left(2\mathbf{y} - S(w_{\mathbf{y}}^K,\theta) - S(w_{\mathbf{y}},\theta)\right)^T \left(S(w_{\mathbf{y}},\theta) - S(w_{\mathbf{y}}^K,\theta)\right) \\ &\leq \|2\mathbf{y} - S(w_{\mathbf{y}}^K,\theta) - S(w_{\mathbf{y}},\theta)\|_2 \|S(w_{\mathbf{y}},\theta) - S(w_{\mathbf{y}}^K,\theta)\|_2 \\ &\leq 2(B+R)\|S(w_{\mathbf{y}},\theta) - S(w_{\mathbf{y}}^K,\theta)\|_2 \\ &\leq \tfrac{4(B+R)\zeta\sqrt{p}}{\lambda} \cdot \|w_{\mathbf{y}} - w_{\mathbf{y}}^K\|_2, \end{aligned} \tag{C.12}$$

where $\zeta = \max_{l\in[p],\mathbf{x}\in X(\theta),\theta\in\Theta} |f_l(\mathbf{x},\theta)|$. The third inequality is due to Cauchy Schwarz inequality. Under Assumptions 2.2.1 - 3.4.1, we can apply Lemma 3.4.6 to yield the last inequality.

Next, we will show that $\forall w \in \mathscr{W}_p$, the distance between w and its closest weight sample among $\{w_k\}_{k\in[K]}$ is upper bounded by the function of K and p and nothing else. More precisely, we will show that

$$\sup_{w\in\mathscr{W}_p} \min_{k\in[K]} \|w - w_k\|_2 \leq \frac{\sqrt{2}}{\Lambda - 1}. \tag{C.13}$$

Here, Λ is the number of evenly spaced weight samples between any two extreme points of $\mathscr{W}_p$.

Note that $\{w_k\}_{k\in[K]}$ are evenly sampled from $\mathscr{W}_p$, and that the distance between any two extreme points of $\mathscr{W}_p$ equals to $\sqrt{2}$. Hence, the distances between any two neighboring weight samples are equal and can be calculated as the distance between any two extreme points of $\mathscr{W}_p$ divided by $\Lambda - 1$. Proof of (C.13) can be done by further noticing that the distance between any w and $\{w_k\}_{k\in[K]}$ is upper bounded by the distances between any two neighboring weight samples.

Combining (C.12) and (C.13) yields that

$$0 \leq l_K(\mathbf{y},\theta) - l(\mathbf{y},\theta) \leq \frac{4(B+R)\zeta}{\lambda} \cdot \frac{\sqrt{2p}}{\Lambda - 1}, \tag{C.14}$$

Then, we can prove that the total number of weight samples K and Λ has the following relationship:

$$K = \binom{\Lambda + p - 2}{p - 1} \tag{C.15}$$

Proof of (C.15) can be done by induction with respect to p. Obviously, (C.15) holds when $p = 2$ as $K = \Lambda$. Assume (C.15) holds for the $\leq p - 1$ cases. For ease of notation, denote

$$K_p^\Lambda = \binom{\Lambda + p - 2}{p - 1}. \tag{C.16}$$

Then, for the p case, we note that the weight samples can be classified into two categories: $w_p = 0; w_p > 0$. For $w_p = 0$, the number of weight samples is simply K_{p-1}^{Λ}. For $w_p > 0$, the number of weight samples is $K_p^{\Lambda-1}$. Thus,

$$K = K_{p-1}^{\Lambda} + K_p^{\Lambda-1}. \tag{C.17}$$

Iteratively expanding $K_p^{\Lambda-1}$ through the same argument as (C.15) and using the fact that

$$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}, \tag{C.18}$$

we have

$$\begin{aligned} K &= K_{p-1}^{\Lambda} + K_p^{\Lambda-1} = K_{p-1}^{\Lambda} + K_{p-1}^{\Lambda-1} + K_p^{\Lambda-2} \\ &\vdots \\ &= K_{p-1}^{\Lambda} + K_{p-1}^{\Lambda-1} + \cdots + K_{p-1}^{2} + K_p^{1} \\ &= \binom{\Lambda+p-3}{p-2} + \binom{\Lambda+p-4}{p-2} + \cdots + \binom{p-1}{p-2} + \binom{p-1}{p-1} \\ &= \tfrac{(\Lambda+p-2)!}{(\Lambda-1)!(p-1)!} \end{aligned} \tag{C.19}$$

To this end, we complete the proof of (C.15).

Furthermore, we notice that

$$K = \frac{(\Lambda+p-2)!}{(\Lambda-1)!(p-1)!} \leq \frac{(\Lambda+p-2)^{p-1}}{(p-1)!} < \left(\frac{\Lambda+p-2}{p-1}\right)^{p-1} \cdot e^{p-1}. \tag{C.20}$$

Then, when $\Lambda \geq p (K \geq 2^{p-1})$, through simple algebraic calculation we have

$$\frac{e}{K^{\frac{1}{p-1}}} > \frac{p-1}{\Lambda+p-2} > \frac{1}{4} \cdot \frac{p}{\Lambda-1} \tag{C.21}$$

We complete the proof by combining (C.14) and (C.21) and noticing that $\sqrt{2p} \leq p$. $\square$

C.2.10 Proof of Proposition 3.4.8

Proof. Note that $\forall \theta \in \Theta$, $S(w, \theta)$ is single-valued due to the fact that $\mathbf{f}$ is strongly convex. $\forall \mathbf{y} \in \mathcal{Y}$, let $\mathbf{x}_\mathbf{y} \in X_P(\theta)$ be the nearest point to $\mathbf{y}$. By Proposition 3.2.2, there exists a $w_\mathbf{y} \in \mathscr{W}_p$ such that $\mathbf{x}_\mathbf{y} = S(w_\mathbf{y}, \theta)$. Let $w_\mathbf{y}^K$ be the nearest one to $w_\mathbf{y}$ among the weight samples $\{w_k\}_{k\in[K]}$. Then,

$$
\begin{aligned}
M_K(\theta) &= \mathbb{E}\bigg(l_K(\mathbf{y}, \theta) \bigg) \\
&\leq \mathbb{E}\bigg(\|\mathbf{y} - S(w_\mathbf{y}^K, \theta)\|_2^2 \bigg) \\
&= \mathbb{E}\bigg(\|\mathbf{y} - S(w_\mathbf{y}, \theta)\|_2^2 \bigg) + \mathbb{E}\bigg(\|S(w_\mathbf{y}, \theta) - S(w_\mathbf{y}^K, \theta)\|_2^2 \bigg) \\
&\quad +2\mathbb{E}\bigg(\big\langle \mathbf{y} - S(w_\mathbf{y}, \theta), S(w_\mathbf{y}, \theta) - S(w_\mathbf{y}^K, \theta) \big\rangle \bigg) \\
&\leq \mathbb{E}\bigg(\|\mathbf{y} - S(w_\mathbf{y}, \theta)\|_2^2 \bigg) + \mathbb{E}\bigg(\|S(w_\mathbf{y}, \theta) - S(w_\mathbf{y}^K, \theta)\|_2^2 \bigg) \\
&\quad +2\mathbb{E}\bigg(\|\mathbf{y} - S(w_\mathbf{y}, \theta)\|_2 \|S(w_\mathbf{y}, \theta) - S(w_\mathbf{y}^K, \theta)\|_2 \bigg) \quad \text{(Cauchy Schwarz inequality)} \\
&= M(\theta) + \mathbb{E}\bigg(\|S(w_\mathbf{y}, \theta) - S(w_\mathbf{y}^K, \theta)\|_2^2 \bigg) \\
&\quad +2\mathbb{E}\bigg(\|\mathbf{y} - S(w_\mathbf{y}, \theta)\|_2 \|S(w_\mathbf{y}, \theta) - S(w_\mathbf{y}^K, \theta)\|_2 \bigg),
\end{aligned}
\tag{C.22}
$$

where the first inequality is due to the fact that $l_K(\mathbf{y}, \theta) = \min_{k\in[K]}\{\|\mathbf{y} - \mathbf{x}_k\|_2^2 : \mathbf{x}_k = S(w_k, \theta)\} \leq \|\mathbf{y} - S(w_\mathbf{y}^K, \theta)\|_2^2$.

Let $A_K := \sup_{\mathbf{y}\in\mathcal{Y}, \theta\in\Theta} \|S(w_\mathbf{y}, \theta) - S(w_\mathbf{y}^K, \theta)\|_2$. Then,

$$
\mathbb{E}\bigg(\|S(w_\mathbf{y}, \theta) - S(w_\mathbf{y}^K, \theta)\|_2^2 \bigg) \leq A_K^2. \tag{C.23}
$$

Moreover,

$$
\begin{aligned}
\mathbb{E}\bigg(\|\mathbf{y} - S(w_\mathbf{y}, \theta)\|_2 \|S(w_\mathbf{y}, \theta) - S(w_\mathbf{y}^K, \theta)\|_2 \bigg) &\leq A_K \mathbb{E}\bigg(\|\mathbf{y} - S(w_\mathbf{y}, \theta)\|_2 \bigg) \\
&\leq A_K \mathbb{E}\bigg(\|\mathbf{y}\|_2 + \|S(w_\mathbf{y}, \theta)\|_2 \bigg) \\
&\leq A_K \mathbb{E}\bigg(\|\mathbf{y}\|_2 + B \bigg).
\end{aligned}
\tag{C.24}
$$

Note that $\mathbb{E}\Big(\|\mathbf{y}\|_2 + B\Big)$ in (C.24) is a finite number under our assumptions. Putting (C.23) and (C.24) into (C.22), and further noticing that $M_K(\theta) \geq M(\theta)$ by part (a) of Lemma 3.4.5, we have

$$0 \leq M_K(\theta) - M(\theta) \leq A_K\Big(A_K + 2B + 2\mathbb{E}\big(\|\mathbf{y}\|_2\big)\Big). \tag{C.25}$$

By (C.25), we will conclude the proof if we can show $A_K \longrightarrow 0$ in K. By Lemma 3.4.6,

$$A_K \leq \frac{2L}{\lambda} \sup_{\mathbf{y}\in\mathcal{Y}} \|w_{\mathbf{y}} - w_{\mathbf{y}}^K\|_2. \tag{C.26}$$

(C.26) implies that we only need to show $\|w_{\mathbf{y}} - w_{\mathbf{y}}^K\|_2^2 \longrightarrow 0$ in K for any $\mathbf{y} \in \mathcal{Y}$. It suffices to show that given any $w \in \mathscr{W}_p$, the nearest w_k to w among $\{w_k\}_{k\in[K]}$ can be arbitrarily small as $K \to \infty$. This is readily satisfied since we evenly sample $\{w_k\}_{k\in[K]}$ from $\mathscr{W}_p$. □

C.2.11 Proof of Proposition 3.4.9

Proof. We use notations here similar to those in Proposition 3.4.8. We have

$$\begin{aligned}
M_K^N(\theta) &= \tfrac{1}{N} \sum_{i\in[N]} \min_{k\in[K]} \|\mathbf{y}_i - \mathbf{x}_k\|_2^2 \\
&\leq \tfrac{1}{N} \sum_{i\in[N]} \|\mathbf{y}_i - S(w_{\mathbf{y}_i}^K, \theta)\|_2^2 \\
&= \tfrac{1}{N} \sum_{i\in[N]} \|\mathbf{y}_i - S(w_{\mathbf{y}_i}, \theta)\|_2^2 + \tfrac{1}{N} \sum_{i\in[N]} \|S(w_{\mathbf{y}_i}, \theta) - S(w_{\mathbf{y}_i}^K, \theta)\|_2^2 \\
&\quad + \tfrac{2}{N} \sum_{i\in[N]} \big\langle \mathbf{y}_i - S(w_{\mathbf{y}_i}, \theta), S(w_{\mathbf{y}_i}, \theta) - S(w_{\mathbf{y}_i}^K, \theta)\big\rangle \\
&\leq \tfrac{1}{N} \sum_{i\in[N]} \|\mathbf{y}_i - S(w_{\mathbf{y}_i}, \theta)\|_2^2 + \tfrac{1}{N} \sum_{i\in[N]} \|S(w_{\mathbf{y}_i}, \theta) - S(w_{\mathbf{y}_i}^K, \theta)\|_2^2 \\
&\quad + \tfrac{2}{N} \sum_{i\in[N]} \|\mathbf{y}_i - S(w_{\mathbf{y}_i}, \theta)\|_2 \|S(w_{\mathbf{y}_i}, \theta) - S(w_{\mathbf{y}_i}^K, \theta)\|_2.
\end{aligned} \tag{C.27}$$

Moreover, by part (b) of Lemma 3.4.5, we have $M_K^N(\theta) - M^N(\theta) \geq 0$. To this end, through a similar argument as in the proof of Proposition 3.4.8, we have

$$0 \leq M_K^N(\theta) - M^N(\theta) \leq A_K\Big(A_K + 2B + 2R\Big), \tag{C.28}$$

where the last inequality follows from the fact that $\max_{i\in[N],\theta\in\Theta} \|S(w_{\mathbf{y}_i}, \theta) - S(w_{\mathbf{y}_i}^K, \theta)\|_2 \leq A_K$. The remaining proof is exactly the same as that of Proposition 3.4.8. □

C.2.12 Proof of Proposition 3.4.10

Proof. $\forall\theta\in\Theta$, $|M_K^N(\theta)-M(\theta)| \xrightarrow{P} 0$ if and only if $\forall\delta>0,\forall\epsilon>0$, $\exists J$, s.t. $\forall N,K\geq J$,

$$\mathbb{P}(|M_K^N(\theta)-M(\theta)|>\epsilon)<\delta. \tag{C.29}$$

To prove the above statement, we first note that

$$\begin{aligned}\mathbb{P}(|M_K^N(\theta)-M(\theta)|>\epsilon) &= \mathbb{P}(|M_K^N(\theta)-M^N(\theta)+M^N(\theta)-M(\theta)|>\epsilon)\\ &\leq \mathbb{P}(|M_K^N(\theta)-M^N(\theta)|+|M^N(\theta)-M(\theta)|>\epsilon)\\ &\leq \mathbb{P}(|M_K^N(\theta)-M^N(\theta)|>\epsilon/2)+\mathbb{P}(|M^N(\theta)-M(\theta)|>\epsilon/2).\end{aligned} \tag{C.30}$$

For the first term on the last line of (C.30), by Proposition 3.4.9, $\exists K_1$, s.t. $\forall K\geq K_1$, $\forall N$,

$$\mathbb{P}(|M_K^N(\theta)-M^N(\theta)|>\epsilon/2)<\delta/2. \tag{C.31}$$

For the second term on the last line of (C.30), by Proposition 3.4.3, $\exists N_1$, s.t. $\forall N\geq N_1$,

$$\mathbb{P}(|M^N(\theta)-M(\theta)|>\epsilon/2)<\delta/2. \tag{C.32}$$

Now, let $J=\max\{N_1,K_1\}$. Putting (C.31) and (C.32) in (C.30), we have $\forall N,K\geq J$,

$$\mathbb{P}(|M_K^N(\theta)-M(\theta)|>\epsilon)<\delta. \tag{C.33}$$

Hence, we complete the proof. □

C.2.13 Proof of Theorem 3.4.11

Proof. Let $\theta^*\in\Theta^*$, and $\hat{\theta}^N\in\arg\min\{M^N(\theta):\theta\in\Theta\}$. Then, $M(\hat{\theta}^N)-M(\theta^*)\geq 0$. Also,

$$M(\hat{\theta}^N)-M(\theta^*) = M(\hat{\theta}^N)-M^N(\hat{\theta}^N)+M^N(\hat{\theta}^N)-M(\theta^*) \tag{C.34}$$

$$\leq M(\hat{\theta}^N)-M^N(\hat{\theta}^N)+M^N(\theta^*)-M(\theta^*) \tag{C.35}$$

$$\leq 2\sup_{\theta\in\Theta}|M^N(\theta)-M(\theta)|, \tag{C.36}$$

where the first inequality follows the fact that $M^N(\hat{\theta}^N)\leq M^N(\theta^*)$. Hence, applying Proposition 3.4.3 yields that $M(\hat{\theta}^N)-M(\theta^*)\xrightarrow{p}0$. □

C.2.14 Proof of Lemma 3.4.14

Proof. Let $\mathcal{G}$ be a class of functions g mapping from Z to $\mathbb{R}$, where

$$g(Z) = \frac{f(Z) - a}{b - a}. \tag{C.37}$$

Note that $g(Z) \in [0, 1]$. By Theorem 3.1 in [103], we have

$$\mathbb{E}[g(Z)] \leq \frac{1}{N} \sum_{i \in [N]} g(Z_i) + 2Rad_N(\mathcal{G}) + \sqrt{\frac{log(1/\delta)}{2N}}. \tag{C.38}$$

Using part 3 in Theorem 12 of [83], and the translation invariant property, i.e., $Rad_N(\mathcal{F}-a) = Rad_N(\mathcal{F})$, we have

$$Rad_N(\mathcal{G}) = Rad_N\left(\frac{\mathcal{F} - a}{b - a}\right) = \frac{Rad_N(\mathcal{F})}{b - a}. \tag{C.39}$$

Plugging (C.37) and (C.39) in (C.38) yields the main result. □

C.2.15 Proof of Lemma 3.4.15

Proof. By the definition of Rademacher complexity, we have

$$\begin{aligned} Rad_N\big(\mathcal{F}\big) &= \tfrac{1}{N}\mathbb{E}\Big[\sup_{f \in \mathcal{F}} \sum_{i \in [N]} \sigma_i f(\mathbf{y}_i, \theta)\Big] \\ &= \tfrac{1}{N}\mathbb{E}\Big[\sup_{\theta \in \Theta} \sum_{i \in [N]} \sigma_i \min_{k \in [K]} \|\mathbf{y}_i - \mathbf{x}_k\|_2^2\Big] \\ &= \tfrac{1}{N}\mathbb{E}\Big[\sup_{\theta \in \Theta} \sum_{i \in [N]} \sigma_i \min_{k \in [K]} \big(\|\mathbf{y}_i\|_2^2 - 2\langle \mathbf{y}_i, \mathbf{x}_k\rangle + \|\mathbf{x}_k\|_2^2\big)\Big] \\ &= \tfrac{1}{N}\mathbb{E}\Big[\sup_{\theta \in \Theta} \sum_{i \in [N]} \sigma_i \min_{k \in [K]} \big(-2\langle \mathbf{y}_i, \mathbf{x}_k\rangle + \|\mathbf{x}_k\|_2^2\big)\Big]. \end{aligned} \tag{C.40}$$

Note the fact $\mathbb{P}(\|\mathbf{x}\|_2 \leq B) = 1$ by Assumption 3.4.1. Through a similar argument in statement (ii) of Lemma 4.3 in [104], we get

$$\frac{1}{N}\mathbb{E}\Big[\sup_{\theta \in \Theta} \sum_{i \in [N]} \sigma_i \min_{k \in [K]} \big(-2\langle \mathbf{y}_i, \mathbf{x}_k\rangle + \|\mathbf{x}_k\|_2^2\big)\Big] \leq 2K\left(\frac{1}{N}\mathbb{E}\Big[\sup_{\|\mathbf{x}\|_2 \leq B} \sum_{i \in [N]} \sigma_i \langle \mathbf{y}_i, \mathbf{x}\rangle\Big] + \frac{B^2}{2\sqrt{N}}\right). \tag{C.41}$$

The first term on the right-hand side of (C.41) can be upper bounded in the following way:

$$\begin{aligned}
\frac{1}{N}\mathbb{E}\Big[\sup_{\|\mathbf{x}\|_2\leq B}\sum_{i\in[N]}\sigma_i\langle\mathbf{y}_i,\mathbf{x}\rangle\Big] &= \frac{1}{N}\mathbb{E}\Big[\sup_{\|\mathbf{x}\|_2\leq B}\langle\sum_{i\in[N]}\sigma_i\mathbf{y}_i,\mathbf{x}\rangle\Big] \\
&\leq \frac{1}{N}\mathbb{E}\sup_{\|\mathbf{x}\|_2\leq B}\|\mathbf{x}\|_2\|\sum_{i\in[N]}\sigma_i\mathbf{y}_i\|_2 \qquad \text{(Cauchy-Schwarz inequality)} \\
&\leq \frac{B}{N}\mathbb{E}\|\sum_{i\in[N]}\sigma_i\mathbf{y}_i\|_2 \\
&\leq \frac{B}{N}\sqrt{\mathbb{E}\|\sum_{i\in[N]}\sigma_i\mathbf{y}_i\|_2^2} \qquad \text{(Jensen's inequality)} \\
&= \frac{B}{N}\sqrt{N\mathbb{E}\|\mathbf{y}\|_2^2} \\
&\leq \frac{BR}{\sqrt{N}} \qquad (\mathbb{P}(\|\mathbf{y}\|_2\leq R)=1).
\end{aligned} \tag{C.42}$$

Plugging the result of (C.42) in (C.41), we get the bound for the Rademacher complexity of $\mathcal{F}$. □

C.2.16 Proof of Theorem 3.4.16

Proof. We specialize Lemmas 3.4.14 and 3.4.15 to prove the theorem. Note that

$$0\leq f(\mathbf{y},\theta)=\min_{k\in[K]}\|\mathbf{y}-\mathbf{x}_k\|_2^2\leq(B+R)^2. \tag{C.43}$$

Let $a=0, b=(B+R)^2$ in Lemma 3.4.14. Then, combining the results in Lemmas 3.4.14 and 3.4.15 yields this theorem. □

C.2.17 Proof of Lemma 3.4.17

Proof. Sufficiency: $d_{sH}(X,Y)=0$ implies that $\inf_{y\in Y}\|x-y\|_2=0, \forall x\in X$. That is, $\exists y\in Y$, st. $x=y$. Hence, $X\subseteq Y$. Necessity: $X\subseteq Y$ implies that $\forall x\in X$, $\exists y\in Y$, s.t. $y=x$. Thus, $\inf_{y\in Y}\|x-y\|_2=0$. Therefore, $d_{sH}(X,Y)=0$. □

C.2.18 Proof of Theorem 3.4.18

Proof. First, we show that θ_0 minimizes $M(\theta)$ among Θ. This is readily true since $M(\theta_0) = 0$ by noting that there is no noise in the data. By Theorem 3.4.13, a direct result is $M(\hat{\theta}_K^N) \xrightarrow{P} M(\theta_0) = 0$. Second, we show that θ_0 is the unique solution that minimizes $M(\theta)$ among Θ. $\forall \theta' \in \Theta \setminus \theta$, $M(\theta) = \mathbb{E}_{\mathbf{y} \in X_P(\theta_0)}\big(\min_{\mathbf{x} \in X_P(\theta)} \|\mathbf{y} - \mathbf{x}\|_2^2\big) > 0$ as $d_{sH}(X_P(\theta), X_P(\theta')) > 0$. Consequently, we have $M(\theta) > M(\theta_0) = 0$. Finally, since 3.1 is identifiable at θ_0, then $\forall \epsilon > 0$, $\exists \delta > 0$, s.t. $M(\theta) - M(\theta_0) > \delta$ for every θ with $d(\theta, \theta_0) > \epsilon$. Thus, the event $\{d(\hat{\theta}_K^N, \theta_0) > \epsilon\}$ is contained in the event $\{M(\hat{\theta}_K^N) - M(\theta_0) > \delta\}$. Namely, $\mathbb{P}(d(\hat{\theta}_K^N, \theta_0) > \epsilon) \leq \mathbb{P}(M(\hat{\theta}_K^N) - M(\theta_0) > \delta)$. We complete the proof by noting that the probability of the right term converges to 0 as $M(\hat{\theta}_K^N) \xrightarrow{P} M(\theta_0)$. □

C.2.19 Proof of Theorem 3.4.19

Proof. First, note that

$$\begin{aligned} \|S(w_{\mathbf{y}}^{NK}, \theta_0) - S(w_{\mathbf{y}}, \theta_0)\|_2 &= \|S(w_{\mathbf{y}}^{NK}, \theta_0) - S(w_{\mathbf{y}}^{NK}, \hat{\theta}_K^N) \\ &\quad + S(w_{\mathbf{y}}^{NK}, \hat{\theta}_K^N) - S(w_{\mathbf{y}}, \theta_0)\|_2 \\ &\leq \|S(w_{\mathbf{y}}^{NK}, \theta_0) - S(w_{\mathbf{y}}^{NK}, \hat{\theta}_K^N)\|_2 \\ &\quad + \|S(w_{\mathbf{y}}^{NK}, \hat{\theta}_K^N) - S(w_{\mathbf{y}}, \theta_0)\|_2. \end{aligned} \tag{C.44}$$

By Theorem 3.4.18, we have $\hat{\theta}_K^N \xrightarrow{P} \theta_0$. Note that $S(w, \theta)$ is continuous in $\theta \in \Theta$. By continuous mapping theorem, the first term in the last line of (C.44) $\|S(w_{\mathbf{y}}^{NK}, \theta_0) - S(w_{\mathbf{y}}^{NK}, \hat{\theta}_K^N)\|_2 \xrightarrow{P} 0$.

By the argument in the proof of Theorem 3.4.18, the second term in the last line of (C.44) $\|S(w_{\mathbf{y}}^{NK}, \hat{\theta}_K^N) - S(w_{\mathbf{y}}, \theta_0)\|_2 \xrightarrow{P} 0$ almost surely. Otherwise,

$$M(\hat{\theta}_K^N) = \mathbb{E}_{\mathbf{y} \in X_P(\theta_0)}\big(\min_{\mathbf{x} \in X_P(\hat{\theta}_K^N)} \|\mathbf{y} - \mathbf{x}\|_2^2\big) = \mathbb{E}_{\mathbf{y} \in X_P(\theta_0)} \|S(w_{\mathbf{y}}^{NK}, \hat{\theta}_K^N) - S(w_{\mathbf{y}}, \theta_0)\|_2^2 > 0,$$

and thus will not converge to $M(\theta_0)$.

Putting the above two results into (C.44) yields $\|S(w_{\mathbf{y}}^{NK}, \theta_0) - S(w_{\mathbf{y}}, \theta_0)\|_2 \xrightarrow{P} 0$ almost surely.

Next, note that $S(w, \theta_0)$ is continuous in w, and that $MOP(\theta_0)$ is bijective. Then, we have that $S(\cdot, \theta_0) : \mathscr{W}_p \to X_P(\theta_0)$ is a one-to-one correspondence. Thus, $S(\cdot, \theta_0)$ is a homeomorphism by the inverse mapping theorem [105], meaning that the inverse map $S^{-1}(\cdot, \theta_0) : X_P(\theta_0) \to \mathscr{W}_p$ is also continuous. Therefore, $\|S(w_{\mathbf{y}}^{NK}, \theta_0) - S(w_{\mathbf{y}}, \theta_0)\|_2 \xrightarrow{P} 0$ implies that $\|w_{\mathbf{y}} - w_{\mathbf{y}}^{NK}\|_2 \xrightarrow{P} 0$ by the continuous mapping theorem. □

C.2.20 Proof of Proposition 3.4.20

Proof. Since both examples are strongly convex MOPs, any Pareto optimal solution of them can be obtained by solving 3.2 according to Proposition 3.2.2. Also, every optimal solution of the weighting problem is a Pareto optimal solution by part (b) of Proposition 3.2.1.

Let $w \in [0, 1]$ be the weight of the first function. The optimal solutions for 3.2 in Example 3.4.2 can be characterized parametrically by w as

$$x_1^1(w) = \begin{cases} \frac{6-9w}{2-w}, & \text{if } 0 \le w \le 2/3, \\ 0, & \text{if } 2/3 < w \le 1, \end{cases} \quad x_2^1(w) = \begin{cases} 3, & \text{if } 0 \le w \le 2/9, \\ \frac{5-6w}{1+w}, & \text{if } 2/9 < w \le 5/6, \\ 0, & \text{otherwise.} \end{cases} \tag{C.45}$$

Similarly, the optimal solutions for the 3.2 in Example 3.4.3 can be characterized parametrically as

$$x_1^2(w) = \begin{cases} \frac{36-45w}{12-5w}, & \text{if } 0 \le w \le 4/5, \\ 0, & \text{otherwise,} \end{cases} \quad x_2^2(w) = \begin{cases} 3, & \text{if } 0 \le w \le 4/15, \\ \frac{30-30w}{6+5w}, & \text{otherwise.} \end{cases} \tag{C.46}$$

We can show that $x_1^1(w) = x_1^2(\frac{6}{5}w)$ and $x_2^1(w) = x_2^2(\frac{6}{5}w)$ for $0 \le w \le \frac{5}{6}$. In addition, $x_1^1(w) = x_2^1(w) = 0$ for $\frac{5}{6} \le w \le 1$. Therefore, these parametric points in (C.45) and (C.46) correspond to the same curve. Hence, EXAMPLE 3.4.2 and EXAMPLE 3.4.3 have the same Pareto optimal set. □

C.2.21 Proof of Lemma 3.5.1

Proof. First, $M_K^N(\theta)$ decreases in the **Assignment step** since each $\mathbf{y}_i$ is assigned to the closest $\mathbf{x}_k$. So the distance $\mathbf{y}_i$ contributes to $M_K^N(\theta)$ decreases. Second, $M_K^N(\theta)$ decreases in the **Update step** because the new θ and $\{\mathbf{x}_k\}_{k\in[K]}$ are the ones for which $M_K^N(\theta)$ attains its minimum. □

C.3 Omitted Algorithms

C.3.1 ADMM for IMOP

The ADMM was originally proposed in [106] and [107], and recently revisited by [108]. In practice, ADMM often exhibits a substantially faster convergence rate than traditional methods in solving convex optimization problems. Characterizing the convergence rate of ADMM for convex optimization problems is still a popular research topic [109, 110, 111]. Although ADMM might not converge even for convex problems with more than two blocks of variables [112], many recent papers have numerically demonstrated the fast and appealing convergence behavior of ADMM on nonconvex problems [113, 114, 115]. Hence, we apply ADMM as a heuristic to solve the nonconvex problem 3.9.

3.9 is closely related to the global consensus problem discussed heavily in [108], but with the important difference that 3.9 is a nonconvex problem. In order to use ADMM, we first partition $\{\mathbf{y}_i\}_{i\in[N]}$ equally into T groups, and denote $\{\mathbf{y}_i\}_{i\in[N_t]}$ the observations in t-th group. Then, we introduce a set of new variables $\{\theta^t\}_{t\in T}$, typically called local variables, and transform 3.9 equivalently to the following problem:

$$\begin{aligned} \min_{\theta\in\Theta,\theta^t\in\Theta} \quad & \sum_{t\in T}\sum_{i\in[N_t]} l_K(\mathbf{y}_i,\theta^t) \\ \text{s.t.} \quad & \theta^t=\theta, \qquad\qquad \forall t\in[T]. \end{aligned} \tag{C.47}$$

ADMM for problem (C.47) can be derived directly from the augmented Lagrangian

$$L_\rho(\theta,\{\theta^t\}_{t\in[T]},\{\mathbf{v}^t\}_{t\in[T]})=\sum_{t\in[T]}\left(\sum_{i\in[N_t]} l_K(\mathbf{y}_i,\theta^t)+<\mathbf{v}^t,\theta^t-\theta>+(\rho/2)\|\theta^t-\theta\|_2^2\right), \tag{C.48}$$

where $\rho > 0$ is an algorithm parameter, $\mathbf{v}^t$ is the dual variable for the constraint $\theta^t = \theta$.

Let $\overline{\theta}^k = \frac{1}{|T|}\sum_{t\in T}\theta^{t,k}$. As suggested in [108], the primal and dual residuals are

$$r^k_{pri} = \left(\theta^{1,k} - \overline{\theta}^k, \ldots, \theta^{|T|,k} - \overline{\theta}^k\right), \quad r^k_{dual} = -\rho\left(\overline{\theta}^k - \overline{\theta}^{k-1}, \ldots, \overline{\theta}^k - \overline{\theta}^{k-1}\right), \tag{C.49}$$

so their squared norms are

$$\|r^k_{pri}\|_2^2 = \sum_{t\in T}\|\theta^{t,k} - \overline{\theta}^k\|_2^2, \;\; \|r^k_{dual}\|_2^2 = |T|\rho^2\|\overline{\theta}^k - \overline{\theta}^{k-1}\|_2^2. \tag{C.50}$$

$\|r^k_{pri}\|_2^2$ is $|T|$ times the variance of $\{\theta^{t,k}\}_{t\in T}$, which can be interpreted as a natural measure of (lack of) consensus. Similarly, $\|r^k_{dual}\|_2^2$ is a measure of the step length. These suggest that a reasonable stopping criterion is that the primal and dual residuals must be small.

The resulting ADMM algorithm in scaled form is formally presented in the following.

Algorithm 6 ADMM for 3.9

1: **Input:** Noisy decisions $\{\mathbf{y}_i\}_{i\in[N]}$, weight samples $\{w_k\}_{k\in[K]}$.
2: Set $k = 0$ and initialize θ^0 and $\mathbf{v}^{t,0}$ for each $t \in T$.
3: **while** stopping criterion is not satisfied **do**
4: **for** $t \in [T]$ **do**
5: $\theta^{t,k+1} \leftarrow \arg\min_{\theta^t}\left\{\sum_{i\in N_t} l_K(\mathbf{y}_i, \theta^t) + (\rho/2)\|\theta^t - \theta^k + \mathbf{v}^{t,k}\|_2^2\right\}$.
6: **end for**
7: $\theta^{k+1} \leftarrow \frac{1}{|T|}\sum_{t\in T}\left(\theta^{t,k+1} + \mathbf{v}^{t,k}\right)$.
8: **for** $t \in [T]$ **do**
9: $\mathbf{v}^{t,k+1} \leftarrow \mathbf{v}^{t,k} + \theta^{t,k+1} - \theta^{k+1}$.
10: **end for**
11: $k \leftarrow k + 1$.
12: **end while**

Remark C.3.1. (*i*) With a slight abuse of notation, we use θ^k to denote the estimation of θ in the k-th iteration, and θ^t to denote the local variable for the observations in t-th group. (*ii*) The stopping criterion could be that $\|r^k_{pri}\|_2 < \epsilon^{pri}$ and $\|r^k_{dual}\|_2 < \epsilon^{dual}$, or the maximum iteration number is reached. (*iii*) Note that $L_\rho(\theta, \{\theta^t\}_{t\in[T]}, \{\mathbf{v}^t\}_{t\in[T]})$ is separable in θ^t. Hence, the θ^t-update step splits into $|T|$ independent problems that can

be implemented in parallel. We show in experiments parallel computing would dramatically improve the computational efficiency. For the same reason, the dual variables $\mathbf{v}^t$-update step can be carried out in parallel for each $t \in [T]$.

Remark C.3.2. For the initialization of θ^0 in Algorithm 6, we can incorporate the idea in [1] that imputes a convex objective function by minimizing the residuals of KKT conditions incurred by noisy data. This leads to the following initialization problem:

$$
\begin{aligned}
\min_{\theta\in\Theta} \quad & \tfrac{1}{N}\sum_{i\in[N]} \left(r^i_{comp} + r^i_{stat}\right) \\
\text{s.t.} \quad & \mathbf{u}_i \geq \mathbf{0}_m, && \forall i \in [N], \\
& |\mathbf{u}_i^T \mathbf{g}(\mathbf{y}_i, \theta)| \leq r^i_{comp}, && \forall i \in [N], \\
& \bigvee_{k\in[K]} \left[\ \|\nabla w_k^T \mathbf{f}(\mathbf{y}_i, \theta) + \mathbf{u}_i^T \nabla \mathbf{g}(\mathbf{y}_i, \theta)\|_2 \leq r^i_{stat} \ \right], && \forall i \in [N], \\
& \mathbf{u}_i \in \mathbb{R}^m_+, \ \ r^i_{comp} \in \mathbb{R}_+, \ \ r^i_{stat} \in \mathbb{R}_+, && \forall i \in [N],
\end{aligned}
\tag{C.51}
$$

where r^i_{comp} and r^i_{stat} are residuals corresponding to the complementary slackness and stationarity in KKT conditions for the i-th noisy decision $\mathbf{y}_i$. The disjunction constraints are imposed to assign one of the weight samples to $\mathbf{y}_i$. Similarly, we can integrate the approach of minimizing the slackness needed to render observations to (approximately) satisfy variational inequalities [13] into our model, to provide an initialization of θ^0.

C.4 Omitted Mathematical Formulations

C.4.1 Reformulation of 3.30 Using KKT Conditions

$$
\begin{aligned}
\max_{\theta\in\Theta} \quad & \|\theta-\hat{\theta}_K^N\|_1 \\
s.t. \quad & \mathbf{u}_i \geq \mathbf{0}, && \forall i\in[N'], \\
& \mathbf{u}_i^T\mathbf{g}(\mathbf{x}_i,\theta)=0, && \forall i\in[N'], \\
& \|\nabla_{\mathbf{x}_i} w_k^T\mathbf{f}(\mathbf{x}_i,\theta)+\mathbf{u}_i^T\nabla_{\mathbf{x}_i}\mathbf{g}(\mathbf{x}_i,\theta)\|_2 \leq M(1-z_{ik}), && \forall i\in[N'], k\in[K'], \\
& \sum_{k\in[K']} z_{ik}=1, && \forall i\in[N'], \\
& z_{ik}\in\{0,1\},\ \ \mathbf{u}_i\in\mathbb{R}_+^q, && \forall i\in[N'], k\in[K'].
\end{aligned}
\tag{C.52}
$$

C.4.2 Single Level Reformulation for Inferring Objective Functions of MLP

$$
\begin{aligned}
\min_{\mathbf{c}_1,\cdots,\mathbf{c}_p} \quad & \sum_{i\in[N]}\|\mathbf{y}_i-\sum_{k\in[K]}\eta_{ik}\|_2 \\
s.t. \quad & \mathbf{c}_l\in C_l, && \forall l\in[p], \\
& \left[\begin{array}{l}
\mathbf{A}\mathbf{x}_k\geq\mathbf{b},\ \mathbf{x}_k\geq\mathbf{0}, \\
\mathbf{A}^T\mathbf{u}_k\leq w_k^1\mathbf{c}_1+\cdots+w_k^p\mathbf{c}_p,\ \mathbf{u}_k\geq 0, \\
\mathbf{x}_k\leq M_1\mathbf{t}_{1k}, \\
w_k^1\mathbf{c}_1+\cdots+w_k^p\mathbf{c}_p-\mathbf{A}^T\mathbf{u}_k\leq M_1(1-\mathbf{t}_{1k}), \\
\mathbf{u}_k\leq M_2\mathbf{t}_{2k}, \\
\mathbf{A}\mathbf{x}_k-\mathbf{b}\leq M_2(1-\mathbf{t}_{2k})
\end{array}\right], && \forall k\in[K], \\
& 0\leq\eta_{ik}\leq M_{ik}z_{ik}, \\
& \mathbf{x}_k-M_{ik}(1-z_{ik})\leq\eta_{ik}\leq\mathbf{x}_k, \\
& \sum_{k\in[K]} z_{ik}=1, && \forall i\in[N], \\
& \mathbf{x}_k\in\mathbb{R}_+^n,\ \ \mathbf{u}_k\in\mathbb{R}_+^m,\ \ \mathbf{t}_{1k}\in\{0,1\}^n,\ \ \mathbf{t}_{2k}\in\{0,1\}^m,\ \ z_{ik}\in\{0,1\},
\end{aligned}
\tag{C.53}
$$

where C_l is a convex compact set for each $l \in [p]$. M_1, M_2 and M_{ik} are Big-Ms used to linearize the program. One can establish similar reformulations for inferring RHS of MLP.

C.4.3 Single Level Reformulation for Inferring RHS of MQP

$$
\begin{aligned}
\min_{\mathbf{b}} \quad & \sum_{i\in[N]} \|\mathbf{y}_i - \sum_{k\in[K]} \eta_{ik}\|_2 \\
\text{s.t.} \quad & \mathbf{b} \in B, \\
& \left[\begin{array}{l} \mathbf{A}\mathbf{x}_k \geq \mathbf{b},\ \mathbf{u}_k \geq \mathbf{0}, \\ \mathbf{u}_k \leq M_1 \mathbf{t}_k, \\ \mathbf{A}\mathbf{x}_k - \mathbf{b} \leq M_1(1-\mathbf{t}_k), \\ (w_k^1 Q_1 + \cdots + w_k^p Q_p)\mathbf{x}_i + w_k^1 \mathbf{c}_1 + \cdots + w_k^p \mathbf{c}_p - \mathbf{A}^T \mathbf{u}_k = 0, \end{array}\right], \quad \forall k \in [K], \\
& 0 \leq \eta_{ik} \leq M_{ik} z_{ik}, \\
& \mathbf{x}_k - M_{ik}(1-z_{ik}) \leq \eta_{ik} \leq \mathbf{x}_k + M_{ik}(1-z_{ik}), \\
& \sum_{k\in[K]} z_{ik} = 1, \qquad \forall i \in [N], \\
& \mathbf{b} \in \mathbb{R}^m,\ \ \mathbf{x}_k \in \mathbb{R}^n,\ \ \mathbf{u}_k \in \mathbb{R}^m_+,\ \ \mathbf{t}_k \in \{0,1\}^m,\ \ z_{ik} \in \{0,1\},
\end{aligned}
\tag{C.54}
$$

where B is a convex compact set. M_1 and M_{ik} are Big-Ms used to linearize the program. One can establish similar reformulations for inferring objectives of MQP.

C.5 Detailed Experiment Results

C.5.1 Learning the RHS of an MQP

Table 13: Estimation Error $\|\hat{\mathbf{b}} - \mathbf{b}_{true}\|_2$ for Different N and K

	$N = 5$	$N = 10$	$N = 20$	$N = 50$	$N = 100$	$N = 150$
$K = 6$	1.496	1.063	0.861	0.601	0.531	0.506
$K = 11$	1.410	0.956	0.524	0.378	0.217	0.199
$K = 21$	1.382	0.925	0.498	0.313	0.138	0.117
$K = 41$	1.380	0.924	0.484	0.295	0.127	0.111

C.5.2 Learning the Objective functions of an MQP

Table 14: Prediction Error $M(\hat{\theta}_K^N)$ for Different N and K

	$N = 50$	$N = 100$	$N = 250$	$N = 500$	$N = 1000$	$N = 5000$	$N = 10000$	$N = 50000$
$K = 6$	0.050	0.043	0.039	0.040	0.039	0.040	0.038	0.038
$K = 11$	0.030	0.028	0.028	0.027	0.027	0.026	0.026	0.025
$K = 21$	0.026	0.025	0.024	0.024	0.024	0.024	0.024	0.024
$K = 41$	0.025	0.024	0.024	0.023	0.023	0.023	0.023	0.023

Table 15: True Expected Return

Security	1	2	3	4	5	6	7	8
Expected Return	0.1791	0.1143	0.1357	0.0837	0.1653	0.1808	0.0352	0.0368

Table 16: True Return Covariances Matrix

Security	1	2	3	4	5	6	7	8
1	0.1641	0.0299	0.0478	0.0491	0.058	0.0871	0.0603	0.0492
2	0.0299	0.0720	0.0511	0.0287	0.0527	0.0297	0.0291	0.0326
3	0.0478	0.0511	0.0794	0.0498	0.0664	0.0479	0.0395	0.0523
4	0.0491	0.0287	0.0498	0.1148	0.0336	0.0503	0.0326	0.0447
5	0.0580	0.0527	0.0664	0.0336	0.1073	0.0483	0.0402	0.0533
6	0.0871	0.0297	0.0479	0.0503	0.0483	0.1134	0.0591	0.0387
7	0.0603	0.0291	0.0395	0.0326	0.0402	0.0591	0.0704	0.0244
8	0.0492	0.0326	0.0523	0.0447	0.0533	0.0387	0.0244	0.1028

Appendix D

D.1 Omitted Proofs

D.1.1 Example 4.3.1

We will show next that $\kappa = 2R$. Let $\theta_1 = (Q_1^1, Q_2^1, \mathbf{c}_1^1, \mathbf{c}_2^1), \theta_2 = (Q_1^2, Q_2^2, \mathbf{c}_1^1, \mathbf{c}_2^2)$. Then,

$$\begin{aligned} h(\mathbf{x}, w, \theta_1, \theta_2) &= w^T \mathbf{f}(\mathbf{x}, \theta_1) - w^T \mathbf{f}(\mathbf{x}, \theta_2) \\ &= \tfrac{1}{2} w_1 \mathbf{x}^T Q_1^1 \mathbf{x} + w_1 \mathbf{x}^T \mathbf{c}_1^1 + \tfrac{1}{2} w_2 \mathbf{x}^T Q_2^1 \mathbf{x} + w_2 \mathbf{x}^T \mathbf{c}_2^1 \\ &\quad -\tfrac{1}{2} w_1 \mathbf{x}^T Q_1^2 \mathbf{x} - w_1 \mathbf{x}^T \mathbf{c}_1^2 - \tfrac{1}{2} w_2 \mathbf{x}^T Q_2^2 \mathbf{x} - w_2 \mathbf{x}^T \mathbf{c}_2^2. \end{aligned}$$

Since $h(\mathbf{x}, w, \theta_1, \theta_2)$ is continuously differentiable in $\mathbf{x}$, the Lipschitz constant can be estimated by bounding the norm of the gradient of h. We have

$$\frac{\partial h}{\partial \mathbf{x}} = w_1(Q_1^1 - Q_1^2)\mathbf{x} + w_1(\mathbf{c}_1^1 - \mathbf{c}_1^2) + w_2(Q_2^1 - Q_2^2)\mathbf{x} + w_2(\mathbf{c}_2^1 - \mathbf{c}_2^2).$$

Thus,

$$\begin{aligned} \sup_{\|\mathbf{x}\|_2 \le R} \|\tfrac{\partial h}{\partial \mathbf{x}}\|_2 &= \sup_{\|\mathbf{x}\|_2 \le R} \|w_1(Q_1^1 - Q_1^2)\mathbf{x} + w_1(\mathbf{c}_1^1 - \mathbf{c}_1^2) + w_2(Q_2^1 - Q_2^2)\mathbf{x} + w_2(\mathbf{c}_2^1 - \mathbf{c}_2^2)\|_2 \\ &\le \sup_{\|\mathbf{x}\|_2 \le R} \|w_1(Q_1^1 - Q_1^2)\mathbf{x}\|_2 + \|w_1(\mathbf{c}_1^1 - \mathbf{c}_1^2)\|_2 \\ &\quad + \sup_{\|\mathbf{x}\|_2 \le R} \|w_2(Q_2^1 - Q_2^2)\mathbf{x}\|_2 + \|w_2(\mathbf{c}_2^1 - \mathbf{c}_2^2)\|_2 \\ &\le w_1 \|Q_1^1 - Q_1^2\|_F \cdot \sup_{\|\mathbf{x}\|_2 \le R} \|\mathbf{x}\|_2 + w_1 \|\mathbf{c}_1^1 - \mathbf{c}_1^2\|_2 \\ &\quad + w_2 \|Q_2^1 - Q_2^2\|_F \cdot \sup_{\|\mathbf{x}\|_2 \le R} \|\mathbf{x}\|_2 + w_2 \|\mathbf{c}_2^1 - \mathbf{c}_2^2\|_2 \\ &\le R\|Q_1^1 - Q_1^2\|_F + \|\mathbf{c}_1^1 - \mathbf{c}_1^2\|_2 + R\|Q_2^1 - Q_2^2\|_F + \|\mathbf{c}_2^1 - \mathbf{c}_2^2\|_2 \\ &\le 2R\sqrt{\|Q_1^1 - Q_1^2\|_F^2 + \|\mathbf{c}_1^1 - \mathbf{c}_1^2\|_2^2 + \|Q_2^1 - Q_2^2\|_F^2 + \|\mathbf{c}_2^1 - \mathbf{c}_2^2\|_2^2} \\ &= 2R\|\theta_1 - \theta_2\|_2. \end{aligned}$$

where the last inequality follows from the Power mean inequality. Hence, $h(\cdot, w, \theta_1, \theta_2)$ is $2R\|\theta_1 - \theta_2\|_2$-Lipschitz continuous on $\mathcal{Y}$.

D.1.2 Proof of Lemma 4.3.1

Proof. (a) Proof of (a) is straightforward. $l_K(\mathbf{y},\theta) = \min_{\mathbf{x}\in \bigcup_{k\in[K]} S(w_k,\theta)} \|\mathbf{y}-\mathbf{x}\|_2^2 \leq (\|\mathbf{y}\|_2+B)^2 \leq (R+B)^2$.

(b) $\forall\theta\in\Theta$, $\forall\mathbf{y}_1,\mathbf{y}_2\in\mathcal{Y}$, let

$$l_K(\mathbf{y}_1,\theta) = \|\mathbf{y}_1 - S(w^1,\theta)\|_2^2,\ l_K(\mathbf{y}_2,\theta) = \|\mathbf{y}_2 - S(w^2,\theta)\|_2^2.$$

Without of loss of generality, let $l_K(\mathbf{y}_1,\theta) \geq l_K(\mathbf{y}_2,\theta)$. Then,

$$\begin{aligned}
&|l_K(\mathbf{y}_1,\theta) - l_K(\mathbf{y}_2,\theta)| = l_K(\mathbf{y}_1,\theta) - l_K(\mathbf{y}_2,\theta)\\
&= \|\mathbf{y}_1 - S(w^1,\theta)\|_2^2 - \|\mathbf{y}_2 - S(w^2,\theta)\|_2^2\\
&\leq \|\mathbf{y}_1 - S(w^2,\theta)\|_2^2 - \|\mathbf{y}_2 - S(w^2,\theta)\|_2^2\\
&= \langle \mathbf{y}_1-\mathbf{y}_2, \mathbf{y}_1+\mathbf{y}_2-2S(w^2,\theta)\rangle\\
&\leq 2(B+R)\|\mathbf{y}_1-\mathbf{y}_2\|_2
\end{aligned} \tag{D.1}$$

The last inequality is due to Cauchy-Schwartz inequality and the Assumptions 3.1(a), that is

$$\|\mathbf{y}_1+\mathbf{y}_2-2S(w^2,\theta)\|_2 \leq 2(B+R) \tag{D.2}$$

Plugging (D.2)in (D.1) yields the claim.

(c) $\forall\mathbf{y}\in\mathcal{Y}$, $\forall\theta_1,\theta_2\in\Theta$, let

$$l_K(\mathbf{y},\theta_1) = \|\mathbf{y} - S(w^1,\theta_1)\|_2^2,\ l_K(\mathbf{y},\theta_2) = \|\mathbf{y}_2 - S(w^2,\theta_2)\|_2^2.$$

Without of loss of generality, let $l_K(\mathbf{y},\theta_1) \geq l_K(\mathbf{y},\theta_2)$. Then,

$$\begin{aligned}
&|l_K(\mathbf{y},\theta_1) - l_K(\mathbf{y},\theta_2)| = l_K(\mathbf{y},\theta_1) - l_K(\mathbf{y},\theta_2)\\
&= \|\mathbf{y} - S(w^1,\theta_1)\|_2^2 - \|\mathbf{y} - S(w^2,\theta_2)\|_2^2\\
&\leq \|\mathbf{y} - S(w^2,\theta_1)\|_2^2 - \|\mathbf{y} - S(w^2,\theta_2)\|_2^2\\
&= \langle S(w^2,\theta_2) - S(w^2,\theta_1), 2\mathbf{y} - S(w^2,\theta_1) - S(w^2,\theta_2)\rangle\\
&\leq 2(B+R)\|S(w^2,\theta_2) - S(w^2,\theta_1)\|_2
\end{aligned} \tag{D.3}$$

The last inequality is due to Cauchy-Schwartz inequality and the Assumptions 3.1(a), that is

$$\|2\mathbf{y} - S(w^2,\theta_1) - S(w^2,\theta_2)\|_2 \leq 2(B+R) \tag{D.4}$$

Next, we will apply Proposition 6.1 in [57] to bound $\|S(w^2,\theta_2) - S(w^2,\theta_1)\|_2$.

Under Assumptions 3.1 - 3.2, the conditions of Proposition 6.1 in [57] are satisfied. Therefore,

$$\|S(w^2, \theta_2) - S(w^2, \theta_1)\|_2 \leq \frac{2\kappa}{\lambda}\|\theta_1 - \theta_2\|_2 \tag{D.5}$$

Plugging (D.4) and (D.5) in (D.3) yields the claim.

□

D.1.3 Proof of Theorem 4.3.2

Proof. Under Assumption 4.2.1, we know that Θ is compact. Similarly, $\mathcal{Y}$ is also compact under Assumption 4.2.2 (a). By lemma 4.3.1 (a), $\forall \mathbf{y} \in \mathcal{Y}, \theta \in \Theta, 0 \leq l_K(\mathbf{y}, \theta) \leq (B + R)^2$, and thus $l_K(\mathbf{y}, \theta)$ is bounded. In addition, by lemma 4.3.1 (b), $l_K(\mathbf{y}, \theta)$ is continuous in θ for any $\mathbf{y} \in \mathcal{Y}$. Finally, by lemma 4.3.1 (c), $l_K(\mathbf{y}, \theta)$ is uniformly $\frac{4(B+R)\kappa}{\lambda}$-Lipschitz continuous in $\mathbf{y}$. Hence, applying Corollary 3.8 of [29] yields the result. □

D.1.4 Proof of Theorem 4.3.3

Proof. For ease of notation, we denote $(\widehat{\theta}^s, \widehat{\mathbf{v}}^s)$ the solution found in Step 4 in the sth iteration of Algorithm 1.

Suppose that for $s = 1, \ldots, S$ the algorithm has not terminated, i.e. $\max_{i \in [N]} l_K(\widetilde{\mathbf{y}}_i, \widehat{\theta}) - \widehat{v}_{N+1} \cdot \|\widetilde{\mathbf{y}}_i - \mathbf{y}_i\|_2 - \widehat{v}_i > \delta$.

Let $i^* = \arg\max_{i \in [N]} l_K(\widetilde{\mathbf{y}}_i, \widehat{\theta}) - \widehat{v}_{N+1} \cdot \|\widetilde{\mathbf{y}}_i - \mathbf{y}_i\|_2 - \widehat{v}_i$. Then, $\widetilde{\mathbf{y}}_{i^*}$ is added to $\widetilde{\mathcal{Y}}_{i^*}$. We have

$$l_K(\widetilde{\mathbf{y}}_{i^*}, \widehat{\theta}^s) - \widehat{v}^s_{N+1} \cdot \|\widetilde{\mathbf{y}}_{i^*} - \mathbf{y}_{i^*}\|_2 - \widehat{v}^s_{i^*} > \delta. \tag{D.6}$$

$\forall t > s$, we know that

$$l_K(\widetilde{\mathbf{y}}_{i^*}, \widehat{\theta}^t) - \widehat{v}^t_{N+1} \cdot \|\widetilde{\mathbf{y}}_{i^*} - \mathbf{y}_{i^*}\|_2 - \widehat{v}^t_{i^*} \leq 0. \tag{D.7}$$

Combining (D.6) and (D.7), we have that, for $s < t$,

$$l_K(\widetilde{\mathbf{y}}_{i^*}, \widehat{\theta}^s) - \widehat{v}^s_{N+1} \cdot \|\widetilde{\mathbf{y}}_{i^*} - \mathbf{y}_{i^*}\|_2 - \widehat{v}^s_{i^*} - (l_K(\widetilde{\mathbf{y}}_{i^*}, \widehat{\theta}^t) - \widehat{v}^t_{N+1} \cdot \|\widetilde{\mathbf{y}}_{i^*} - \mathbf{y}_{i^*}\|_2 - \widehat{v}^t_{i^*}) > \delta. \tag{D.8}$$

Note that

$$
\begin{aligned}
&l_K(\widetilde{\mathbf{y}}_{i^*}, \widehat{\theta}^s) - \widehat{v}^s_{N+1} \cdot \|\widetilde{\mathbf{y}}_{i^*} - \mathbf{y}_{i^*}\|_2 - \widehat{v}^s_{i^*} - (l_K(\widetilde{\mathbf{y}}_{i^*}, \widehat{\theta}^t) - \widehat{v}^t_{N+1} \cdot \|\widetilde{\mathbf{y}}_{i^*} - \mathbf{y}_{i^*}\|_2 - \widehat{v}^t_{i^*}) \\
&= l_K(\widetilde{\mathbf{y}}_{i^*}, \widehat{\theta}^s) - l_K(\widetilde{\mathbf{y}}_{i^*}, \widehat{\theta}^t) - (\widehat{v}^s_{N+1} \cdot \|\widetilde{\mathbf{y}}_{i^*} - \mathbf{y}_{i^*}\|_2 - \widehat{v}^t_{N+1} \cdot \|\widetilde{\mathbf{y}}_{i^*} - \mathbf{y}_{i^*}\|_2) - (\widehat{v}^s_{i^*} - \widehat{v}^t_{i^*}) \\
&\leq \tfrac{4(B+R)\kappa}{\lambda} \|\widehat{\theta}^s - \widehat{\theta}^t\|_2 + 2R \|\widehat{v}^s_{N+1} - \widehat{v}^t_{N+1}\|_2 + |\widehat{v}^s_{i^*} - \widehat{v}^t_{i^*}| \\
&\leq (1 + 2R + \tfrac{4(B+R)\kappa}{\lambda}) \|(\widehat{\theta}^s, \widehat{\mathbf{v}}^s) - (\widehat{\theta}^t, \widehat{\mathbf{v}}^t)\|_2
\end{aligned}
\tag{D.9}
$$

where the first inequality is due to the Lipschitz condition in Lemma 4.3.1(c).

Combining (D.8) and (D.9), we have that, for $s < t$,

$$
\|(\widehat{\theta}^s, \widehat{\mathbf{v}}^s) - (\widehat{\theta}^t, \widehat{\mathbf{v}}^t)\|_2 > \frac{\delta}{G}. \tag{D.10}
$$

Thus, the minimum distance between any two solutions $(\widehat{\theta}^1, \widehat{\mathbf{v}}^1), \ldots, (\widehat{\theta}^S, \widehat{\mathbf{v}}^S)$ exceeds δ/G.

Let $\mathscr{B}_s \in \mathbb{R}^{n_\theta+N+1}$ denote the ball centered at $(\widehat{\theta}^s, \widehat{\mathbf{v}}^s)$ with radius δ/G. Let $R_0 = \sqrt{D^2 + N((m+1)V_2 - mV_1)^2 + (\frac{(V_2-V_1)}{\epsilon})^2}$. Let $\mathscr{B}$ denote the ball centered at origin with radius $R_0 + \frac{\delta}{G}$. It follows that the balls $\{\mathscr{B}_s\}_{s\in[S]}$, which do not intersect with each other, are covered by $\mathscr{B}$. Thus, we have

$$
S\beta_{n_\theta+N+1}(\frac{\delta}{G})^{n_\theta+N+1} \leq \beta_{n_\theta+N+1}(R_0 + \frac{\delta}{G})^{n_\theta+N+1}
$$

where $\beta_{n_\theta+N+1}$ is the volume of the unit ball in $\mathbb{R}^{n_\theta+N+1}$. Thus, we conclude that

$$
S \leq (\frac{GR_0}{\delta} + 1)^{n_\theta+N+1}.
$$

□

D.1.5 Proof of Theorem 4.3.4

Proof. Our proof of the excess risk bound relies on Theorem 2 in [35]. We first verify the assumptions. By Lemma 4.3.1 (a), Assumption 1 is obviously satisfied since $diam(\mathcal{Y}) \leq 2R$, where $diam(\mathcal{Y})$ is the diameter of the observation space $\mathcal{Y}$. $\forall \theta \in \Theta$, the loss function $l_K(\mathbf{y}, \theta)$ is $2(B+R)$-Lipschitz continuous in $\mathbf{y}$, and $0 \leq l_K(\mathbf{y}, \theta) \leq (B+R)^2$ by Lemma 4.3.1 (a). Thus, Assumption 2 holds. In addition, Assumption 3 is naturally satisfied according to Lemma 4.3.1 (b).

Denote $\mathcal{F} := \{l_K(\cdot, \theta) : \theta \in \Theta\}$ the class of the loss functions. Before evaluating the Dudley entropy integral, we need to estimate the covering number $\mathcal{N}(\mathcal{F}, \|\cdot\|_\infty, \cdot)$. First observe that for any two $l_K(\cdot, \theta_1), l_K(\cdot, \theta_2) \in \mathcal{F}$ corresponding to $\theta_1, \theta_2 \in \Theta$, we have

$$|l_K(\mathbf{y}, \theta_1) - l_K(\mathbf{y}, \theta_2)| \leq \frac{4(B+R)\kappa}{\lambda} \|\theta_1 - \theta_2\|_2.$$

Since Θ belongs to the ball in $\mathbb{R}^{n_\theta}$ with radius D,

$$\mathcal{N}(\mathcal{F}, \|\cdot\|_\infty, u) \leq \mathcal{N}(\Theta, \|\cdot\|_2, \frac{\lambda}{4(B+R)\kappa} u) \leq \left(\frac{12D(B+R)}{\kappa u}\right)^{n_\theta}$$

for $0 < u < \frac{4D(B+R)}{\kappa}$, and $\mathcal{N}(\mathcal{F}, \|\cdot\|_\infty, u) = 1$ for $u \geq \frac{4D(B+R)}{\kappa}$, which leads to

$$\begin{aligned}
\int_0^\infty \sqrt{\log \mathcal{N}(\mathcal{F}, \|\cdot\|_\infty, u)} du &\leq \int_0^{\frac{4D(B+R)}{\kappa}} \sqrt{n_\theta \log\left(\frac{12D(B+R)}{\kappa u}\right)} du \\
&= \frac{12D(B+R)}{\kappa} \sqrt{n_\theta} \int_0^{1/3} \sqrt{\log(1/u)} du \\
&\leq \frac{6D(B+R)}{\kappa} \sqrt{n_\theta}.
\end{aligned}$$

Substituting this into the bound provided in Theorem 2 of [35], we get the desired estimate.

□

D.2 Data for the Portfolio Optimization Problem

Table 17: True expected return

Security	1	2	3	4	5	6	7	8
Expected Return	0.1791	0.1143	0.1357	0.0837	0.1653	0.1808	0.0352	0.0368

Table 18: True return covariances Matrix

Security	1	2	3	4	5	6	7	8
1	0.1641	0.0299	0.0478	0.0491	0.058	0.0871	0.0603	0.0492
2	0.0299	0.0720	0.0511	0.0287	0.0527	0.0297	0.0291	0.0326
3	0.0478	0.0511	0.0794	0.0498	0.0664	0.0479	0.0395	0.0523
4	0.0491	0.0287	0.0498	0.1148	0.0336	0.0503	0.0326	0.0447
5	0.0580	0.0527	0.0664	0.0336	0.1073	0.0483	0.0402	0.0533
6	0.0871	0.0297	0.0479	0.0503	0.0483	0.1134	0.0591	0.0387
7	0.0603	0.0291	0.0395	0.0326	0.0402	0.0591	0.0704	0.0244
8	0.0492	0.0326	0.0523	0.0447	0.0533	0.0387	0.0244	0.1028

Bibliography

[1] Arezou Keshavarz, Yang Wang, and Stephen Boyd. Imputing a convex objective function. In *Intelligent Control (ISIC), 2011 IEEE International Symposium on*, pages 613–619. IEEE, 2011.

[2] Ravindra K Ahuja and James B Orlin. Inverse optimization. *Operations Research*, 49(5):771–783, 2001.

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

[4] Andrew J. Schaefer. Inverse integer programming. *Optimization Letters*, 3(4):483–489, 2009.

[5] Lizhi Wang. Cutting plane algorithms for the inverse mixed integer linear programming problem. *Operations Research Letters*, 37(2):114–116, 2009.

[6] Andreas Bärmann, Sebastian Pokutta, and Oskar Schneider. Emulating the expert: Inverse optimization through online learning. In *International Conference on Machine Learning*, pages 400–410, 2017.

[7] Anil Aswani, Zuo-Jun Shen, and Auyon Siddiq. Inverse optimization with noisy data. *Operations Research*, 2018.

[8] Timothy CY Chan, Tim Craig, Taewoo Lee, and Michael B Sharpe. Generalized inverse multiobjective optimization with application to cancer therapy. *Operations Research*, 62(3):680–695, 2014.

[9] Dimitris Bertsimas, Vishal Gupta, and Ioannis Ch Paschalidis. Inverse optimization: A new perspective on the Black-Litterman model. *Operations Research*, 60(6):1389–1403, 2012.

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

[11] Chaosheng Dong and Bo Zeng. Inferring parameters through inverse multiobjective optimization. *arXiv preprint arXiv:1808.00935*, 2018.

[12] Charu C Aggarwal. *Recommender Systems: The Textbook*. Springer, 2016.

[13] Dimitris Bertsimas, Vishal Gupta, and Ioannis Ch Paschalidis. Data-driven estimation in equilibrium using inverse optimization. *Mathematical Programming*, 153(2):595–633, 2015.

[14] Chaosheng Dong, Yiran Chen, and Bo Zeng. Generalized inverse optimization through online learning. *arXiv preprint arXiv:1810.01920*, 2018.

[15] Salvatore Greco, J Figueira, and M Ehrgott. *Multiple Criteria Decision Analysis*. Springer, 2016.

[16] C-L Hwang and Abu Syed Md Masud. *Multiple Objective Decision Making—methods and Applications: A State-of-the-art Survey*, volume 164. Springer Science & Business Media, 2012.

[17] Harry Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952.

[18] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297, 1967.

[19] S. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, March 1982.

[20] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[21] Daniel Aloise, Amit Deshpande, Pierre Hansen, and Preyas Popat. Np-hardness of euclidean sum-of-squares clustering. *Machine learning*, 75(2):245–248, 2009.

[22] Meena Mahajan, Prajakta Nimbhorkar, and Kasturi Varadarajan. The planar k-means problem is np-hard. *Theoretical Computer Science*, 442:13–21, 2012.

[23] Kiri Wagstaff, Claire Cardie, Seth Rogers, and Stefan Schroedl. Constrained k-means clustering with background knowledge. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 577–584. Citeseer, 2001.

[24] Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

[25] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

[26] Timothy CY Chan, Taewoo Lee, and Daria Terekhov. Inverse optimization: Closed-form solutions, geometry, and goodness of fit. *Management Science*, 2018.

[27] Soroosh Shafieezadeh-Abadeh, Peyman Mohajerin Esfahani, and Daniel Kuhn. Distributionally robust logistic regression. In *NIPS*, 2015.

[28] Rui Gao and Anton J Kleywegt. Distributionally robust stochastic optimization with wasserstein distance. *arXiv preprint arXiv:1604.02199*, 2016.

[29] Fengqiao Luo and Sanjay Mehrotra. Decomposition algorithm for distributionally robust optimization using wasserstein metric. *arXiv preprint arXiv:1704.03920*, 2017.

[30] Peyman Mohajerin Esfahani and Daniel Kuhn. Data-driven distributionally robust optimization using the wasserstein metric: Performance guarantees and tractable reformulations. *Mathematical Programming*, 171(1-2):115–166, 2018.

[31] Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

[32] Hongseok Namkoong and John C Duchi. Variance-based regularization with convex objectives. In *NIPS*, 2017.

[33] Soroosh Shafieezadeh Abadeh, Viet Anh Nguyen, Daniel Kuhn, and Peyman Mohajerin Mohajerin Esfahani. Wasserstein distributionally robust kalman filtering. In *NeurIPS*, 2018.

[34] Rui Gao, Liyan Xie, Yao Xie, and Huan Xu. Robust hypothesis testing using wasserstein uncertainty sets. In *NeurIPS*, 2018.

[35] Jaeho Lee and Maxim Raginsky. Minimax statistical learning with wasserstein distances. In *NeurIPS*, 2018.

[36] Didier Burton and Ph L Toint. On an instance of the inverse shortest paths problem. *Mathematical Programming*, 53(1):45–61, 1992.

[37] Jianzhong Zhang and Zhenhong Liu. Calculating some inverse linear programming problems. *Journal of Computational and Applied Mathematics*, 72(2):261–273, 1996.

[38] Ravindra K Ahuja and James B Orlin. A faster algorithm for the inverse spanning tree problem. *Journal of Algorithms*, 34(1):177–193, 2000.

[39] Ravindra K Ahuja and James B Orlin. Combinatorial algorithms for inverse network flow problems. *Networks*, 40(4):181–187, 2002.

[40] Clemens Heuberger. Inverse combinatorial optimization: A survey on problems, methods, and results. *Journal of Combinatorial Optimization*, 8(3):329–361, 2004.

[41] Adrian Deaconu. The inverse maximum flow problem considering l_∞ norm. *RAIRO-Operations Research*, 42(3):401–414, 2008.

[42] Çiğdem Güler and Horst W Hamacher. Capacity inverse minimum cost flow problem. *Journal of Combinatorial Optimization*, 19(1):43–59, 2010.

[43] Jianzhong Zhang and Chengxian Xu. Inverse optimization for linearly constrained convex separable programming problems. *European Journal of Operational Research*, 200(3):671–679, 2010.

[44] Jianzhong Zhang and Liwei Zhang. An augmented lagrangian method for a class of inverse quadratic programming problems. *Applied Mathematics and Optimization*, 61(1):57, 2010.

[45] Andrew Y Ng and Stuart J Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 663–670. Morgan Kaufmann Publishers Inc., 2000.

[46] Stephan Dempe and Sebastian Lohse. Inverse linear programming. In *Recent Advances in Optimization*, pages 19–28. Springer, 2006.

[47] Timothy C. Y. Chan, Taewoo Lee, and Daria Terekhov. Inverse optimization: Closed-form solutions, geometry and goodness of fit. ArXiv e-prints arXiv:1511.04650, 2015.

[48] Hai Yang, Tsuna Sasaki, Yasunori Iida, and Yasuo Asakura. Estimation of origin-destination matrices from link traffic counts on congested networks. *Transportation Research Part B: Methodological*, 26(6):417–434, 1992.

[49] Michel Baes, Moritz Diehl, and Ion Necoara. Every continuous nonlinear control system can be obtained by parametric convex programming. *IEEE Transactions on Automatic Control*, 53(8):1963–1967, 2008.

[50] Ngoc Anh Nguyen, Sorin Olaru, Pedro Rodriguez-Ayerbe, Morten Hovd, and Ion Necoara. Inverse parametric convex programming problems via convex liftings. *IFAC Proceedings Volumes*, 47(3):2489–2494, 2014.

[51] Andreas B Hempel, Paul J Goulart, and John Lygeros. Inverse parametric optimization with an application to hybrid system control. *IEEE Transactions on Automatic Control*, 60(4):1064–1069, 2015.

[52] Ngoc Anh Nguyen, Sorin Olaru, and Pedro Rodriguez-Ayerbe. Any discontinuous PWA function is optimal solution to a parametric linear programming problem. In *Decision and Control (CDC), 2015 IEEE 54th Annual Conference on*, pages 5926–5931. IEEE, 2015.

[53] N. A. Nguyen, S. Olaru, P. Rodriguez-Ayerbe, M. Hovd, and I. Necoara. Constructive solution of inverse parametric linear/quadratic programming problems. *Journal of Optimization Theory and Applications*, 172(2):623–648, 2017.

[54] Marvin D Troutt, Wan-Kai Pang, and Shui-Hung Hou. Behavioral estimation of mathematical programming objective function coefficients. *Management Science*, 52(3):422–434, 2006.

[55] Li Cheng, Dale Schuurmans, Shaojun Wang, Terry Caelli, and Svn Vishwanathan. Implicit online learning with kernels. In *Advances in Neural Information Processing Systems*, pages 249–256, 2007.

[56] Brian Kulis and Peter L Bartlett. Implicit online learning. In *Proceedings of the 27th International Conference on Machine Learning*, pages 575–582. Citeseer, 2010.

[57] J Frédéric Bonnans and Alexander Shapiro. Optimization problems with perturbations: A guided tour. *SIAM Review*, 40(2):228–264, 1998.

[58] J Frédéric Bonnans and Alexander Shapiro. *Perturbation Analysis of Optimization Problems*. Springer Science & Business Media, 2013.

[59] Jialei Wang, Weiran Wang, and Nathan Srebro. Memory and communication efficient distributed stochastic optimization with minibatch prox. In *Proceedings of the 2017 Conference on Learning Theory*, volume 65, pages 1882–1919, 2017.

[60] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

[61] Nicholas A. Nystrom, Michael J. Levine, Ralph Z. Roskies, and J. Ray Scott. Bridges: A uniquely flexible HPC resource for new communities and data analytics. In *Proceedings of the 2015 XSEDE Conference: Scientific Advancements Enabled by Enhanced Cyberinfrastructure*, XSEDE '15, pages 30:1–30:8. ACM, 2015.

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

[63] Andreu Mas-Collell, Michael Whinston, and Jerry R Green. Microeconomic theory. 1995.

[64] Julien Roland, Yves De Smet, and José Rui Figueira. Inverse multi-objective combinatorial optimization. *Discrete Applied Mathematics*, 161(16):2764–2771, 2013.

[65] Timothy CY Chan and Taewoo Lee. Trade-off preservation in inverse multi-objective convex optimization. *European Journal of Operational Research*, 270(1):25–39, 2018.

[66] Saul Gass and Thomas Saaty. The computational algorithm for the parametric objective function. *Naval Research Logistics*, 2(1-2):39–45, 1955.

[67] Kaisa Miettinen. *Nonlinear Multiobjective Optimization*, volume 12. Springer Science & Business Media, 2012.

[68] Matthias Ehrgott and Margaret M Wiecek. Mutiobjective programming. In *Multiple Criteria Decision Analysis: State of the Art Surveys*, pages 667–708. Springer, 2005.

[69] Arthur R Warburton. Quasiconcave vector maximization: connectedness of the sets of pareto-optimal and weak pareto-optimal alternatives. *Journal of Optimization Theory and Applications*, 40(4):537–557, 1983.

[70] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, 2001.

[71] O. Chapelle, B. Schölkopf, and A. Zien. *Semi-supervised Learning*. MIT Press, 2006.

[72] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

[73] Adil M Bagirov. Modified global k-means algorithm for minimum sum-of-squares clustering problems. *Pattern Recognition*, 41(10):3192–3199, 2008.

[74] Daniel Aloise and Pierre Hansen. A branch-and-cut sdp-based algorithm for minimum sum-of-squares clustering. *Pesquisa Operacional*, 29(3):503–516, 2009.

[75] Lawrence K Saul and Sam T Roweis. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *Journal of machine learning research*, 4(Jun):119–155, 2003.

[76] Zhenyue Zhang and Hongyuan Zha. Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. *SIAM journal on scientific computing*, 26(1):313–338, 2004.

[77] Andrew Smith, Hongyuan Zha, and Xiao-ming Wu. Convergence and rate of convergence of a manifold-based dimension reduction algorithm. In *Advances in Neural Information Processing Systems*, pages 1529–1536, 2009.

[78] Trevor Hastie and Werner Stuetzle. Principal curves. *Journal of the American Statistical Association*, 84(406):502–516, 1989.

[79] Robert I Jennrich. Asymptotic properties of non-linear least squares estimators. *The Annals of Mathematical Statistics*, 40(2):633–643, 1969.

[80] William W Hogan. Point-to-set maps in mathematical programming. *SIAM Review*, 15(3):591–603, 1973.

[81] T Tanino and Y Sawaragi. Stability of nondominated solutions in multicriteria decision-making. *Journal of Optimization Theory and Applications*, 30(2):229–253, 1980.

[82] Eitan Greenshtein and Ya'Acov Ritov. Persistence in high-dimensional linear predictor selection and the virtue of overparametrization. *Bernoulli*, 10(6):971–988, 2004.

[83] Peter L Bartlett and Shahar Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

[84] Vladimir Vapnik. *The Nature of Statistical Learning Theory*. Springer Science & Business Media, 2013.

[85] Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. Introduction to statistical learning theory. In *Advanced Lectures on Machine Learning*, pages 169–207. Springer, 2004.

[86] Stephan Dempe, Vyacheslav Kalashnikov, Gerardo A Pérez-Valdés, and Nataliya Kalashnykova. *Bilevel Programming Problems: Theory, Algorithms and Applications to Energy Networks*. Springer, 2015.

[87] Inc. Gurobi Optimization. Gurobi optimizer reference manual, 2016.

[88] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *International Conference on Artificial Neural Networks*, pages 583–588. Springer, 1997.

[89] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

[90] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.

[91] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(Dec):3371–3408, 2010.

[92] Matthew Brand. Charting a manifold. In *Advances in neural information processing systems*, pages 985–992, 2003.

[93] Jacob Goldberger, Geoffrey E Hinton, Sam T Roweis, and Ruslan R Salakhutdinov. Neighbourhood components analysis. In *Advances in neural information processing systems*, pages 513–520, 2005.

[94] Yafeng Yin and Siriphong Lawphongpanich. Internalizing emission externality on road networks. *Transportation Research Part D: Transport and Environment*, 11(4):292–301, 2006.

[95] Hai Yan and William HK Lam. Optimal road tolls under conditions of queueing and congestion. *Transportation Research Part A: Policy and Practice*, 30(5):319–332, 1996.

[96] Anna Nagurney. Congested urban transportation networks and emission paradoxes. *Transportation Research Part D: Transport and Environment*, 5(2):145–151, 2000.

[97] Rainer Hettich and Kenneth O Kortanek. Semi-infinite programming: theory, methods, and applications. *SIAM Review*, 35(3):380–429, 1993.

[98] Nikolaos V Sahinidis. Baron: A general purpose global optimization software package. *Journal of global optimization*, 8(2):201–205, 1996.

[99] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[100] Imre Pólik and Tamás Terlaky. A survey of the s-lemma. *SIAM Review*, 49(3):371–418, 2007.

[101] Almir Mutapcic and Stephen Boyd. Cutting-set methods for robust convex optimization with pessimizing oracles. *Optimization Methods & Software*, 24(3):381–406, 2009.

[102] Claude Berge. *Topological Spaces: Including a Treatment of Multi-valued Functions, Vector Spaces, and Convexity*. Courier Corporation, 1963.

[103] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. MIT Press, 2012.

[104] Gérard Biau, Luc Devroye, and Gábor Lugosi. On the performance of clustering in hilbert spaces. *IEEE Transactions on Information Theory*, 54(2):781–790, 2008.

[105] Wilson A Sutherland. *Introduction to Metric and Topological Spaces*. Oxford University Press, 2009.

[106] Roland Glowinski and A Marroco. Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires. *Revue Française D'automatique, Informatique, Recherche Opérationnelle. Analyse Numérique*, 9(2):41–76, 1975.

[107] Daniel Gabay and Bertrand Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications*, 2(1):17–40, 1976.

[108] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[109] Wei Shi, Qing Ling, Kun Yuan, Gang Wu, and Wotao Yin. On the linear convergence of the admm in decentralized consensus optimization. *IEEE Transactions on Signal Processing*, 7(62):1750–1761, 2014.

[110] Wei Deng and Wotao Yin. On the global and linear convergence of the generalized alternating direction method of multipliers. *Journal of Scientific Computing*, 66(3):889–916, 2016.

[111] Mingyi Hong and Zhi-Quan Luo. On the linear convergence of the alternating direction method of multipliers. *Mathematical Programming*, 162(1):165–199, 2017.

[112] Caihua Chen, Bingsheng He, Yinyu Ye, and Xiaoming Yuan. The direct extension of admm for multi-block convex minimization problems is not necessarily convergent. *Mathematical Programming*, 155(1-2):57–79, 2016.

[113] Steven Diamond, Reza Takapoui, and Stephen Boyd. A general system for heuristic solution of convex problems over nonconvex sets. ArXiv e-prints arXiv:1601.07277, 2016.

[114] Sindri Magnússon, Pradeep Chathuranga Weeraddana, Michael G Rabbat, and Carlo Fischione. On the convergence of alternating direction Lagrangian methods for nonconvex structured optimization problems. *IEEE Transactions on Control of Network Systems*, 3(3):296–309, 2016.

[115] A. Alavian and M. C. Rotkowitz. Improving admm-based optimization of mixed integer objectives. In *2017 51st Annual Conference on Information Sciences and Systems*, pages 1–6, March 2017.