informs
http://pubsonline.informs.org/journal/ijoc

INFORMS JOURNAL ON COMPUTING
Vol. 34, No. 5, September–October 2022, pp. 2720–2735
ISSN 1091-9856 (print), ISSN 1526-5528 (online)

Decomposition and Adaptive Sampling for Data-Driven Inverse Linear Optimization

Rishabh Gupta,[a] Qi Zhang[a,*]

[a] Department of Chemical Engineering and Materials Science, University of Minnesota, Minneapolis, Minnesota 55455
*Corresponding author
Contact: rishabh@umn.edu (RG); qizh@umn.edu, https://orcid.org/0000-0001-8862-4675 (QZ)

Received: September 27, 2020
Revised: April 14, 2021; October 26, 2021; November 19, 2021; November 23, 2021; December 6, 2021
Accepted: December 14, 2021
Published Online in Articles in Advance: June 8, 2022

https://doi.org/10.1287/ijoc.2022.1162

Abstract. This work addresses inverse linear optimization, where the goal is to infer the unknown cost vector of a linear program. Specifically, we consider the data-driven setting in which the available data are noisy observations of optimal solutions that correspond to different instances of the linear program. We introduce a new formulation of the problem that, compared with other existing methods, allows the recovery of a less restrictive and generally more appropriate admissible set of cost estimates. It can be shown that this inverse optimization problem yields a finite number of solutions, and we develop an exact two-phase algorithm to determine all such solutions. Moreover, we propose an efficient decomposition algorithm to solve large instances of the problem. The algorithm extends naturally to an online learning environment where it can be used to provide quick updates of the cost estimate as new data become available over time. For the online setting, we further develop an effective adaptive sampling strategy that guides the selection of the next samples. The efficacy of the proposed methods is demonstrated in computational experiments involving two applications: customer preference learning and cost estimation for production planning. The results show significant reductions in computation and sampling efforts.
Summary of Contribution: Using optimization to facilitate decision making is at the core of operations research. This work addresses the inverse problem (i.e., inverse optimization), which aims to infer unknown optimization models from decision data. It is, conceptually and computationally, a challenging problem. Here, we propose a new formulation of the data-driven inverse linear optimization problem and develop an efficient decomposition algorithm that can solve problem instances up to a scale that has not been addressed previously. The computational performance is further improved by an online adaptive sampling strategy that substantially reduces the number of required data points.

History: Accepted by Antonio Frangioni, Area Editor for Design & Analysis of Algorithms—Continuous.
Funding: The authors gratefully acknowledge the financial support from the National Science Foundation [Grant 2044077]. R. Gupta acknowledges financial support from a departmental fellowship sponsored by 3M.
Supplemental Material: The software that supports the findings of this study is available within the paper and its Supplementary Information [https://pubsonline.informs.org/doi/suppl/10.1287/ijoc.2022.1162] or is available from the IJOC GitHub software repository (https://github.com/INFORMSJoC) at http://dx.doi.org/10.5281/zenodo.5710020.

Keywords: inverse optimization • online learning • adaptive sampling

1. Introduction

Inverse optimization is an emerging new paradigm for uncovering hidden decision-making mechanisms from observed decision data. Following the principle of optimality (Schoemaker 1991) that is commonly applied in various fields including economics, psychology, and evolutionary biology, the key idea in inverse optimization is to model a decision-making process as an optimization problem. Decisions can then be viewed as optimal or near-optimal solutions of an optimization model, and the inverse optimization problem (IOP) is to infer this model, if otherwise unknown, from observations. A major advantage of this approach is its ability to explicitly include constraints. This eases the incorporation of domain knowledge, which is often readily available in the form of constraints, significantly. As a result, compared with common black-box machine learning methods, inverse optimization offers the promise of models with enhanced prediction accuracy and interpretability.

The notion of inverse optimization was first introduced by Burton and Toint (1992), who consider the problem of determining travel costs on a network as perceived by the users given the routes they have taken. This has inspired research on several inverse network optimization problems (Yang et al. 1997, Zhang and Cai 1998, Zhang and Liu 1999, Liu and Zhang 2006). Since then, inverse optimization has found application in a myriad of fields, such as radiation therapy planning (Chan et al. 2014, Babier et al. 2021), investment portfolio optimization (Bertsimas et al. 2012), electricity demand forecasting (Saez-Gallego and Morales 2018), auction mechanism design (Beil and Wein 2003, Birge et al. 2017), biological systems (Burgard and Maranas 2003, Arechavaleta et al. 2008, Terekhov et al. 2010), and optimal control (Hempel et al. 2015, Westermann et al. 2020).

Early works in inverse optimization focus on determining an objective function that makes the observed decisions, given the constraints of the problem, exactly optimal. In their seminal paper, Ahuja and Orlin (2001) present a generalized solution method for inverse optimization with linear forward optimization problems (FOPs). Some of the later works extend the theory to consider conic (Iyengar and Kang 2005, Zhang and Xu 2010), discrete (Schaefer 2009, Wang 2009, Bulut and Ralphs 2021), and nonlinear (Chow et al. 2014) FOPs.

More recently, the research focus has shifted toward *data-driven* inverse optimization, in which we observe an agent's decisions in multiple instances, which can be viewed as instances of the same optimization problem that differ in their input parameter values (Mohajerin Esfahani et al. 2018). With data-driven inverse optimization, one has a much greater chance of learning an optimization model that has true predictive power with respect to future decisions in unseen instances. Here, the observations are generally considered to be *noisy*, with the following being the three key sources of the noise: (i) measurement errors, (ii) bounded rationality of the decision maker, and (iii) model specification mismatch (Aswani et al. 2018, Mohajerin Esfahani et al. 2018). The existing literature for this setting is limited to the case of convex FOPs. The main distinction among the various proposed formulations is in terms of the loss function employed to fit the data. Minimization of the slack required to make the noisy data satisfy an optimality condition is considered by Keshavarz et al. (2011), Bertsimas et al. (2015), and Mohajerin Esfahani et al. (2018). However, Aswani et al. (2018) show that this kind of loss function can lead to statistically inconsistent estimates and propose to minimize the sum of some norm of residuals with respect to the decision variables. In the data-driven context, most existing works assume that the observations are available in a single batch, whereas more recent contributions also address online learning environments in which the observations are made sequentially (Bärmann et al. 2018, Dong et al. 2018, Shahmoradi and Lee 2021).

In this work, we consider data-driven inverse *linear* optimization with noisy observations in both batch and online learning settings. Here, the goal is to estimate the unknown cost vector of a linear program (LP). Inverse linear optimization constitutes an important class of IOPs, as many decision-making problems can be formulated or approximated as LPs. Although inverse linear optimization falls into the broader category of inverse convex optimization for which established solution methods exist, these more general methods often yield overly restricted sets of admissible cost estimates when applied to inverse linear optimization with noisy data (as discussed in detail in Section 2). This limitation is to a great extent shared by tailored approaches specifically designed to solve the IOP in the linear case more efficiently (Chan et al. 2019, Babier et al. 2021). We note that the majority of the inverse linear optimization literature does not consider the data-driven case but focuses on the single-instance setting (with possibly multiple noisy observations).

Our proposed framework is designed to recover the complete set of admissible solutions for the inverse linear optimization problem while incorporating the notion of a reference cost vector that represents the user's prior belief, which facilitates the selection of an appropriate point estimate. On the basis of a polyhedral understanding of the problem, we propose a two-phase approach that separates the tasks of denoising the data and parameter estimation. To solve large instances of the IOP, we develop an exact decomposition algorithm, which processes the data sequentially and can hence also serve as an efficient update method in online inverse optimization. For the online setting, we further develop an adaptive sampling strategy that guides the selection of the next samples in an effort to reduce the amount of required data. Although adaptive sampling is quite a mainstream idea in machine learning (Domingo et al. 2002, Chang et al. 2005, Cozad et al. 2014), to the best of our knowledge, it has not yet been considered in inverse optimization. We believe that the development of such a framework can go a long way in increasing the acceptance of inverse optimization as an alternative to black-box modeling methods, especially in situations where data acquisition is expensive or time intensive.

The main contributions of this work are as follows:

1. We introduce a new general formulation of the data-driven inverse linear optimization problem that considers multiple noisy observations collected for multiple experiments, which are problem instances with different input parameter values. We highlight several geometrical properties of the problem and show that by assuming that optimal solutions lie at the vertices of the feasible region, we can recover the complete set of admissible cost estimates.

2. We show that the proposed IOP formulation yields a finite number of solutions. We introduce a two-phase algorithm that can recover all such solutions. Furthermore, we show that under a very mild condition, the IOP is guaranteed to have a unique solution.

3. We develop an efficient sequential decomposition algorithm to solve large instances of the IOP. The algorithm directly extends itself to online inverse optimization where it can be used to provide quick updates of the cost estimate as new data become available.

4. We propose an effective adaptive sampling strategy that guides the choice of the experiments in online inverse optimization. The adaptive sampling problem is formulated as a mixed-integer nonlinear program (MINLP) for which we provide an efficient heuristic solution algorithm.

5. We demonstrate the effectiveness of the proposed framework through a comprehensive set of computational experiments, addressing the problems of customer preference learning and cost estimation for production planning. The results indicate that generally, reasonable prediction accuracies can be achieved with relatively small numbers of experiments. Also, one can observe significant reductions in solution time and required number of samples as a result of the proposed decomposition and adaptive sampling methods, respectively.

The remainder of this paper is organized as follows. In Section 2, we present a formal description of the inverse linear optimization problem. In Section 3, we propose a new formulation that utilizes a reference cost vector to find reasonable cost estimates, discuss its properties, and develop a two-phase solution algorithm. Section 4 introduces an exact decomposition algorithm that allows the efficient solution of large instances of the IOP and naturally extends to online inverse optimization. Our proposed adaptive sampling framework is detailed in Section 5. In Section 6, results from the computational studies are presented. Finally, we conclude in Section 7. All omitted proofs can be found in Section A of the online supplement.

2. Background of Inverse Linear Optimization

Consider a decision-making problem that can be represented as an LP of the following form:

$$\begin{aligned} \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad & c^\top x \\ \text{subject to} \quad & Ax \leq b, \end{aligned} \tag{FOP}$$

which we call FOP, short for the forward optimization problem. The cost vector $c \in \mathbb{R}^n$ is unknown; however, experiments perturbing values in $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$ can be designed to help improve our estimate of c. These experiments are subject to certain problem-specific restrictions on A and b, and we denote the set of their allowed values by Π (i.e., $(A, b) \in \Pi$). Moreover, we assume that Π is such that for any $(A, b) \in \Pi$, the polyhedron represented by $Ax \leq b$ is compact and nonempty. This is a mild assumption, as in essentially all real-world problems, the decision variables are bounded. The results of the perturbation experiments, which are assumed to be optimal solutions to mathematical optimization problem (FOP), are observed with some random noise. For a specific experiment, multiple samples can be collected such that one can recover the true optimal solution with some confidence. In what follows, we refer to this estimate of the true optimal solution as the *denoised* estimate and the process of obtaining it as *denoising* the data.

Given observations, the inverse optimization problem (IOP) is to obtain an estimate of c, $\hat{c}$, such that the difference between the observations and the solutions obtained from solving (FOP) with $\hat{c}$ as the cost vector is minimized. The IOP is commonly formulated as follows:

$$\begin{aligned} \underset{\hat{c} \in \mathbb{R}^n,\, \hat{x}}{\text{minimize}} \quad & \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}_i} \|x_{ij} - \hat{x}_{ij}\| \\ \text{subject to} \quad & \hat{x}_{ij} \in \arg\min_{\tilde{x} \in \mathbb{R}^n} \{\hat{c}^\top \tilde{x} : A_i \tilde{x} \leq b_i\} \quad \forall i \in \mathcal{I},\, j \in \mathcal{J}_i, \end{aligned} \tag{1}$$

where $\mathcal{I}$ is the set of experiments, where each experiment i is associated with inputs (A_i, b_i); $\mathcal{J}_i$ denotes the set of noisy observations for experiment i; and x_{ij} is the observed output for $j \in \mathcal{J}_i$. The objective is to choose $\hat{c}$ and $\hat{x}$ such that the loss function, which is defined as the sum of some norm of the residuals, is minimized. Formulation (1) generalizes existing variants of the IOP from the literature. Some consider a setting in which A and b cannot be changed, which leads to the case of $|\mathcal{I}| = 1$ (Chan and Lee 2018, Chan et al. 2019). Others consider random sampling of A and b without assigning the samples to distinct sets corresponding to specific inputs (Aswani et al. 2018); in this case, we have $|\mathcal{I}| = N$, with N being the total number of samples, and $|\mathcal{J}_i| = 1$ for all $i \in \mathcal{I}$. In fact, splitting the set of samples into input-specific subsets does not affect a formulation such as (1); however, it will be an essential feature of our proposed alternative approach (described more in Section 3).

Problem (1) is a bilevel optimization problem and is typically solved by replacing its lower-level problems with their optimality conditions. Whereas most existing works make use of strong duality (Aswani et al. 2018, Chan et al. 2019, Shahmoradi and Lee 2021), some have also applied reformulations based on the Karush-Kuhn-Tucker (KKT) conditions (Keshavarz et al. 2011, Saez-Gallego et al. 2016). In the case of LPs, both optimality conditions are equivalent. In this work, we use a KKT-based approach, as the duality-based formulation is nonlinear and nonconvex in the constraints because of the presence of bilinear terms, whereas the constraints of the KKT-based formulation can be linearized by introducing binary variables, which is advantageous from a computational standpoint. Thus, we arrive at the following mixed-integer reformulation of (1):

$$\underset{\hat{c}\in\mathbb{R}^n,\,\hat{x},\,s,\,\lambda,\,z}{\text{minimize}} \quad \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}_i}\|x_{ij}-\hat{x}_{ij}\| \tag{2a}$$

$$\text{subject to} \quad \hat{c}+A_i^\top\lambda_{ij}=0 \quad \forall i\in\mathcal{I},\ j\in\mathcal{J}_i, \tag{2b}$$

$$A_i\hat{x}_{ij}+s_{ij}=b_i \quad \forall i\in\mathcal{I},\ j\in\mathcal{J}_i, \tag{2c}$$

$$\lambda_{ij}\le Mz_{ij} \quad \forall i\in\mathcal{I},\ j\in\mathcal{J}_i, \tag{2d}$$

$$s_{ij}\le M(e-z_{ij}) \quad \forall i\in\mathcal{I},\ j\in\mathcal{J}_i, \tag{2e}$$

$$\hat{x}_{ij}\in\mathbb{R}^n,\ s_{ij}\in\mathbb{R}^m_+,\ \lambda_{ij}\in\mathbb{R}^m_+,\ z_{ij}\in\{0,1\}^m \quad \forall i\in\mathcal{I},\ j\in\mathcal{J}_i, \tag{2f}$$

where M is a sufficiently large parameter, and e denotes the all-ones vector. Constraints (2b), (2c), and (2d) and (2e) correspond to the stationarity, primal feasibility, and complementary slackness conditions, respectively. The following theorem characterizes the solution set of (2).

Theorem 1. *For a given feasible $\hat{x}$, the set of feasible $\hat{c}$ in problem (2) is a polyhedral cone.*

Proof. Consider (2) for a specific experiment i and sample $j\in\mathcal{J}_i$ and a corresponding feasible $\hat{x}_{ij}$. Let $\mathcal{K}_i$ be the set of constraints for experiment i, (i.e., resulting from A_i and b_i). From (2f), we have that $\lambda_{ij}\ge 0$; hence, if $\lambda_{ijk}=0$ for all $k\in\mathcal{K}_i$, then (2b) imply that $\hat{c}=0$. Otherwise, if $\exists k\in\mathcal{K}_i$ such that $\lambda_{ijk}>0$, then from constraints (2d) and (2e), we have that $s_{ijk}=0$, and (2c) imply that $\hat{x}_{ij}$ is such that $a_{ik}^\top\hat{x}_{ij}=b_{ik}$, where $a_{ik}\in\mathbb{R}^n$ defines the kth row of A_i. Hence, from (2b), we have that $\hat{c}\in\text{cone}(\{-a_{it}\}_{t\in\mathcal{T}(\hat{x}_{ij})})$, where $\mathcal{T}(\hat{x}_{ij})$ denotes the set of constraints active at $\hat{x}_{ij}$. Because $\mathcal{T}(\hat{x}_{ij})$ is a finite set, the feasible region for $\hat{c}$ associated with experiment i and sample j is a polyhedral cone. As this statement holds for every $i\in\mathcal{I}$ and $j\in\mathcal{J}_i$, we have

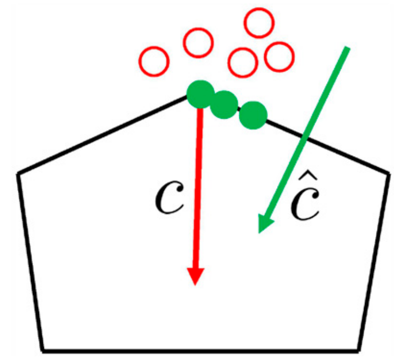
$$\hat{c}\in\bigcap_{i\in\mathcal{I},\,j\in\mathcal{J}_i}\text{cone}\Big(\{-a_{it}\}_{t\in\mathcal{T}(\hat{x}_{ij})}\Big),$$

which is the intersection of a finite number of polyhedral cones and, hence, also a polyhedral cone. □

2.1. Admissible Set

As a consequence of Theorem 1, there is no unique solution to problem (1), or equivalently, (2), because for any optimal $\hat{c}$, $\alpha\hat{c}$, with α being any nonnegative scalar different than 1, yields another optimal solution. Instead, by means of Theorem 1, we can determine the full set of "inverse-optimal" $\hat{c}$, which we refer to as the *admissible set*. Also, to avoid the trivial solution $\hat{c}=0$, it is common to use a slight variant of (1) that restricts the length of $\hat{c}$ by adding a norm constraint (e.g., $\|\hat{c}\|_p=1$) (Mohajerin Esfahani et al. 2018, Chan et al. 2019, Shahmoradi and Lee 2021). However, the use of a norm constraint introduces additional nonconvexity into the problem formulation. Although the choice of p-norm has been arbitrary, under some special conditions on c, the 1-norm and ∞-norm have been shown to lead to tractable formulations (Chan et al. 2019).

Figure 1. (Color online) The Polytope Represents the Feasible Region of the FOP Defined by $A_ix\le b_i$, Where Noisy Data Samples x_{ij} Are Depicted by Red Hollow Circles, and Their Denoised Estimates $\hat{x}_{ij}$ Are Shown as Green Filled Circles



Note. The true cost vector is c, whereas $\hat{c}$ is the solution to (1) with an additional p-norm constraint on $\hat{c}$.

2.2. Noisy Observations

As (FOP) is an LP, with a nonzero c, any optimal solution will lie on the boundary of the polyhedral feasible region. Problem (1) with a p-norm constraint on c can be interpreted as the projection of noisy observations onto one of the polyhedron's facets such that the total projection distance is minimized (Chan et al. 2019). Although this approach provides good solutions when the FOP is strongly convex, it often leads to a severely restricted admissible set when the FOP is an LP. As illustrated in Figure 1, even if the true solution lies at a vertex, noise in the data can cause the observations to get projected onto one of the facets, making a vector orthogonal to that facet the only feasible $\hat{c}$. As highlighted by Shahmoradi and Lee (2021), this formulation also leads to unstable predictions in the presence of outliers in the data.

2.3. Reference Cost

It is important to note that the problem of estimating c given noisy observations consists of two tasks that have to be performed simultaneously: (i) denoising the data to obtain estimates of the real optimal solutions and (ii) using these solutions to estimate c. Formulations of the form (1) and existing variants thereof mainly address the first aspect but have deficiencies, as elucidated in the preceding, when it comes to finding a good point estimate of c, especially when the admissible set is large or the number of observations is small. Traditionally, when sampling is deterministic, inverse optimization has been facilitated by using a reference cost vector $\bar{c}$ and employing an objective function of the form $\|\bar{c} - \hat{c}\|_2^2$ (Ahuja and Orlin 2001, Heuberger 2004). Such a reference cost represents a prior belief that is available in most practical applications (e.g., obtained through first-principles modeling or expert knowledge). A cost estimate $\hat{c}$ that is close to a reference $\bar{c}$ is often desired. It seems that the notion of such a reference, which can aid the process of recovering the real cost vector, has largely been ignored in the noisy case.

3. A Two-Phase Approach to Inverse Optimization

In this section, we propose a general inverse linear optimization model that finds a nontrivial estimate of the cost vector that, among all the ones that minimize the loss function, most closely resembles a reference cost vector. We discuss the main properties of this IOP and develop an exact two-phase solution algorithm.

3.1. Problem Formulation and Properties

We first define $\mathcal{C}'$ as follows:

$$\mathcal{C}' := \left\{ \hat{c} : (\hat{c}, \hat{x}) \in \underset{\hat{c} \in \mathbb{R}^n \setminus \{0\},\, \hat{x}}{\arg\min} \left\{ \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}_i} \|x_{ij} - \hat{x}_{ij}\| : \hat{x}_{ij} \in \underset{\tilde{x} \in \mathbb{R}^n}{\arg\min} \left\{ \hat{c}^\top \tilde{x} : A_i \tilde{x} \le b_i \right\} \ \forall i \in \mathcal{I}, j \in \mathcal{J}_i \right\} \right\}, \tag{3}$$

where it is assumed that for all $i \in \mathcal{I}$, (A_i, b_i) is chosen from some set Π such that the polyhedron $\{\tilde{x} : A_i \tilde{x} \le b_i\}$ is compact and nonempty.

Note that the loss minimization problem (3) is essentially problem (1); however, the above-mentioned representation emphasizes the fact that (1) does not have a unique optimal $\hat{c}$ but rather a set of optimal solutions. The set $\mathcal{C}'$ consists of all these optimal $\hat{c}$ except for the trivial solution $\hat{c} = 0$. Next, we solve the following problem to choose a $\hat{c}$ from $\mathcal{C} = \mathcal{C}' \cup \{0\}$ that most resembles a known reference $\bar{c}$:

$$\begin{aligned} \underset{\hat{c}}{\text{minimize}} \quad & \|\bar{c} - \hat{c}\|_2^2 \\ \text{subject to} \quad & \hat{c} \in \mathcal{C}. \end{aligned} \tag{4}$$

Although we exclude the trivial solution $\hat{c} = 0$ from $\mathcal{C}'$, problem (4) considers $\mathcal{C}$ as the set of admissible cost estimates, which does include the trivial solution. The rationale behind this setup is the following: When determining $\mathcal{C}'$, we do not want to consider $\hat{c} = 0$, which allows a minimum loss but does not provide any useful information. However, admitting the trivial solution in (4) can be helpful because the unlikely case in which $\hat{c} = 0$ is the optimal solution to (4) would immediately indicate that $\bar{c}$ is a very bad reference. In addition, as discussed later in this section, admitting $\mathcal{C}$ in (4) results in useful theoretical properties.

As mentioned in Section 2, solving problem (4) provides good results in the deterministic case; however, it fails when noisy data are considered because the projection of data onto a facet causes $\mathcal{C}$ to be a single ray. To overcome this issue, we propose to consider, instead of $\mathcal{C}'$, the following slightly modified set:

$$\widehat{\mathcal{C}}' := \left\{ \hat{c} : (\hat{c}, \hat{x}) \in \underset{\hat{c} \in \mathbb{R}^n \setminus \{0\},\, \hat{x}}{\arg\min} \left\{ \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}_i} \|x_{ij} - \hat{x}_i\| : \hat{x}_i \in \underset{\tilde{x} \in \mathbb{R}^n}{\arg\min} \left\{ \hat{c}^\top \tilde{x} : A_i \tilde{x} \le b_i \right\} \cap \mathcal{V}_i \ \forall i \in \mathcal{I} \right\} \right\}, \tag{5}$$

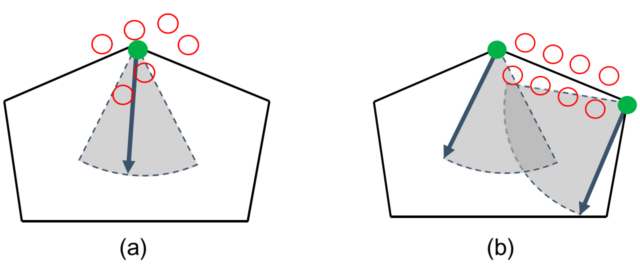
where $\mathcal{V}_i$ denotes the set of extreme points of $\{\tilde{x} : A_i \tilde{x} \le b_i\}$. This leads to the following IOP:

$$\begin{aligned} \underset{\hat{c}}{\text{minimize}} \quad & \|\bar{c} - \hat{c}\|_2^2 \\ \text{subject to} \quad & \hat{c} \in \widehat{\mathcal{C}}, \end{aligned} \tag{IOP}$$

where $\widehat{\mathcal{C}} = \widehat{\mathcal{C}}' \cup \{0\}$. The set $\widehat{\mathcal{C}}'$ considers the projection of the data for each experiment i onto one of the vertices of $\{\tilde{x} : A_i \tilde{x} \le b_i\}$ such that the total projection distance is minimized. We consider two different cases depending on the nature of (FOP) to argue why this approach leads to a more appropriate admissible set. For the ease of exposition, our discussion is restricted to the case of a single experiment (i.e., $|\mathcal{I}| = 1$) with multiple samples:

1. If (FOP) has a unique optimal solution, the noisy samples are likely to be located close to the vertex at which the optimal solution lies (see Figure 2(a)). By solving the loss minimization problem in (5), we obtain a vertex $\hat{x}_i$, and the resulting $\widehat{\mathcal{C}}$ is an exhaustive set of cost vectors that can make $\hat{x}_i$ optimal for (FOP). Recall from Theorem 1 that any $\hat{c} \in \widehat{\mathcal{C}}$ can be expressed as a conic combination of the vectors orthogonal to the

Figure 2. (Color online) Different Cases When Projection of Noisy Data Is Restricted to One of the Vertices of the Polyhedral Feasible Region



Notes. Arrows show the true cost vector c, noisy data samples x_{ij} are depicted by red hollow circles, and their denoised estimates obtained from solving the loss minimization problem in (5) are shown as green filled circles. The gray shaded regions indicate the set $\widehat{\mathcal{C}}$. In panel (b), the loss minimization problem will output only one of the two candidate optimal solutions (green filled circles), and the corresponding gray shaded region will form the set $\hat{\mathcal{C}}$.

facets associated with the constraints active at $\hat{x}_i$. Thus, if $\hat{x}_i$ is the "right" vertex (i.e., the true optimal solution), the true cost vector c will be one of the vectors in $\widehat{\mathcal{C}}$. This is in contrast to $\mathcal{C}$, which in this case would be a single ray that does not contain c.

2. If (FOP) has multiple optimal solutions, the noisy samples are likely to be located close to the facet that represents the set of optimal solutions (see Figure 2(b)). If solving the loss minimization problem in (5) results in a vertex $\hat{x}_i$ that is an optimal solution to (FOP), c will be one of the extreme rays of $\widehat{\mathcal{C}}$. For this case, in most practical instances, we expect the data to show a preference toward a particular vertex of the facet; that is, there is a unique optimal $\hat{x}_i$. Even if multiple $\hat{x}_i$ achieve the same loss, we still have $c \in \widehat{\mathcal{C}}$ as long as at least one of the $\hat{x}_i$ is, in fact, an optimal solution to (FOP). In this case, one may argue that $\widehat{\mathcal{C}}$ is overly large compared with $\mathcal{C}$; however, if $\bar{c} = c$, the true cost vector will be recovered when solving (IOP).

The particular way in which $\widehat{\mathcal{C}}$ is constructed results in number of useful properties, as stated in the following.

Lemma 1. *The set $\widehat{\mathcal{C}}'$ is nonempty.*

This lemma leads to the following theorem.

Theorem 2. *The set of optimal solutions to* (IOP) *is finite.*

Proof of Theorem 2. We start by characterizing the set $\widehat{\mathcal{C}}'$. Consider the minimization problem that describes $\widehat{\mathcal{C}}'$:

$$\begin{aligned} &\underset{\hat{c}\in\mathbb{R}^n\setminus\{0\},\,\hat{x}}{\text{minimize}} && \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}_i} \|x_{ij} - \hat{x}_i\| \\ &\text{subject to} && \hat{x}_i \in \arg\min_{\tilde{x}\in\mathbb{R}^n} \{\hat{c}^\top \tilde{x} : A_i\tilde{x} \le b_i\} \cap \mathcal{V}_i \quad \forall i \in \mathcal{I}. \end{aligned} \tag{6}$$

As stated in Lemma 1, the solution set of problem (6) is nonempty. Furthermore, as $\mathcal{V}_i$ is finite and $\hat{x}_i \in \mathcal{V}_i$, the set of feasible $\hat{x}_i$ is finite. Because we also have a finite number of experiments, the set of feasible $\hat{x}$ to (6) is finite. Hence, the set of optimal $\hat{x}$, which we denote by $\widehat{\mathcal{X}}^*$, is finite. From Theorem 1, the set of feasible $\hat{c}$ for any $\hat{x} \in \widehat{\mathcal{X}}^*$ is $\cap_{i\in\mathcal{I}}\text{cone}(\{-a_{it}\}_{t\in\mathcal{T}(\hat{x}_i)}) \setminus \{0\}$, which is also the set of optimal $\hat{c}$ for that given optimal $\hat{x}$, as $\hat{c}$ does not appear in the objective function of (6). It follows that, considering all $\hat{x} \in \widehat{\mathcal{X}}^*$, the set $\widehat{\mathcal{C}}'$ can be expressed as follows:

$$\widehat{\mathcal{C}}' = \bigcup_{\hat{x}\in\hat{\mathcal{X}}^*} \left\{ \bigcap_{i\in\mathcal{I}} \text{cone}\Big(\{-a_{it}\}_{t\in\mathcal{T}(\hat{x}_i)}\Big) \right\} \setminus \{0\}. \tag{7}$$

Problem (IOP) can then be solved by solving the following problem for every $\hat{x} \in \widehat{\mathcal{X}}^*$:

$$\begin{aligned} &\underset{\hat{c}}{\text{minimize}} && \|\bar{c} - \hat{c}\|_2^2 \\ &\text{subject to} && \hat{c} \in \bigcap_{i\in\mathcal{I}} \text{cone}\Big(\{-a_{it}\}_{t\in\mathcal{T}(\hat{x}_i)}\Big), \end{aligned} \tag{8}$$

which is the minimization of a strictly convex function over a convex feasible region and hence has a unique optimal solution, which we denote by $\tilde{c}(\hat{x})$. The set $\widetilde{\mathcal{C}} := \{\tilde{c}(\hat{x})\}_{\hat{x}\in\hat{\mathcal{X}}^*}$ is finite. Hence, the set of optimal solutions to (IOP), which can be expressed as

$$\widetilde{\mathcal{C}}^* := \left\{ \hat{c} : \hat{c} \in \widetilde{\mathcal{C}},\ \|\bar{c} - \hat{c}\|_2^2 = \min_{\hat{x}\in\widehat{\mathcal{X}}^*} \|\bar{c} - \tilde{c}(\hat{x})\|_2^2 \right\}, \tag{9}$$

is also finite. □

Condition 1. *There is a unique optimal $\hat{x}$ to problem* (6).

At first glance, Condition 1 seems to be very restrictive. But, in fact, it holds in almost all practical instances. Consider an instance in which at any optimal solution to (6), there is a unique optimal $\hat{x}_i$ for each experiment i except for one experiment p. Let $\hat{x}_i^*$ be the unique optimal $\hat{x}_i$ for all $i \in \mathcal{I} \setminus \{p\}$, and let $\widehat{\mathcal{X}}_p^*$ denote the set of multiple optimal $\hat{x}_p$. Then, the following two conditions have to hold: (i) The resulting loss associated with experiment p (i.e., $\sum_{j\in\mathcal{J}_p} \|x_{pj} - \hat{x}_p\|_2^2$) is the same for all $\hat{x}_p \in \widehat{\mathcal{X}}_p^*$. (ii) For every $\hat{x}_p \in \widehat{\mathcal{X}}_p^*$, there exists a $\hat{c}$ that renders all $\hat{x}_i^*$, $i \in \mathcal{I} \setminus \{p\}$, and $\hat{x}_p$ optimal for the FOP, which is equivalent to the following condition:

$$\text{cone}\left(\{-a_{pt}\}_{t\in\mathcal{T}(\hat{x}_p)}\right) \bigcap \left(\bigcap_{i\in\mathcal{I}\setminus\{p\}} \text{cone}\Big(\{-a_{it}\}_{t\in\mathcal{T}(\hat{x}_i^*)}\Big) \right) \ne \{0\} \quad \forall \hat{x}_p \in \widehat{\mathcal{X}}_p^*. \tag{10}$$

Although the first condition is already very unlikely to hold in practice, the second also becomes more improbable for $|\widehat{\mathcal{X}}_p^*| > 1$ as the number of experiments increases. Hence, we conclude that the case of $|\widehat{\mathcal{X}}^*| > 1$ is highly unlikely, which is consistent with our observation in our computational experiments. In theory, however, especially if $|\mathcal{I}|$ is small, there is the possibility that Condition 1 does not hold. In the next subsection, we present a two-phase algorithm that is guaranteed to find the complete set of optimal solutions to (IOP) even if Condition 1 does not hold.

Corollary 1. *If Condition 1 holds,* (IOP) *has a unique solution.*

Thus, in most cases, solving (IOP) will result in a unique $\hat{c}$ that is closest to the prior belief $\bar{c}$. This is in contrast to other approaches in the literature where a variant of (6) is solved that results in a set of solutions from which a $\hat{c}$ is randomly selected. Note that although all $\hat{c} \in \widehat{\mathcal{C}}$ achieve the same prediction accuracy on the training set, they may not show the same performance on unseen data. Our approach of incorporating a prior belief $\bar{c}$ resolves this ambiguity and determines whether $\bar{c}$ is in the admissible set or, if not,

INFORMS Journal on Computing, 2022, vol. 34, no. 5, pp. 2720–2735, © 2022 INFORMS

how much it is outside the admissible set. This is a generally desirable feature in practice.

3.2. Two-Phase Algorithm

We start by presenting tractable reformulations of (6) and (8), and then we show how they can be combined to obtain the set of optimal solutions to (IOP).

Applying a KKT-based approach, we obtain the following reformulation of (6):

$$
\begin{aligned}
\underset{\hat{c},\,\hat{x},\,s,\,\lambda,\,z,\,w,\,\hat{c}^+,\,\hat{c}^-}{\text{minimize}} \quad & \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}_i}\|x_{ij}-\hat{x}_i\| && \text{(P1a)}\\
\text{subject to} \quad & \hat{c}+A_i^\top\lambda_i=0 \quad \forall i\in\mathcal{I}, && \text{(P1b)}\\
& A_i\hat{x}_i+s_i=b_i \quad \forall i\in\mathcal{I}, && \text{(P1c)}\\
& \lambda_i\le Mz_i \quad \forall i\in\mathcal{I}, && \text{(P1d)}\\
& s_i\le M(e-z_i) \quad \forall i\in\mathcal{I}, && \text{(P1e)}\\
& e^\top z_i\ge n \quad \forall i\in\mathcal{I}, && \text{(P1f)}\\
& \hat{c}=\hat{c}^+-\hat{c}^-, && \text{(P1g)}\\
& \hat{c}^+\le w, && \text{(P1h)}\\
& \hat{c}^-\le e-w, && \text{(P1i)}\\
& e^\top(\hat{c}^++\hat{c}^-)=1, && \text{(P1j)}\\
& \hat{x}_i\in\mathbb{R}^n,\ s_i\in\mathbb{R}^m_+,\ \lambda_i\in\mathbb{R}^m_+,\\
& \quad z_i\in\{0,1\}^m \quad \forall i\in\mathcal{I}, && \text{(P1k)}\\
& \hat{c}\in\mathbb{R}^n,\hat{c}^+\in\mathbb{R}^n_+,\hat{c}^-\in\mathbb{R}^n_+,\\
& \qquad\qquad w\in\{0,1\}^n, && \text{(P1l)}
\end{aligned}
$$

where M is a sufficiently large parameter. Constraints (P1b)–(P1e) correspond to the KKT optimality conditions of the lower-level problems in (6), (P1f) ensures that $\hat{x}_i$ is a vertex of the polyhedron $\{\tilde{x} : A_i\tilde{x}\le b_i\}$, and (P1g)–(P1j) represent a linearization of the condition $\|\hat{c}\|_1=1$, which excludes $\hat{c}=0$ from the set of feasible solutions. Note that (P1) is not an exact reformulation of (6) because the constraint $\|\hat{c}\|_1=1$ cuts off more than just the single point $\hat{c}=0$. However, the following property suggests that this restriction does not affect the solution set of (IOP).

Lemma 2. *Problems* (6) *and* (P1) *have the same set of feasible* $\hat{x}$.

An exact reformulation of (8) directly follows from the definition of polyhedral cones:

$$
\begin{aligned}
\underset{\hat{c},\,\gamma}{\text{minimize}} \quad & \|\bar{c}-\hat{c}\|_2^2\\
\text{subject to} \quad & \hat{c}=-\sum_{t\in\mathcal{T}(\hat{x}_i)}\gamma_{it}\,a_{it} \quad \forall i\in\mathcal{I},\\
& \gamma_{it}\ge 0 \quad \forall i\in\mathcal{I},\ t\in\mathcal{T}(\hat{x}_i), && \text{(P2)}
\end{aligned}
$$

where the constraints require that $\hat{c}$ can be expressed as a conic combination of $-a_{it}$, $t\in\mathcal{T}(\hat{x}_i)$, for every $i\in\mathcal{I}$.

Proposition 1. *If Condition* 1 *holds, the optimal solution to* (IOP) *can be obtained by solving* (P1), *which provides the unique optimal* $\hat{x}^*$, *and subsequently solving* (P2) *with* $\hat{x}=\hat{x}^*$.

In the general case where Condition 1 does not necessarily hold (or where we do not know in advance that it holds), the complete set $\widehat{\mathcal{X}}^*$ can be recovered by resolving (P1) with added integer cuts of the following form until the objective function value changes:

$$
\sum_{i\in\mathcal{I}}\sum_{t\in\mathcal{T}(\hat{x}_i)} z_{it}\le n\,|\mathcal{I}|-1. \tag{11}
$$

Because the set of active constraints at a particular $\hat{x}_i$ is given by the values of the binary variables z_i, we use (11) to impose the condition that for a different optimal solution to exist, at least one of the $\hat{x}_i$ has to induce a different set of active constraints. A two-phase algorithm incorporating integer cuts to obtain the complete set of optimal solutions to (IOP) as described in the proof of Theorem 2 is shown in Algorithm 1.

Algorithm 1 (Two-Phase Algorithm for Solving (IOP))

PHASE 1
solve (P1), obtain $\hat{x}^*$, $r^*\leftarrow\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}_i}\|x_{ij}-\hat{x}_i^*\|$
initialize: $k\leftarrow 1$, $r^1\leftarrow r^*$, $\hat{x}^1\leftarrow\hat{x}^*$
while $r^k=r^*$ **do**
  add integer cut (11) for $\hat{x}^k$ to (P1)
  solve (P1), obtain $\hat{x}^*$, $\hat{x}^{k+1}\leftarrow\hat{x}^*$, $r^{k+1}\leftarrow\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}_i}\|x_{ij}-\hat{x}_i^{k+1}\|$
  $k\leftarrow k+1$
end while
PHASE 2
$K\leftarrow k-1$
for all $k=1,\ldots,K$ **do**
  solve (P2) with $\hat{x}=\hat{x}^k$, obtain $\hat{c}^*$, $\hat{c}^k\leftarrow\hat{c}^*$, $v^k\leftarrow\|\bar{c}-\hat{c}^k\|_2^2$
end for
return all $\hat{c}^k$ for which $v^k=\min_{k'=1,\ldots,K}v^{k'}$

Remark 1. Problems (P1) and (P2) do not include any additional constraints on the values of c (and therefore $\hat{c}$) apart from it being a nonzero vector in (P1). In practice, it is likely that we have additional information about the true cost vector. For example, if the cost coefficients to be estimated represent costs of items, one can safely assume them to be strictly positive. Such information can be directly incorporated into (P1) and (P2) in the form of additional constraints, which can further restrict the admissible set and help obtain a reasonable cost estimate. This can be especially useful in situations where the observed data have a high variance or the vertices of the polyhedron are very close to each other. We illustrate this point through an example in Section C of the online supplement.

Remark 2. So far, we have defined the loss function—that is, the objective function of (P1)—to be the sum of a general norm of the residuals. Typically, a p-norm is used. Although the 2-norm seems to work well in most cases, ideally, the norm should be chosen based on the type of noise in the data. Specifically, if the observations are known to be prone to outliers, the 1-norm can be used, as it is known to be robust against outliers; however, if the data are highly accurate and the hypothesis of a linear objective function is being tested, one might want to use the ∞-norm to minimize the worst-case residuals.

4. A Sequential Decomposition Algorithm for (P1)

Problem (P1) is a mixed-integer linear program (MILP) or MINLP (depending on the choice of loss function) whose size increases with the number of experiments, inducing computational challenges when the data set is large. Therefore, in the following, under the assumption that Condition 1 holds everywhere, we present an exact decomposition algorithm that can substantially reduce the computation time in large instances.

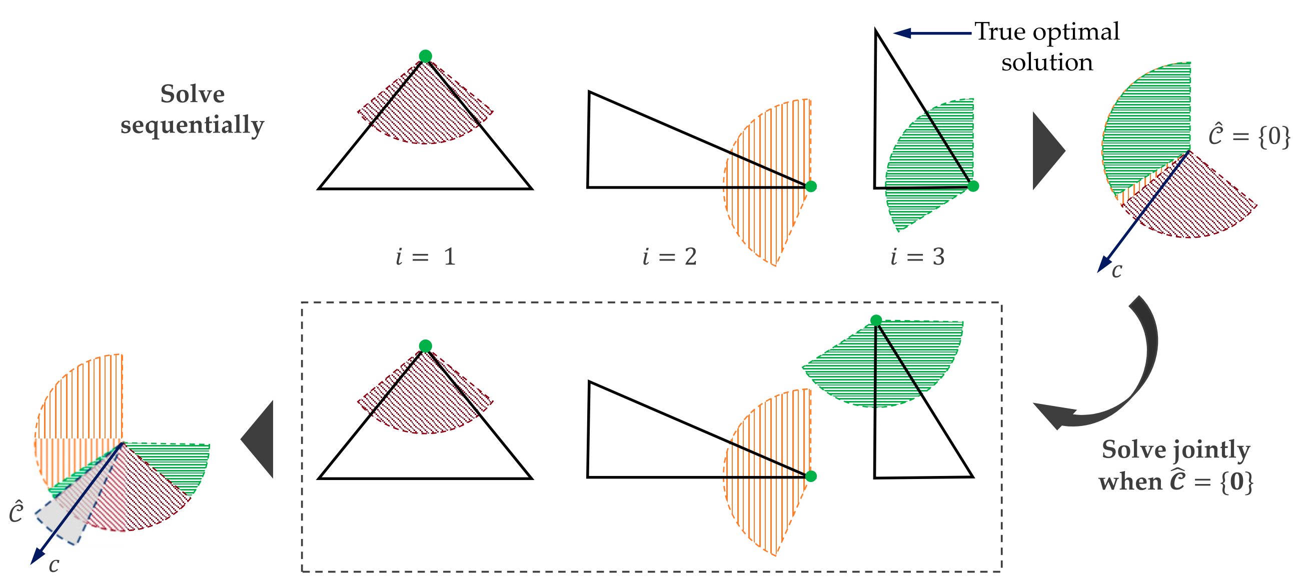
Notice that (P1) has a clear decomposable structure. Specifically, $\hat{c}$ act as linking variables such that with fixed $\hat{c}$, (P1) decomposes into $|\mathcal{I}|$ independent subproblems, one for each experiment. However, because of the nonconvex nature of the subproblems, traditional Benders-type decomposition methods cannot be directly applied to solve the problem to provable optimality. Instead, we develop a decomposition method that is more akin to Lagrangean decomposition but exploits the structure of the problem such that the exact optimal solution can be obtained. We start by presenting some properties of (P1) that form the basis of our solution algorithm.

Lemma 3. *Given a set of experiments $\{1, \ldots, N\}$, let $(\text{P1})_i$ with $i \le N$ be an instance of (P1) with $\mathcal{I} = \{i\}$, and let its optimal value be $\bar{r}_i^*$. If r^* denotes the optimal value of (P1) with $\mathcal{I} = \{1, \ldots, N\}$, then $\sum_{i=1}^N \bar{r}_i^* \le r^*$.*

Corollary 2. *Let $(\text{P1})_{[i]}$ be an instance of (P1) with $\mathcal{I} = \{1, \ldots, i\}$, and let $\bar{r}_{[i]}^*$ be its optimal value. Then, $\bar{r}_{[i-1]}^* + \bar{r}_i^* \le \bar{r}_{[i]}^*$.*

Our sequential decomposition approach for (P1) is a direct consequence of Corollary 2. We first provide an intuitive description of the method using the illustrative example shown in Figure 3, which involves three experiments. The main idea is to solve (P1) *sequentially* (i.e., one experiment followed by another experiment) instead of directly solving the full-space problem considering all experiments. With only one experiment, (P1) reduces to the problem of projecting the noisy observations onto the vertex that minimizes the loss. When the level of noise is small, it is likely that this already results in the solution that is optimal for the full problem, as is the case for experiments 1 and 2 in Figure 3. In experiment 3, however, the level of noise is so high that the projection yields the wrong vertex. A solution to the full problem involving all experiments has to provide a $\hat{c}$ that renders all $\hat{x}_i$ optimal, which implies that the resulting admissible set $\widehat{\mathcal{C}}$ must not only contain 0. As shown at the top right of Figure 3, although the true cost vector c lies in the intersection of the two cones resulting from the correct projections, the intersection of all three cones including the one associated with the incorrect projection is $\{0\}$. When $\widehat{\mathcal{C}} = \{0\}$, we solve (P1), considering all

Figure 3. (Color online) Depiction of the Proposed Decomposition Algorithm



Note. Note that only the vertex projections are shown (filled green circles); the corresponding noisy observations are omitted for the sake of clearer visualization.

experiments up to that point jointly in order to correct the projections such that the resulting $\widehat{\mathcal{C}}$ is a proper polyhedral cone (see the bottom left of Figure 3).

Detecting such infeasible projections is crucial for the proposed decomposition algorithm, and it turns out that this can be accomplished by solving a very efficient feasibility problem, which we establish in the following proposition.

Proposition 2. *Let $(\hat{x}_1, \dots, \hat{x}_{\ell-1})$ and $\hat{x}_\ell$ be feasible solutions to problems $(\text{P1})_{[\ell-1]}$ and $(\text{P1})_\ell$, respectively. Then, $(\hat{x}_1, \dots, \hat{x}_\ell)$ is a feasible solution to $(\text{P1})_{[\ell]}$ if the following problem is feasible:*

$$
\begin{aligned}
\underset{\hat{c}, \hat{c}^+, \hat{c}^-, \gamma, w}{\text{minimize}} \quad & 0 \\
\text{subject to} \quad & \hat{c} = \sum_{t \in \mathcal{T}(\hat{x}_i)} \gamma_{it}(-a_{it}) \quad \forall i \in \{1, \dots, \ell\}, \\
& \hat{c} = \hat{c}^+ - \hat{c}^-, \\
& \hat{c}^+ \le w, \\
& \hat{c}^- \le e - w, \\
& e^\top(\hat{c}^+ + \hat{c}^-) = 1, \\
& \gamma_{it} \ge 0 \quad \forall i \in \{1, \dots, \ell\}, t \in \mathcal{T}(\hat{x}_i), \\
& \hat{c} \in \mathbb{R}^n, \hat{c}^+ \in \mathbb{R}^n_+, \hat{c}^- \in \mathbb{R}^n_+, w \in \{0,1\}^n.
\end{aligned}
\qquad (\text{FP})_\ell
$$

Note that $(\text{FP})_\ell$ is a feasibility problem (as opposed to an optimization problem), and therefore, the objective function is arbitrarily set to a constant 0.

Corollary 3. *Let $(\hat{x}_1, \dots, \hat{x}_{\ell-1})$ and $\hat{x}_\ell$ be optimal solutions to problems $(\text{P1})_{[\ell-1]}$ and $(\text{P1})_\ell$, respectively. Then, $(\hat{x}_1, \dots, \hat{x}_\ell)$ is an optimal solution to $(\text{P1})_{[\ell]}$ if $(\text{FP})_\ell$ is feasible.*

Although $(\text{FP})_\ell$ is an MILP, its number of binary variables does not change with the number of data points, which allows it to remain tractable for instances with many experiments. Therefore, in the algorithm, after every experiment ℓ, we solve $(\text{FP})_\ell$ to check the validity of the partial solution obtained up to that point for problem $(\text{P1})_{[\ell]}$. If at some point $(\text{FP})_\ell$ is found infeasible, we solve the full problem $(\text{P1})_{[\ell]}$. Here, it is important to note that because the solution obtained for $(\text{P1})_{[\ell-1]}$ in the previous iteration is guaranteed to be feasible for $(\text{P1})_{[\ell]}$, it can be used to warm-start the solver, which is another crucial factor with regard to the computational efficiency of the algorithm. In practice, with a low level of noise or a sufficiently large number of samples $|\mathcal{J}_i|$ for each experiment $i \in \mathcal{I}$, one can expect the single-experiment projection to be accurate most of the time. As a result, one may only need to solve a small number of problems involving multiple experiments. The pseudocode for the complete algorithm for solving (IOP) with this sequential decomposition approach is shown in Algorithm 2.

Algorithm 2 (Decomposition-Based Algorithm for Solving (IOP))

initialize: $\widehat{\mathcal{C}} \leftarrow \mathbb{R}^n$
for all $\ell \in \{1, \dots, N\}$ **do**
  solve $(\text{P1})_\ell$, obtain $\hat{x}^*_\ell$
  $\widehat{\mathcal{C}} \leftarrow \widehat{\mathcal{C}} \bigcap \text{cone}\Big(\{-a_{\ell t}\}_{t \in \mathcal{T}(\hat{x}^*_\ell)}\Big)$
  solve $(\text{FP})_\ell$
  if $(\text{FP})_\ell$ is infeasible **then**
    solve $(\text{P1})_{[\ell]}$, warm-start with $(\hat{x}^*_1, \dots, \hat{x}^*_{\ell-1})$, obtain $\hat{x}^*_{[\ell]} = (\hat{x}^*_1, \dots, \hat{x}^*_\ell)$
    $\widehat{\mathcal{C}} \leftarrow \bigcap_{i \in \{1, \dots, \ell\}} \text{cone}\Big(\{-a_{it}\}_{t \in \mathcal{T}(\hat{x}^*_i)}\Big)$
  end if
end for
solve (P2), obtain $\hat{c}^*$

So far, we have assumed that all data are available prior to the inverse optimization process. However, in many situations, it is reasonable to expect the data from different experiments to become available at different time points. This has motivated the development of online learning frameworks for inverse optimization where the cost estimate is updated as new data arrive (Bärmann et al. 2018, Dong et al. 2018, Shahmoradi and Lee 2021). A naive extension of our two-phase framework for online learning can be to repeatedly solve (IOP) with the new data added to it. However, note that Algorithm 2 provides a much more efficient mechanism to address this problem and can be adapted as a method for inverse optimization in an online setting. We include such an adaptation in Section B of the online supplement.

5. Online Adaptive Sampling

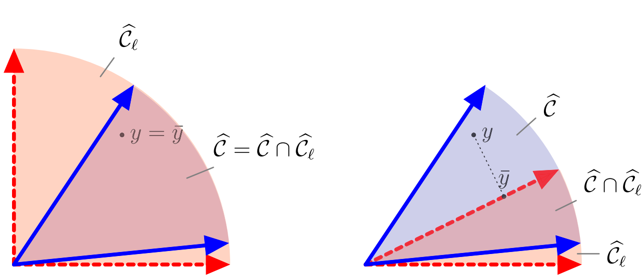
In this section, we introduce a novel adaptive sampling approach that can be used to reduce the number of experiments required to obtain a good cost estimate in an online setting. Our method derives from the geometrical understanding of (IOP) discussed in the previous sections. As a motivating example, let us revisit the case with three experiments shown in Figure 3. Notice that the admissible set $\widehat{\mathcal{C}}$ obtained with the first two experiments is the same as the one obtained with all three experiments. The reason is that the cone associated with the projected solution from experiment 3 is a superset of the cone from experiment 1; hence, the intersection of cones does not change with the third experiment. This means that experiment 3 cannot help improve the cost estimate or, more precisely, our confidence in the cost estimate, which only increases if the size of $\widehat{\mathcal{C}}$ decreases. Therefore, the goal here is to use the current best cost estimate to design subsequent experiments in a way such that the size of $\widehat{\mathcal{C}}$ will likely be further reduced. In the following, we formulate the adaptive sampling problem as an optimization problem, and devise an efficient heuristic method to solve it.

5.1. Mathematical Formulation

Consider the point in an online inverse optimization process at which $\ell - 1$ experiments are completed, and the problem is to choose input parameters $(A_\ell, b_\ell) \in \Pi$ for the ℓth experiment. Let $\mathcal{I}$ be the set of the first $\ell - 1$ experiments, and let $\widehat{\mathcal{C}}$ be the admissible set obtained using these experiments. Suppose $\widehat{\mathcal{C}}_\ell$ denotes the (unknown) cone formed by the active constraints of the FOP with its feasible region represented by the polyhedron $\{\tilde{x} : A_\ell \tilde{x} \le b_\ell\}$. The goal as illustrated in the aforementioned example is to choose A_ℓ and b_ℓ such that $\widehat{\mathcal{C}} \cap \widehat{\mathcal{C}}_\ell \subset \widehat{\mathcal{C}}$. Although it is impossible to determine $\widehat{\mathcal{C}}_\ell$ without knowing c exactly, we make use of a randomly sampled vector $\tilde{c}_{\ell-1}$ from $\widehat{\mathcal{C}}$ to predict which constraints may become active for given sets of potential input parameters.

Assuming we have an accurate estimate of $\widehat{\mathcal{C}}_\ell$, we utilize the fact that $\widehat{\mathcal{C}} \cap \widehat{\mathcal{C}}_\ell \subset \widehat{\mathcal{C}}$ if and only if we can find a point that belongs to $\widehat{\mathcal{C}}$ but not to $\widehat{\mathcal{C}}_\ell$. As we illustrate in Figure 4, the existence of such a point can be determined by obtaining the minimum-distance projections of all points in $\widehat{\mathcal{C}}$ on $\widehat{\mathcal{C}}_\ell$ and searching for one that yields a nonzero projection distance. In the figure, $\bar{y}$ denotes the minimum-distance projection of a point y from $\widehat{\mathcal{C}}$ on $\widehat{\mathcal{C}}_\ell$. In the case of Figure 4(a), all points $y \in \widehat{\mathcal{C}}$ are such that $\bar{y} = y$, and hence, the intersection of the two cones does not result in a reduction in the size of the admissible set. In Figure 4(b), however, one can find a y and its corresponding $\bar{y}$ with $\|y - \bar{y}\| > 0$, and therefore, the intersection of the two cones is a strict subset of $\widehat{\mathcal{C}}$.

Figure 4. (Color online) The Cone Formed by the Blue Solid Arrows Represents $\hat{\mathcal{C}}$, Whereas the Red Dashed Arrows Form the Cone $\hat{\mathcal{C}}_\ell$



(a) $\widehat{\mathcal{C}} = \widehat{\mathcal{C}} \cap \widehat{\mathcal{C}}_\ell$ **(b)** $\widehat{\mathcal{C}} \cap \widehat{\mathcal{C}}_\ell \subset \widehat{\mathcal{C}}$

Note. The dotted line in the second case shows the minimum 2-norm projection of y on $\hat{\mathcal{C}}_\ell$.

Given $\widehat{\mathcal{C}}$ and $\widehat{\mathcal{C}}_\ell$, we find the $y \in \widehat{\mathcal{C}}$ with the maximum projection distance to $\widehat{\mathcal{C}}_\ell$ and use this distance as a measure for the difference between $\widehat{\mathcal{C}}$ and $\widehat{\mathcal{C}} \cap \widehat{\mathcal{C}}_\ell$. Hence, as we wish to reduce the size of the admissible set as much as possible, we need to choose A_ℓ and b_ℓ for which this distance is the largest among all $(A_\ell, b_\ell) \in \Pi$. We formulate the problem of finding such A_ℓ and b_ℓ as the following optimization problem:

$$\begin{aligned}
\underset{y,\gamma,z,A_\ell,b_\ell,\hat{x}_\ell}{\text{maximize}} \quad & \min_{\eta,\bar{y},\bar{\gamma}} \Big\{ \eta : \eta = \|y - \bar{y}\|_1, \\
& \quad \bar{y} = \sum_{k \in \mathcal{K}} \bar{\gamma}_k z_k a_{\ell k},\ \bar{\gamma}_k \ge 0 \ \ \forall k \in \mathcal{K} \Big\} && \text{(ASPa)} \\
\text{subject to} \quad & y = \sum_{t \in \mathcal{T}(\hat{x}_i)} \gamma_{it} \left(-\frac{a_{it}}{\|a_{it}\|_2} \right) \quad \forall i \in \mathcal{I}, && \text{(ASPb)} \\
& 0 \le \gamma_{it} \le \epsilon \quad \forall i \in \mathcal{I}, t \in \mathcal{T}(\hat{x}_i), && \text{(ASPc)} \\
& (A_\ell, b_\ell) \in \Pi, && \text{(ASPd)} \\
& z_k = \begin{cases} 1 & \text{if } k \in \mathcal{T}(\hat{x}_\ell), \text{ where } \hat{x}_\ell \in \arg\min_{\tilde{x} \in \mathbb{R}^n} \\ & \quad \{\tilde{c}_{\ell-1}^\top \tilde{x} : A_\ell \tilde{x} \le b_\ell\} \\ 0 & \text{otherwise} \end{cases} \quad \forall k \in \mathcal{K}, && \text{(ASPe)}
\end{aligned}$$

where $\mathcal{K}$ is the set of constraints for $A_\ell x \le b_\ell$. Constraints (ASPb) state that y is a point in $\widehat{\mathcal{C}}$. Note that the vectors forming cone $\widehat{\mathcal{C}}_i$ for each $i \in \mathcal{I}$ have been normalized to ensure that the lengths of the vectors do not bias the value of η. The upper bound ϵ in (ASPc) ensures that the problem remains bounded. The value of ϵ can be chosen arbitrarily and does not affect the choice of optimal A_ℓ and b_ℓ for the problem. Constraint (ASPd) imposes the restriction that input parameters A_ℓ and b_ℓ must be chosen from the set Π. As stated in constraints (ASPe), z_k is a binary variable that equals 1 if $\tilde{c}_{\ell-1}$ predicts that the kth constraint will be active. Finally, the objective of (ASP) is to maximize the minimum 1-norm distance between y and $\bar{y}$, with $\bar{y}$ being constrained to lie in the cone $\widehat{\mathcal{C}}_\ell$. We denote the distance between y and $\bar{y}$ by η.

5.2. A Heuristic Solution Algorithm for (ASP)

Problem (ASP) is a bilevel optimization problem with lower-level problems embedded in the objective function (ASPa) and constraints (ASPe). Because these embedded problems are LPs, (ASP) can be reformulated into a single-level problem, similar to the bilevel problems considered in previous sections. We present two such single-level reformulations in Section D of the online supplement. Both of them take the forms of nonconvex MINLPs that are only tractable for relatively small problems. Next, we outline a heuristic approach to solve this problem.

Instead of searching through the entire set Π for the best set of input parameters, the proposed algorithm, Algorithm 3, simplifies the problem by limiting the search space to S randomly sampled sets of input parameters. It then solves S instances of (ASP), each corresponding to one of the parameter sets, to choose the one that results in the largest η. Here, because the number of possible candidates for A_ℓ and b_ℓ is finite with their values being explicitly known, the binary vector z for each of them can be determined before (ASP) is solved. This eliminates the lower-level problems in (ASPe). Furthermore, in this algorithm, because the S instances of (ASP) are independent of each other, they can be solved in parallel. This enables the use of a large S to find a heuristic solution as close to the global optimum of (ASP) as possible while still keeping the problem tractable.

Algorithm 3 (A Heuristic Algorithm for Solving (ASP))

1: **for** $i \in \{1, \dots, S\}$ **do**
2: sample $(A_i, b_i) \in \Pi$
3: compute z_i such that

$$z_{ik} = \begin{cases} 1 & \text{if } k \in \mathcal{T}(\hat{x}_i), \text{ where } \hat{x}_i \in \arg\min_{\tilde{x} \in \mathbb{R}^n} \{\tilde{c}_{\ell-1}^\top \tilde{x} : A_i \tilde{x} \le b_i\} \\ 0 & \text{otherwise} \end{cases}$$

4: solve (ASP) with $A_\ell = A_i$, $b_\ell = b_i$, and $z = z_i$, obtain η_i
5: **end for**
6: **return** A_i, b_i for which $\eta_i = \max_{i' = \{1, \dots, S\}} \eta_{i'}$

Remark 3. Problem (ASP) is valid regardless of the restrictions on the input parameters A and b (i.e., the set Π). However, there is a difference in the way the size of $\widehat{\mathcal{C}}$ is reduced when only right-hand-side parameters b are allowed to be varied compared with when the constraint matrix A can also be varied. We highlight this difference in detail in Section E of the online supplement.

6. Computational Case Studies

In this section, we apply the proposed data-driven inverse linear optimization framework to two case studies, one addressing customer preference learning and the other related to cost estimation for multiperiod production planning. Using synthetic data, we compare the computational performances of Algorithms 1 and 2 and evaluate the impact of adaptive sampling. All model instances were implemented in Julia v1.3.0 (Bezanson et al. 2017) using the modeling language JuMP v0.18.6 (Dunning et al. 2017). All instances of (P1) were solved with a 1-norm-based objective function, and Gurobi v9.0.2 (Gurobi Optimization 2020) was applied to solve the resulting MILPs. Nonconvex MINLPs were solved using BARON v19.12.7 (Sahinidis 1996). The Julia code for all computational experiments presented in this section is available from the IJOC GitHub software repository (https://github.com/INFORMSJoC/2020.0231).

6.1. Customer Preference Learning

We consider the problem of learning customers' preferences given their purchasing decisions. The FOP here is based on the premise that with a limited budget, customers will buy products that maximize their utility. Given the price w_p of each product p and a budget b, the customer is assumed to solve the following LP:

$$\begin{aligned} \underset{x \in \mathbb{R}^n_+}{\text{maximize}} \quad & \sum_{p \in \mathcal{P}} u_p x_p, \\ \text{subject to} \quad & \sum_{p \in \mathcal{P}} w_p x_p \le b, \\ & x_p \le 1 \quad \forall p \in \mathcal{P}, \end{aligned} \tag{12}$$

where $\mathcal{P} = \{1, \dots, n\}$ denotes the set of n products available on the market. The goal is to estimate the unknown utility function coefficients u by observing the changes in the customer's decisions x in response to price fluctuations. Different variations of this IOP have previously been considered by Bärmann et al. (2018) and Dong et al. (2018). Bärmann et al. (2018) consider learning the utility function with deterministic data. Dong et al. (2018) account for noise in the data but pose (12) with a strongly concave utility function. Our design of this case study closely follows the scheme presented by Bärmann et al. (2018) but uses noisy data to learn the utility function coefficients.

For each instance of the IOP, the training data are generated as follows. We first create an arbitrary utility vector $u \in \mathbb{R}^n$ by sampling its individual elements from the uniform distribution $\mathcal{U}(1, 1000)$ and normalize it to make its 1-norm equal to 1. We then sample a set of price vectors w_i, which are the input parameters for each $i \in \mathcal{I}$, such that $w_{ip} \sim \mathcal{U}(50, 150)$ for every $p \in \mathcal{P}$. The budget b is set to $0.6 \sum_{p \in \mathcal{P}} w_{1p}$ for all experiments. Next, keeping the utility vector the same, we solve these $|\mathcal{I}|$ instances of (12) to obtain the optimal decisions x_i^*. We then generate the noisy data sets $\mathcal{J}_i$ for each $i \in \mathcal{I}$ by distorting the true optimal solution such that $x_{ij} = x_i^* + \gamma$, where $\gamma \sim \mathcal{N}(0, \sigma^2 \mathbb{I})$.

In this study, we consider FOPs of varying dimensionality n but limit the number of experiments to 100 in all cases. We also consider data sets with varying levels of noise by changing the value of σ. Once n and σ are fixed, the size of the sets $\mathcal{J}_i$ is kept the same for all $i \in \mathcal{I}$ (i.e., $|\mathcal{J}_i| = J$ for all $i \in \mathcal{I}$). A specific case is hence represented by n, σ, and J, and we solve 10 random instances of each case (generated using the scheme described in the preceding). Following Remark 1, to increase the robustness of our two-phase algorithm against such a large number of polyhedral geometries and different levels of noise,

Table 1. Comparison of Computational Performances of Algorithms 1 and 2 on an IOP Based on Random Instances of (12)

			Algorithm 1				Algorithm 2				
			No. of instances solved	Computation time (s)			No. of instances solved	Computation time (s)			Median no. of resolves
n	σ	J		Median	Max	Min		Median	Max	Min	
25	0.01	5	10	33	155	6	10	277	317	253	2
		250	10	73	90	67	10	266	271	264	0
	0.05	10	10	176	264	54	10	283	789	262	6
		250	10	88	965	74	10	268	347	264	0
	0.1	20	10	306	680	22	10	315	499	273	9.5
		250	10	180	5,563	79	10	298	687	265	1
50	0.01	5	10	88	931	7	10	484	638	331	1.5
		250	10	220	1,724	200	10	423	717	414	0
	0.05	10	10	1,037	5,874	430	10	673	6,897	409	7.5
		250	7	314	439	215	10	626	1,617	417	1.5
	0.1	20	8	2,707	7,081	715	10	817	1,841	504	11.5
		250	8	875	6,035	265	10	705	1,651	510	2
100	0.01	5	9	407	488	29	10	428	1,042	362	2
		250	10	489	2,267	419	10	479	1,648	451	0
	0.05	10	3	2,707	2,851	1,902	9	1,021	2,032	586	9
		250	9	2,232	6,610	526	10	865	5,537	455	1.5
	0.1	20	0	n/a	n/a	n/a	8	1,656	2,933	935	12
		250	4	1,089	2,561	542	8	1,337	6,890	565	4.5

Notes. Reported computation times only consider the instances that were solved to optimality. Computation times are indicated as not available (n/a) if none of the corresponding instances were solved to optimality.

we introduce an additional set of constraints in (P1) enforcing $\hat{u}_p \geq 10^{-6}$ for all $p \in \mathcal{P}$.

6.1.1. Computational Performance. The results comparing the computational performance of Algorithms 1 and 2 are summarized in Table 1. All instances were solved with a time limit of 7,200 s utilizing 24 cores on the Mesabi cluster at the Minnesota Supercomputing Institute (MSI). For each of the algorithms, the table lists the number of instances (out of 10) that were solved to optimality. In all instances that could not be solved with Algorithm 1, we find that the solver could not find even a single feasible solution within the given time; hence the optimality gaps are not reported. For the decomposition-based algorithm, an unsolved instance is one where not all 100 experiments could be processed in the given time, but it still yields an estimate for the cost coefficients. Nonetheless, for the comparison of the two solution methods, we consider a run resulting in a partial solution as a failed run. Table 1 shows the median, maximum, and minimum computation times for both algorithms. For Algorithm 1, it is the time required to solve an instance without implementing the integer cuts in Phase 1. We find that adding the integer cuts to identify multiple optimal solutions was only required in 2 of the 180 (~ 1%) instances considered in this study, confirming our assertion that a violation of Condition 1 is rare. Finally, note that Algorithm 2 has an additional column labeled "Median no. of re-solves" that shows the median value of the number of times the feasibility problem $(\text{FP})_\ell$ was found infeasible, which then required the problem involving all experiments up to that point to be resolved.

From the data in Table 1, one can observe that the two algorithms solve the IOP in comparable times when the level of noise is low. However, as the size of the problem or the level of noise increases, Algorithm 2 starts outperforming Algorithm 1. The difference in their performance is especially apparent from the numbers of instances solved and the maximum computation times. For example, in the arguably most difficult case with $n = 100$, $\sigma = 0.1$, and $J = 20$, Algorithm 1 was not able to solve any of the given instances, whereas Algorithm 2 solved 8 out of 10. Irrespective of the solution algorithm, increasing J has a seemingly counterintuitive effect of making the problem easier to solve. This is because increasing the number of samples for an experiment merely increases the number of terms in the objective function of (P1) while making it easier for the problem to find the correct vertex onto which to project. For Algorithm 2, one can also see that increasing J reduces the likelihood of incorrect projections when processing the data for individual experiments separately. This is especially helpful in situations where computing infrastructure is a limitation, as a large amount of data has a significantly higher memory requirement. Overall, the results indicate that Algorithm 2 dominates Algorithm 1 when it comes to solving more difficult instances of (IOP) with data of high dimensionality and level of noise.

6.1.2. Prediction Error. Phase 2 of our two-phase algorithm requires a reference $\bar{u}$ to yield an estimate $\hat{u}$. Ideally, this reference is based on some prior intuition about the unknown utility function, but here, we obtain $\hat{u}$ using a randomly generated $\bar{u}$. The goal is to test the capability of this estimate in generating reliable predictions on unseen data sets in the worst-case scenario where no prior information about the missing parameters is available. To perform this assessment, along with every instance of training data, we also generate a test data set of 100 experiments. These test data consist of (w, x^*) pairs, where w-values are generated in the same manner as for the training data, and x^* are the corresponding true optimal solutions obtained using the same u as the one used to generate the noisy training data. Once a $\hat{u}$ has been found, we use it to solve the problems in the test data set and evaluate the prediction error as the fraction of incorrect predictions.

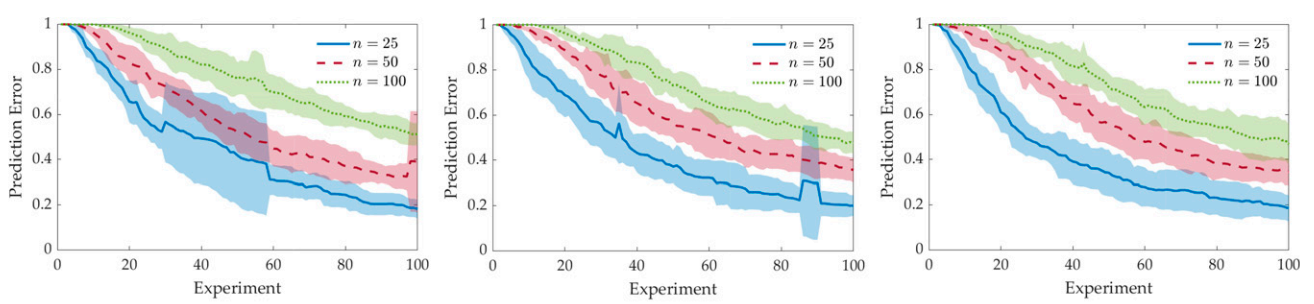
We show the prediction error results for a few selected cases in Figure 5. Here, instead of using just the final estimate obtained with all the $|\mathcal{I}|$ experiments, we make use of the online algorithm (presented in Section B of the online supplement) to show how the prediction error evolves with the addition of new experiments to the training set. As expected, the prediction error generally decreases with the number of experiments, and problems of higher dimensionality require more experiments to reach the same prediction accuracy. Also, the panels show an additional benefit of a large J apart from making (P1) easier to solve. In Figure 5, (a) and (b), where J is kept small, the curves are "spiky," showing a local increase in prediction error. Recall that we solve $(\mathrm{FP})_\ell$ after every experiment ℓ to confirm if the solution obtained with the first ℓ experiments ($\ell \le |\mathcal{I}|$) still holds for $(\mathrm{P1})_{[\ell]}$. Therefore, these local spikes are a consequence of insufficient sampling, which results in the violation of our primary assumption that the vertex with minimum loss on the given data is the "correct" vertex. As seen in Figure 5(c), once an adequate number of samples are used, the likelihood of observing these peaks reduces significantly. We expect that in situations where the quality of data is uncertain and sampling is limited, adding a preprocessing step to remove large outliers from the data set can be a way to prevent the model from making wrong estimates.

6.1.3. Adaptive Sampling. We also apply our adaptive sampling strategy to this example. Following the data generation scheme used for random sampling, we define the set Π as allowing any w for which $50 \le w_p \le 150$ for any $p \in \mathcal{P}$ while keeping the other constraint parameters fixed. Irrespective of the size of (IOP), we find that BARON struggles to find even a feasible solution for (ASP) when we try to solve it exactly. Therefore, we solve it here using the heuristic solution algorithm, Algorithm 3. All the S subproblems were solved with a time limit of 100 s. Although BARON is unable to solve the subproblems to optimality, it still finds feasible solutions that show good potential in reducing the size of the admissible set. The significant impact of our adaptive sampling strategy is shown in Figure 6. In all three cases, the use of adaptive sampling results in a more than 50% reduction in the number of experiments required to achieve the same prediction accuracy as obtained from the standard approach using random sampling after 100 experiments. Furthermore, Figure 6(a) shows the effect of the parameter S in the heuristic solution approach for (ASP). One can see that even a small S results in a large increase in the rate of reduction of the prediction error and moreover, the rate stabilizes very quickly for a rather small S. By performing a similar study on higher-dimensional problems, we find that using an S equal to the dimensionality of the problem n to be a good heuristic to achieve the desired effect with adaptive sampling.

6.2. Cost Estimation for Production Planning

In our second case study, we consider the problem of production planning for a large manufacturing site consisting of multiple processes within an interconnected

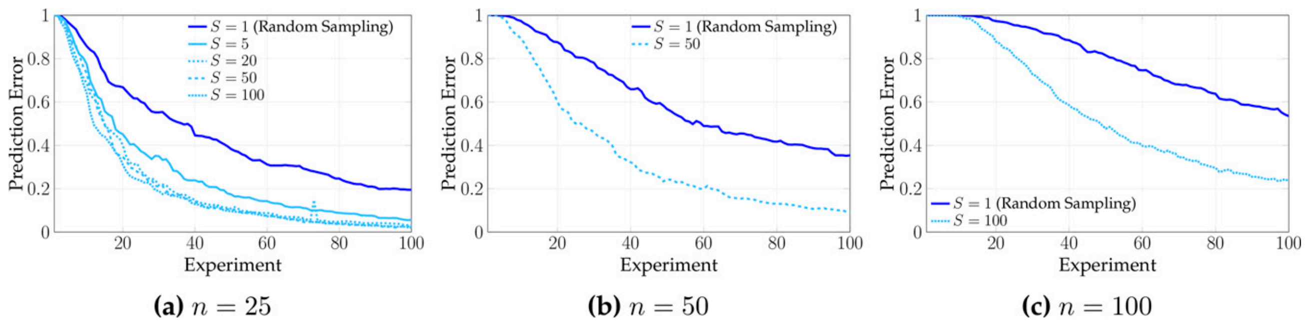
Figure 5. (Color online) Change in Prediction Error as Experiments Are Added



(a) $\sigma = 0.01, J = 5$ **(b)** $\sigma = 0.1, J = 20$ **(c)** $\sigma = 0.1, J = 250$

Notes. Lines show mean values across all solved instances; shaded areas indicate one standard deviation around the mean. For all the cases, prediction error data for individual instances are available in Section F of the online supplement.

Figure 6. (Color online) Effect of Adaptive Sampling on the Evolution of Prediction Error



Notes. Lines show mean values across 10 instances. Training data for all cases were generated using $\sigma = 0.01, J = 30$.

process network. It is commonly formulated as an LP:

$$\underset{x,y,w}{\text{minimize}} \quad \sum_{h \in \mathcal{H}} \sum_{m \in \mathcal{M}} \left(\sum_{p \in \mathcal{P}} c_{pmh} x_{pmh} + f_{mh} w_{mh} \right) \tag{13a}$$

$$\text{subject to} \quad q_m^{\min} \le q_m^0 + \sum_{h'=1}^{h} \left(\sum_{p \in \mathcal{P}} x_{pmh'} + w_{mh'} - d_{mh'} \right)$$

$$\le q_m^{\max} \quad \forall m \in \mathcal{M}, \quad h \in \mathcal{H}, \tag{13b}$$

$$0 \le w_{mh} \le w_{mh}^{\max} \quad \forall m \in \mathcal{M}, h \in \mathcal{H}, \tag{13c}$$

$$x_{pmh} = \mu_{pm} y_{ph} \quad \forall p \in \mathcal{P}, m \in \mathcal{M}, h \in \mathcal{H}, \tag{13d}$$

$$0 \le x_{pmh} \le x_{pmh}^{\max} \quad \forall p \in \mathcal{P}, m \in \mathcal{M}, h \in \mathcal{H}, \tag{13e}$$

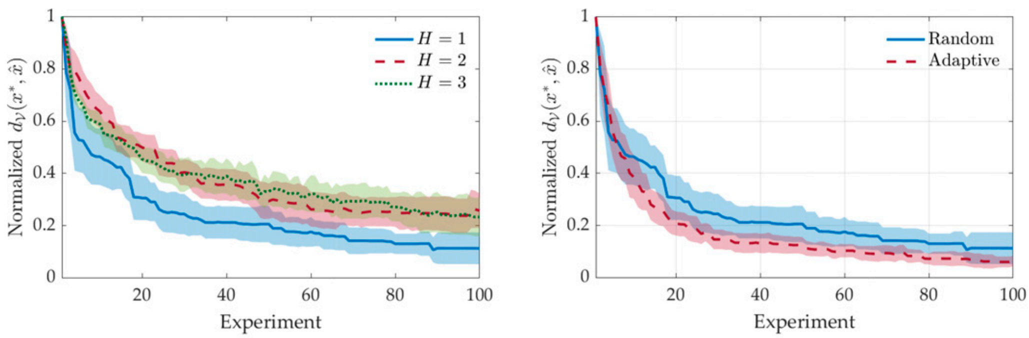
where $\mathcal{P}, \mathcal{M}$, and $\mathcal{H} = \{1, \ldots, H\}$ are the sets of processes, materials, and time periods, respectively. The amount of material m produced or consumed (depending on the sign) by process p in time period h is denoted by x_{pmh}. Product demand and the additional purchase of a material are denoted by d_{mh} and w_{mh}, respectively. Inventory constraints and restrictions on the amounts purchased are stated in constraints (13b) and (13c), respectively. The structure of the process network is defined by constraints (13d), where y_{ph} denotes the amount of a reference material for process p produced in time period h, and μ_{pm} is a conversion factor that specifies how much of a material m is produced or consumed for one unit of the reference material. According to (13a), the objective is to minimize the total production and purchasing cost while satisfying given product demand. Whereas purchasing prices f_{mh} are readily known, production costs c_{pmh} are often difficult to estimate, which is one major reason why, in practice, production planning is still mostly performed manually by human planners (Troutt et al. 2006). Experienced planners have an excellent intuition for the relative differences in costs, but this information is not explicitly expressed in numbers. The goal is to use past production plans, which reflect the planners' decisions, to infer these costs.

Here, we conduct a case study using the process network for a petrochemical site with 28 chemicals and 38 processes from Sahinidis et al. (1989) and Zhang et al. (2016). The parameters required to model this network are given in Section G of the online supplement. We test our methodology on this problem by generating synthetic training data simulating expert planners' decision making under different operating and market conditions. Here, each data point consists of inputs (μ, d) and the corresponding decisions (x, y, w). For different scenarios, the conversion factors μ_{pm} have been assumed to vary between 75% and 100% of their nominal values on account of changing efficiencies of individual processes. Also, the product demand d_{mh} is considered to fluctuate $\pm 10\%$ from its nominal value. For each instance, we generate 100 such scenarios (i.e., $|\mathcal{I}| = 100$). For each of these scenarios, we then generate the respective decisions by solving (13) with the same arbitrarily generated c_{pmh} and f_{mh} values. Because human decision making is often inconsistent, we distort the true optimal solution (x^*, y^*, w^*) as $(x^* + \gamma_1, y^* + \gamma_2, w^* + \gamma_3)$, where $\gamma_i \sim \mathcal{N}(0, \sigma^2)$ for all $i \in \{1, 2, 3\}$.

In this case study, we consider training data instances of three different sizes by varying the number of time periods H. All training data are generated using $\sigma = 3$ and $J = 30$. Each data set is used to solve (IOP) with both Algorithm 1 and the decomposition-based Algorithm 2. The problem instances were solved with a time limit of 14,400 s using 24 cores on the Mesabi cluster at the MSI. We find that whereas Algorithm 1 is unable to find even a feasible solution for any of the three cases, Algorithm 2 can solve these problems in less than 10 minutes.

Unlike the previous case study where we assessed the quality of only a point estimate obtained from Phase 2, here we instead focus on the quality of the set $\widehat{\mathcal{C}}$. To do this, we sample 10 point estimates by solving (P2) with 10 random reference cost vectors. We again

Figure 7. (Color online) Normalized $d_{\mathcal{V}}(x^*, \hat{x})$ as Experiments Are Added to the Training Set



(a) Random sampling for all cases.

(b) Comparison between random and adaptive sampling for $H = 1$. For adaptive sampling, S was set to 100.

Notes. Lines show mean values across all instances; shaded areas indicate one standard deviation around the mean. Training data were generated for $\sigma = 3, J = 30$.

consider a test data set of 100 data points consisting of arbitrary (μ, d) and (x^*, y^*, w^*) pairs. We relax the prediction error criteria to obtain a more realistic metric that measures the closeness of the generated predictions to the true optimal solutions; the metric is defined as $d_{\mathcal{V}}(x^*, \hat{x}) = \sum_{v \in \mathcal{V}} \|x^*_v - \hat{x}_v\|_\infty$, where $\mathcal{V}$ is the set of test data points. Figure 7(a) shows the change in the normalized distance metric with the addition of new experiments. One can verify that the spread around the mean value after 100 experiments is fairly small, implying high confidence in the final point estimate irrespective of the quality of the reference. However, this does not discount the importance of a good reference, as the admissible set also likely contains the true c, which, if used as a reference, would result in a zero $d_{\mathcal{V}}$ even with a single experiment.

Notice that the curve representing the mean value decreases in discrete steps with several flat regions in between two steps. This is a consequence of not all inputs resulting in a reduction in the size of the admissible set. Therefore, we also evaluate the impact of adaptive sampling on mitigating this issue. Because of the larger size of the FOP here compared with (12), we solve the subproblems in Algorithm 3 with an increased time limit of 200 s. However, even with the increased time limit, the solver struggles to find good feasible solutions for the larger instances with $H = 2$ (132 variables) and $H = 3$ (198 variables). Here, we show the effect of adaptive sampling with the case of $H = 1$ in Figure 7(b). As can be observed, adaptive sampling sustains a high rate of decrease in the distance metric for a longer duration compared with naive random sampling. This results in just ~ 52 experiments being required to achieve the same effect as 100 random inputs. Moreover, one can see that the variance around the mean in the case of adaptive sampling is noticeably lower, which shows that in addition to requiring fewer experiments, it also finds estimates with higher confidence levels.

7. Conclusions

In this work, we addressed data-driven inverse linear optimization with noisy observations, for which we introduced a new problem formulation that offers two practical advantages over other existing methods: (i) It allows the recovery of a less restrictive and generally more appropriate admissible set of cost estimates by assuming that the optimal solutions of the FOP lie at the vertices of the feasible region. (ii) Instead of randomly choosing a point estimate from the admissible set, it makes use of a reference cost vector to choose the cost estimate that most resembles the user's prior belief.

An exact two-phase algorithm was developed to solve the IOP, and we further proposed an efficient decomposition algorithm and an adaptive sampling method that are especially suited for an online inverse optimization setting. Results from extensive computational experiments based on two case studies show that the proposed methods are effective in significantly reducing both the computation time and data requirement for generating cost estimates with a reasonably low prediction error on unseen data sets.

Acknowledgments

The authors gratefully acknowledge the Minnesota Supercomputing Institute at the University of Minnesota for providing resources that contributed to the research results reported in this paper.

References

Ahuja RK, Orlin JB (2001) Inverse optimization. *Oper. Res.* 49(5): 771–783.

Arechavaleta G, Laumond JP, Hicheur H, Berthoz A (2008) An optimality principle governing human walking. *IEEE Trans. Robotics* 24(1):5–14.

Aswani A, Shen ZJM, Siddiq A (2018) Inverse optimization with noisy data. *Oper. Res.* 66(3):870–892.

Babier A, Chan TCY, Lee T, Mahmood R, Terekhov D (2021) An ensemble learning framework for model fitting and evaluation in inverse linear optimization. *INFORMS J. Optim.* 3(2):119–138.

Bärmann A, Martin A, Pokutta S, Schneider O (2018) An online-learning approach to inverse optimization. Preprint, submitted October 30, https://arxiv.org/abs/1810.12997.

Beil DR, Wein LM (2003) An inverse-optimization-based auction mechanism to support a multiattribute RFQ process. *Management Sci.* 49(11):1529–1545.

Bertsimas D, Gupta V, Paschalidis IC (2012) Inverse optimization: A new perspective on the Black-Litterman model. *Oper. Res.* 60(6): 1389–1403.

Bertsimas D, Gupta V, Paschalidis IC (2015) Data-driven estimation in equilibrium using inverse optimization. *Math. Programming* 153(2):595–633.

Bezanson J, Edelman A, Karpinski S, Shah VB (2017) Julia: A fresh approach to numerical computing. *SIAM Rev.* 59(1):65–98.

Birge JR, Hortaçsu A, Pavlin JM (2017) Inverse optimization for the recovery of market structure from market outcomes: An application to the MISO electricity market. *Oper. Res.* 65(4):837–855.

Bulut A, Ralphs TK (2021) On the complexity of inverse mixed integer linear optimization. *J. Optim.* (SIAM) 31(4):3014–3043.

Burgard AP, Maranas CD (2003) Optimization-based framework for inferring and testing hypothesized metabolic objective functions. *Biotechnol. Bioengrg.* 82(6):670–677.

Burton D, Toint PL (1992) On an instance of the inverse shortest paths problem. *Math. Programming* 53(1–3):45–61.

Chan TC, Lee T (2018) Trade-off preservation in inverse multi-objective convex optimization. *Eur. J. Oper. Res.* 270(1):25–39.

Chan TC, Lee T, Terekhov D (2019) Inverse optimization: Closed-form solutions, geometry, and goodness of fit. *Management Sci.* 65(3):1115–1135.

Chan TC, Craig T, Lee T, Sharpe MB (2014) Generalized inverse multiobjective optimization with application to cancer therapy. *Oper. Res.* 62(3):680–695.

Chang HS, Fu MC, Hu J, Marcus SI (2005) An adaptive sampling algorithm for solving Markov decision processes. *Oper. Res.* 53(1): 126–139.

Chow JY, Ritchie SG, Jeong K (2014) Nonlinear inverse optimization for parameter estimation of commodity-vehicle-decoupled freight assignment. *Transportation Res. Part E Logist. Transportation Rev.* 67(July):71–91.

Cozad A, Sahinidis NV, Miller DC (2014) Learning surrogate models for simulation-based optimization. *AIChE J.* 60(6):2211–2227.

Domingo C, Gavaldà R, Watanabe O (2002) Adaptive sampling methods for scaling up knowledge discovery algorithms. *Data Mining Knowledge Discovery* 6(2):131–152.

Dong C, Chen Y, Zeng B (2018) Generalized inverse optimization through online learning. Bengio S, Wallach H, Larochelle H, Grauman K, Cesa-Bianchi N, Garnett R, eds. *Advances in Neural Information Processing Systems*, Vol. 31 (Curran Associates, Red Hook, NY), 86–95.

Dunning I, Huchette J, Lubin M (2017) JuMP: A modeling language for mathematical optimization. *SIAM Rev.* 59(2):295–320.

Gupta R, Zhang Q (2021) Decomposition and Adaptive Sampling for Data-Driven Inverse Linear Optimization version v2020.0231. DOI: https://doi.org/10.5281/zenodo.5710020. https://github.com/INFORMSJoC/2020.0231.

Gurobi Optimization (2020) Gurobi optimizer reference manual.

Hempel AB, Goulart PJ, Lygeros J (2015) Inverse parametric optimization with an application to hybrid system control. *IEEE Trans. Automatic Control* 60(4):1064–1069.

Heuberger C (2004) Inverse combinatorial optimization: A survey on problems, methods, and results. *J. Combin. Optim.* 8(3):329–361.

Iyengar G, Kang W (2005) Inverse conic programming with applications. *Oper. Res. Lett.* 33(3):319–330.

Keshavarz A, Wang Y, Boyd S (2011) Imputing a convex objective function. *Proc. IEEE Internat. Sympos. Intelligent Control* (IEEE, Piscataway, NJ), 613–619.

Liu L, Zhang J (2006) Inverse maximum flow problems under the weighted Hamming distance. *J. Combin. Optim.* 12(4):394–407.

Mohajerin Esfahani P, Shafieezadeh-Abadeh S, Hanasusanto GA, Kuhn D (2018) Data-driven inverse optimization with imperfect information. *Math. Programming* 167(1):191–234.

Saez-Gallego J, Morales JM (2018) Short-term forecasting of price-responsive loads using inverse optimization. *IEEE Trans. Smart Grid* 9(5):4805–4814.

Saez-Gallego J, Morales JM, Zugno M, Madsen H (2016) A data-driven bidding model for a cluster of price-responsive consumers of electricity. *IEEE Trans. Power Systems* 31(6):5001–5011.

Sahinidis NV (1996) BARON: A general purpose global optimization software package. *J. Global Optim.* 8(2):201–205.

Sahinidis NV, Grossmann IE, Fornari RE, Chathrathi M (1989) Optimization model for long range planning in the chemical industry. *Comput. Chemical Engrg.* 13(9):1049–1063.

Schaefer AJ (2009) Inverse integer programming. *Optim. Lett.* 3(4): 483–489.

Schoemaker PJ (1991) The quest for optimality: A positive heuristic of science? *Behav. Brain Sci.* 14(2):205–215.

Shahmoradi Z, Lee T (2021) Quantile inverse optimization: Improving stability in inverse linear programming. *Oper Res.*, ePub ahead of print November 24, https://doi.org/10.1287/opre.2021.2143.

Terekhov AV, Pesin YB, Niu X, Latash ML, Zatsiorsky VM (2010) An analytical approach to the problem of inverse optimization with additive objective functions: An application to human prehension. *J. Math. Biol.* 61(3):423–453.

Troutt MD, Pang WK, Hou SH (2006) Behavioral estimation of mathematical programming objective function coefficients. *Management Sci.* 52(3):422–434.

Wang L (2009) Cutting plane algorithms for the inverse mixed integer linear programming problem. *Oper. Res. Lett.* 37(2):114–116.

Westermann K, Lin JF-S, Kulić D (2020) Inverse optimal control with time-varying objectives: Application to human jumping movement analysis. *Sci. Rep.* 10(1):1–15.

Yang C, Zhang J, Ma Z (1997) Inverse maximum flow and minimum cut problems. *Optimization* 40(2):147–170.

Zhang J, Cai MC (1998) Inverse problem of minimum cuts. *Math. Methods Oper. Res.* 47(1):51–58.

Zhang J, Liu Z (1999) A further study on inverse linear programming problems. *J. Comput. Appl. Math.* 106(2):345–359.

Zhang J, Xu C (2010) Inverse optimization for linearly constrained convex separable programming problems. *Eur. J. Oper. Res.* 200(3):671–679.

Zhang Q, Grossmann IE, Lima RM (2016) On the relation between flexibility analysis and robust optimization for linear systems. *AIChE J.* 62(9):3109–3123.