# Learning in Inverse Optimization: Incenter Cost, Augmented Suboptimality Loss, and Algorithms

Pedro Zattoni Scroccaro, Bilge Atasoy and Peyman Mohajerin Esfahani

ABSTRACT. In Inverse Optimization (IO), an expert agent solves an optimization problem parametric in an exogenous signal. From a learning perspective, the goal is to learn the expert's cost function given a dataset of signals and corresponding optimal actions. Motivated by the geometry of the IO set of consistent cost vectors, we introduce the "incenter" concept, a new notion akin to circumcenter recently proposed by [12]. Discussing the geometric and robustness interpretation of the incenter cost vector, we develop corresponding tractable convex reformulations, which are in contrast with the circumcenter, which we show is equivalent to an intractable optimization program. We further propose a novel loss function called *Augmented Suboptimality Loss* (ASL), a relaxation of the incenter concept for problems with inconsistent data. Exploiting the structure of the ASL, we propose a novel first-order algorithm, which we name *Stochastic Approximate Mirror Descent*. This algorithm combines stochastic and approximate subgradient evaluations, together with mirror descent update steps, which is provably efficient for the IO problems with discrete feasible sets with high cardinality. We implement the IO approaches developed in this paper as a Python package called InvOpt. Our numerical experiments are reproducible, and the underlying source code is available as examples in the InvOpt package.

## 1. INTRODUCTION

In Inverse Optimization (IO) problems, our goal is to model the behavior of an expert agent, which given an exogenous signal, returns a response action. The underlying assumption of IO is that to compute its response, the expert agent solves an optimization problem parametric in the exogenous signal. We assume to know the constraints imposed on the expert, but not its cost function. Therefore, our goal is to model the cost function being optimized by the expert, using examples of exogenous signals and corresponding expert response actions. For example, consider the problem of learning how to route vehicles using examples from experienced drivers. Given a set of customer demands (exogenous signal), the experienced driver chooses a certain vehicle route to serve the customers (expert response). Assuming the drivers solve some kind of route optimization problem to compute their routes, where their cost function depends on the drivers' evaluation of how costly it is to drive from one customer to another, one could look at this problem as an IO problem. Thus, one could use IO tools to learn the cost function being used by these drivers, i.e., we learn the preferences of the expert drivers when driving from a certain customer to another. IO tools have also been used in many other application areas, for instance, healthcare problems [17, 18], modeling of consumer behavior [11], transportation problems [16, 22, 64], portfolio selection [10],

network design [27], forecast of electricity prices [49] and control systems [3]. For more examples of applications of IO, we refer the reader to the recent review paper [20] and references therein.

The literature on IO can be roughly divided into so-called *classical* and *data-driven* IO. In classical IO, the goal is usually to find a cost function that renders a *single* signal-response pair optimal, that is, a cost function under which the observed response is optimal. A direct approach to modeling this scenario leads to a bi-level optimization problem, thus, much of the early IO literature focuses on reformulating this problem into a single-level tractable program [2]. This idea has been extended to different classes of optimization problems, such as conic and integer programs [30, 32, 1, 61, 51]. On the other hand, data-driven IO usually deals with problems with multiple signal-response examples, and it is not necessarily assumed that there exists a cost function consistent with all signal-response data. In this scenario, the IO problem can be viewed as a supervised learning problem with multivariate output where the IO model forms a hypothesis class. In this view, one minimizes a training loss function to find a good "fit" to the input (response), and output (optimizer) data. With this in mind, much of the data-driven IO literature focuses on the choice of the training loss function, particularly because the usual supervised learning losses lead to nonconvex programs [6]. Examples of such losses are the *KKT loss* [35], *first-order loss* [11], *predictability loss* [6], and the *suboptimality loss* [39]. It is worth noting that in the standard IO setting, directly learning the cost function (i.e., regression with signal-pair as the input and the cost function value as the scaler output) is typically not an option since the available data contains only the signal-response pair and not the value of the unknown cost function. Recently, in a framework called "predict, then optimize", [26] consider this additional information, and proposes a loss function coined as SPO. Other concepts have also been investigated in the context of IO problems, such as goodness-of-fit [19], robustness against misspecification [28], and learning constraints instead of cost functions [29].

Recently, the authors in [12] show that given a dataset of expert examples and the set $\mathbb{C}$, which is defined as the set of cost vectors consistent with the dataset, the optimal solution to the IO problem, for a certain regret performance measure, is the so-called *circumcenter* of the set $\mathbb{C}$. To the best of our knowledge, this is the first minimax regret result for IO problems. To derive this result, the authors exploit the geometry of IO problems and provide strong regret guarantees. The general optimization program associated with computing the circumcenter vector, however, turns out to contain intractable problem instances. Moreover, when the IO dataset is inconsistent with a single cost function, it is not clear how one could generalize the circumcenter concept to solve the IO problem. These shortcomings motivate us to propose a new concept called "incenter" to select an IO cost vector from the set of consistent costs. In light of this, the main contributions of this work are summarized as follows:

(i) **(Geometric & robustness interpretations)** Motivated by the geometry of the set of consistent cost vectors, we introduce the concept of an incenter (Definition 3.3), and provide insights into its geometry (Figure 1) and robustness (Remark 3.4) in comparison with the *circumcenter* concept from [12].

(ii) **(Convex reformulations & tractability)** We develop tractable convex reformulations of the incenter (Theorem 3.5, Corollary 3.6), along with a geometric interpretation of these characterizations (Figure 2). This may be of particular interest as the corresponding circumcenter concept is equivalent to an intractable optimization program (Theorem 3.2). To establish this intractability result, we draw connections between the circumcenter/incenter concepts and the well-known problem of extremal volume balls (Remark 3.7). Moreover, since the problem of

extremal volume balls is a special case of the problem of extremal volume ellipsoids, this connection allows us to derive "ellipsoidal" versions of the incenter and circumcenter concepts (e.g., Eq. (12)), which performs well in our numerical experiments (Section 6.1).

(iii) **(Generalization to inconsistent data & Augmented Suboptimality loss)** Generalizing to the inconsistent data setting, we propose a novel loss function for IO problems, which we name *Augmented Suboptimality Loss* (ASL) (Definition 4.1). This loss can be interpreted as a *relaxation* of the incenter concept, to handle IO problems with inconsistent data (Eq. (15)). In special cases, this formulation of the IO problem can be shown to be equivalent to the so-called Structured SVM formulation of structured prediction problems, revealing a connection between IO and structure prediction (Remark 4.3). We further propose a general convex reformulation of the ASL for IO problems with mixed-integer feasible sets (Theorem 4.5 and Corollary 4.6), which generalizes several reformulations from the literature (Remark 4.7).

(iv) **(Tailored first-order algorithm: Stochastic Approximate Mirror Decent)** Motivated by the structure of IO loss functions, we propose a novel first-order algorithm, which we name *Stochastic Approximate Mirror Descent* (SAMD). This algorithm exploits the finite sum structure of the IO problem to compute stochastic gradients, uses approximate evaluating IO loss functions to compute approximate subgradients, and uses mirror descent update steps for problems with favorable geometry (Sections 5.1, 5.2, and 5.3). We prove convergence rates for this algorithm (Proposition 5.2) and show how the components of this algorithm can be tailored to exploit the structure of IO loss functions (Algorithm 1).

(v) **(Inverse Optimization Python package)** We implement the IO approaches developed in this paper as a Python package called InvOpt [63]. This is a Python package developed to solve general IO problems. Our numerical experiments are reproducible, and the underlying source code is available as examples in the InvOpt package.

The rest of the paper is organized as follows. In Section 2 we formalize the IO problem. In Section 3 we present approaches based on the incenter concept. In Section 4, we define the Augmented Suboptimality Loss and show how it can be used to tackle IO problems with inconsistent data and mixed-integer feasible sets. In Section 5, we present first-order algorithms tailored to IO problems. In Section 6 we report our numerical results. In Section 7, we conclude the paper and discuss future research directions. In Appendix A we discuss the theoretical properties of the ASL, in Appendix B we present concise reformulations for some special IO problems with continuous feasible sets, in Appendix C we present proofs, and in Appendix D we present further numerical results.

**Notation.** The trace, range space, and Moore-Penrose inverse of a matrix $Q \in \mathbb{R}^{m \times m}$ are denoted as $\text{Tr}(Q)$, $\mathcal{R}(Q)$ and $Q^\dagger$, respectively. For two symmetric matrices $Q, R \in \mathbb{R}^{m \times m}$, $Q \succeq R$ means $Q - R$ is positive semidefinite. The Euclidean inner product between two vectors $x, y \in \mathbb{R}^m$ is denoted by $\langle x, y \rangle$. The set of vectors with nonnegative components is denoted as $\mathbb{R}^n_+ := \{x \in \mathbb{R}^n : x \geq 0\}$. The cardinality, complement, interior, and convex hull of a set $A$ are denoted as $|A|$, $\overline{A}$, $\text{int}(A)$ and $\text{conv}(A)$, respectively. The set of integers from 1 to $N$ is denoted as $[N]$. A set of indexed values is compactly denoted by $\{x_i\}_{i=1}^N := \{x_1, \ldots, x_N\}$. The euclidean projection of $x$ onto a set $A$ is defined as $\Pi_A(x) := \text{argmin}_{y \in A} \|y - x\|_2$. The euclidean angle between two nonzero vectors $\theta$ and $\tilde{\theta}$ is defined as $a(\theta, \tilde{\theta}) := \arccos \langle \theta, \tilde{\theta} \rangle / (\|\theta\|_2 \|\tilde{\theta}\|_2)$. For vectors $x, y \in \mathbb{R}^m$, $x \odot y$ and $\exp(x)$ mean element-wise multiplication and element-wise exponentiation, respectively. Moreover, whenever the arguments of $\exp(\cdot)$ or $\log(\cdot)$ are matrices, they should be interpreted as the usual matrix exponentiation and matrix logarithm, respectively. The vector $\text{vec}(Q) \in \mathbb{R}^{pq}$ denotes the vectorization of the matrix $Q \in \mathbb{R}^{p \times q}$ formed by stacking the columns of $Q$ into a single column vector. The symbol $\otimes$ denotes

the Kronecker product. The vector $e_j \in \mathbb{R}^n$ denotes the vector of zeros except for the $j$'th element which is equal to one. The vector of ones is represented by $\mathbb{1}$.

## 2. Problem Description

Let us begin by formalizing the IO problem. Let $s \in \mathbb{S}$ be an exogenous signal, and $\mathbb{S}$ be the signal space. The expert agent is assumed to solve the parametric *Forward Optimization Problem*

$$\text{(1)} \qquad \min_{x \in \mathbb{X}(s)} \ F(s, x),$$

where $\mathbb{X}(s)$ is the feasible set, and $F : \mathbb{S} \times \mathbb{X} \to \mathbb{R}$ is the expert's cost function, where we define $\mathbb{X} := \bigcup_{s \in \mathbb{S}} \mathbb{X}(s)$. Recall that the cost $F$ is unknown to us, and we only have access to a dataset of $N$ pairs of exogenous signals and respective expert optimal decisions $\widehat{\mathcal{D}} := \{(\hat{s}_i, \hat{x}_i)\}_{i=1}^N$, that is,

$$\hat{x}_i \in \operatorname*{argmin}_{x \in \mathbb{X}(\hat{s}_i)} \ F(\hat{s}_i, x), \quad \forall i \in [N].$$

We use the notation "$\hat{\ }$" to indicate signal-response data. Using this data, our goal is to learn a cost function that when minimized, reproduces the expert's decisions as well as possible.

Naturally, we can only search for functions in a restricted function space. In this work, we consider the following parametric *hypothesis space*:

$$\text{(2)} \qquad \mathcal{F}_\phi := \big\{ \langle \theta, \phi(\,\cdot\,, \cdot\,) \rangle : \theta \in \mathbb{R}^p \big\},$$

where $\theta \in \mathbb{R}^p$ is called the *cost vector* and $\phi : \mathbb{S} \times \mathbb{X} \to \mathbb{R}^p$ is called *feature mapping*, which maps a signal-response pair $(s, x)$ to a feature vector $\phi(s, x) \in \mathbb{R}^p$. In practice, choosing a suitable mapping $\phi$ is part of the modeling of the IO problem. To exemplify the generality of this hypothesis space, the next example shows how to model a quadratic function as a member of $\mathcal{F}_\phi$. Other utility functions from the literature, for instance, the CES function and the Cobb-Douglas function can also be easily fitted into our framework [21, Appendix A].

**Example 2.1** (Quadratic hypothesis). *Consider a quadratic function*

$$F(s, x) = \langle x, Q_{xx} x \rangle + \langle x, Q_{xs} s \rangle + \langle x, q \rangle,$$

*parametrized by* $(Q_{xx}, Q_{xs}, q) \in \mathbb{R}^{n \times n} \times \mathbb{R}^{n \times m} \times \mathbb{R}^n$. *Using the identity* $\operatorname{vec}(AXB) = \langle \operatorname{vec}(X), B \otimes A^\top \rangle$, *we can rewrite this quadratic function as*

$$F(s, x) = \langle \operatorname{vec}(Q_{xx}), x \otimes x \rangle + \langle \operatorname{vec}(Q_{xs}), s \otimes x \rangle + \langle x, q \rangle = \langle \theta, \phi(s, x) \rangle,$$

*where*

$$\theta := \begin{bmatrix} \operatorname{vec}(Q_{xx}) \\ \operatorname{vec}(Q_{xs}) \\ q \end{bmatrix} \quad and \quad \phi(s, x) := \begin{bmatrix} x \otimes x \\ s \otimes x \\ x \end{bmatrix}.$$

Thus, our IO goal is to find a cost vector $\theta$ such that when solving the optimization problem

$$\text{(3)} \qquad \min_{x \in \mathbb{X}(s)} \ \langle \theta, \phi(s, x) \rangle,$$

we can reproduce (or in some sense approximate) the action the expert would have taken, given the same signal $s$. An important object in IO problems is $\mathbb{C}$, the set of cost vectors consistent with the data $\widehat{\mathcal{D}}$.

**Definition 2.2** (Consistent cost vectors). *Given a dataset* $\widehat{\mathcal{D}} = \{(\hat{s}_i, \hat{x}_i)\}_{i=1}^{N}$, *feature mapping* $\phi$ *and feasible set* $\mathbb{X}$, *the* set of consistent cost vectors *is defined as*

$$(4) \qquad \mathbb{C} := \big\{ \theta \in \mathbb{R}^p : \langle \theta, \phi(\hat{s}_i, \hat{x}_i) - \phi(\hat{s}_i, x_i) \rangle \leq 0, \ \forall x_i \in \mathbb{X}(\hat{s}_i), \ \forall i \in [N] \big\}.$$

In other words, $\mathbb{C}$ corresponds to the set of cost vectors $\theta$ that render the expert responses $\hat{x}_i$'s optimal, i.e., $\hat{x}_i \in \operatorname{argmin}_{x_i \in \mathbb{X}(\hat{s}_i)} \langle \theta, \phi(\hat{s}_i, x_i) \rangle$, for all $i \in [N]$. Geometrically, the set $\mathbb{C}$ is a cone, defined by the intersection of half-spaces defined by hyperplanes that pass through the origin.

## 3. Incenter Cost Vector

From Definition (2.2), one can observe that a trivial element in the set $\mathbb{C}$ is $\theta = 0$. However, this trivial choice is not really of interest since it yields any response $x \in \mathbb{X}(s)$ to the signal s an optimal solution, hence no differentiation between different responses. Therefore, when $\mathbb{C}$ contains other elements than $\theta = 0$, that is, the data is consistent with the hypothesis space in the sense that the expert's true cost is $F(s,x) = \langle \theta^\star, \phi(s,x) \rangle$ for some nonzero element $\theta^\star$, it is natural to restrict the search space to $\mathbb{C} \setminus \{0\}$. It is also worth noting that a similar issue concerns the feature map $\phi$. Namely, from a modeling perspective, it is important to consider a feature class where $\phi(s,x)$ is not identically zero in the second argument over the set $\mathbb{X}(s)$ for typical signal variables $s \in \mathbb{S}$. This assumption, together with excluding the trivial cost vector $\theta = 0$ from the search space, is necessary. Thus, in this section, we assume $\mathbb{C} \setminus \{0\} \neq \emptyset$ and that $\phi$ is not trivially zero, and we discuss different ways to choose a cost vector from the set $\mathbb{C}$. A straightforward way to do this is by solving a feasibility optimization problem, which we call *feasibility program*:

$$(5) \qquad \begin{aligned} \min_{\theta} \quad & 0 \\ \text{s.t.} \quad & \langle \theta, \phi(\hat{s}_i, \hat{x}_i) - \phi(\hat{s}_i, x_i) \rangle \leq 0 \qquad \forall x_i \in \mathbb{X}(\hat{s}_i), \ \forall i \in [N] \\ & \|\theta\| = 1. \end{aligned}$$

Problem (5) simply translates the idea that we wish to find some nonzero $\theta \in \mathbb{C}$. Notice that the normalization constraint $\|\theta\| = 1$ excludes the trivial solution $\theta = 0$, and it is nonrestrictive in the sense that $\operatorname{argmin}_{x \in \mathbb{X}(s)} \langle \theta, \phi(s,x) \rangle = \operatorname{argmin}_{x \in \mathbb{X}(s)} \langle \alpha\theta, \phi(s,x) \rangle$, for any $\alpha > 0$. By solving the IO problem using the feasibility program (5), we implicitly suggest that any cost vector $\theta \in \mathbb{C} \setminus \{0\}$ is an equally good solution to the IO problem. However, one could argue that there are more principled ways to choose a cost vector from the set of consistent vectors $\mathbb{C}$.

### 3.1. Geometry, robustness, and tractability

Considering two cost vectors $\theta$ and $\theta^\star$ with their corresponding responses defined as

$$(6a) \qquad x^\theta \in \operatorname*{argmin}_{x \in \mathbb{X}(\hat{s})} \langle \theta, \phi(\hat{s}, x) \rangle \quad \text{and} \quad x^\star \in \operatorname*{argmin}_{x \in \mathbb{X}(\hat{s})} \langle \theta^\star, \phi(\hat{s}, x) \rangle,$$

the authors in [12] define the regret

$$(6b) \qquad R(\theta, \theta^\star) := \langle \theta^\star, \phi(\hat{s}, x^\theta) - \phi(\hat{s}, x^\star) \rangle.$$

The regret (6b) is a distance function between two cost vectors that measures the difference via the impact of their respective optimizers $x^\theta$ and $x^\star$ evaluated through the second argument, here the ground truth $\theta^\star$. As shown in [12], the worst-case optimal cost vector $\theta$ for the regret $R(\theta, \theta^\star)$ is the so-called *circumcenter* of $\mathbb{C}$, as defined in the following.

**Definition 3.1** (Circumcenter [12]). *Let $\mathbb{C}$ be a nonempty set. A* circumcenter *of $\mathbb{C}$ is defined as*

$$(7) \qquad\qquad \theta^{\mathrm{c}} \in \underset{\theta \neq 0}{\operatorname{argmin}} \ \underset{\substack{\tilde{\theta} \in \mathbb{C} \\ \tilde{\theta} \neq 0}}{\max} \ a(\theta, \tilde{\theta}).$$

In the original definition of [12], the non-zero constraints over the cost vectors are enforced via $\|\theta\|_2 = \|\tilde{\theta}\|_2 = 1$, which does not change the circumcenter since the angle in the objective is scale-invariant (recall that the euclidean angle between two nonzero vectors $\theta$ and $\tilde{\theta}$ is defined as $a(\theta, \tilde{\theta}) = \arccos \langle \theta, \tilde{\theta} \rangle / (\|\theta\|_2 \|\tilde{\theta}\|_2)$). Geometrically, $\theta^{\mathrm{c}}$ can be interpreted as a point in the axis of the revolution cone with the smallest aperture angle containing $\mathbb{C}$. Informally speaking, the circumcenter approach chooses a cost vector by looking for the vector in $\mathbb{C}$ furthest away (in terms of the angle) from the "corners" of $\mathbb{C}$ (more precisely, from the extreme rays of $\mathbb{C}$). However, in this study, we identify the following three possible shortcomings when using the circumcenter concept to choose an IO model:

(i) **Computational tractability:** Computing the circumcenter, in general, turns out to be computationally intractable.

(ii) **Robustness interpretation:** While the circumcenter is, by definition, robust to an adversarial (i.e., worst-case) ground truth data-generating model, the challenge in practice is often related to noisy data.

(iii) **Inconsistent data:** The circumcenter is only defined for problems with consistent data (i.e., nonempty $\mathbb{C}$) and, to the best of our knowledge, cannot be straightforwardly extended to IO problems with inconsistent data.

In the rest of this section, we first formalize the tractability result (i), and then introduce a new notion that addresses these possible shortcomings related to circumcenter.

**Theorem 3.2** (Intractability of circumcenter). *Assume $\mathbb{C}$ is a nonempty polyhedral cone. Then, solving the inner maximization of* (7) *for any $\theta$ is equivalent to maximizing a quadratic function over a polytope, which is NP-hard.*

**Definition 3.3** (Incenter). *Let $\mathbb{C}$ be a nonempty set. An* incenter *of $\mathbb{C}$ is defined as*

$$(8) \qquad\qquad \theta^{\mathrm{in}} \in \underset{\theta \neq 0}{\operatorname{argmax}} \ \underset{\substack{\tilde{\theta} \in \overline{\operatorname{int}(\mathbb{C})} \\ \tilde{\theta} \neq 0}}{\min} \ a(\theta, \tilde{\theta}).$$

We will show how the incenter problem (8) can be reformulated as a tractable convex optimization problem (as opposed to the NP-hardness of the circumcenter), and by relaxing the reformulation, it can be straightforwardly extended to handle problems with inconsistent data. Let us first elaborate more on the intuition behind the incenter vector in Definition 3.3 compared to the circumcenter vector in Definition 3.1. Geometrically, an incenter of $\mathbb{C}$ can be interpreted as a vector furthest away (in terms of the angle) from the *exterior* of $\mathbb{C}$, or in other words, the vector in $\mathbb{C}$ furthers away from the boundary of $\mathbb{C}$. To provide insight into these different centers, we visualize them in a simple three-dimensional space in Figure 1. Interestingly, these notions also have robustness interpretations, which are different from existing learning models in which the robustness is introduced explicitly via regularization or distributional robust approaches [39].

**Remark 3.4** (Robustness: circumcenter vs. incenter). *A circumcenter $\theta^{\mathrm{c}}$ in* (7) *can be interpreted as the vector most robust to an adversary with the authority to choose the true data-generating cost vector in $\mathbb{C}$ to be furthest away from our learned model, which is a notion captured by the regret*

(a) Circumcenter $\theta^c$ defined in (7).  (b) Incenter $\theta^{\text{in}}$ defined in (8).
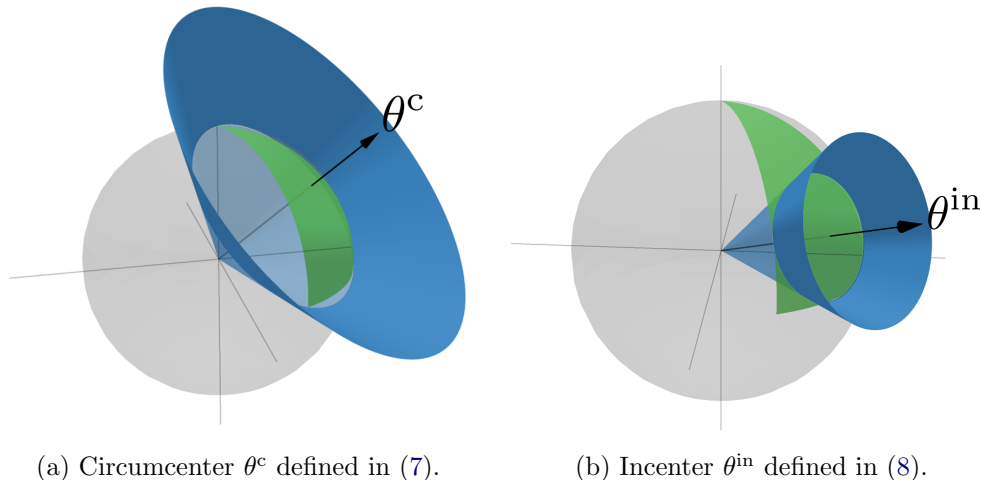
Figure 1. Geometrical visualization of the circumcenter and incenter vectors. The green regions are the intersection of $\mathbb{C}$ with the sphere and the blue cones are revolution cones with an aperture angle equal to the optimal value of (7) and (8).

*performance measure introduced in [12]. On the other hand, the incenter $\theta^{\text{in}}$ (8) can be viewed as the vector furthest away from the boundary of $\mathbb{C}$. Since each facet of $\mathbb{C}$ is determined by a signal-response $(\hat{s}_i, \hat{x}_i)$ pair in the dataset of the IO problem, the incenter vector can be interpreted as the cost vector most robust to perturbations of the training dataset (i.e., perturbations of the facets of $\mathbb{C}$).*

Let us provide some additional details supporting the robustness interpretation in Remark 3.4. To formalize this robustness notion, recall the definition of the set of consistent cost vectors $\mathbb{C} = \{\theta \in \mathbb{R}^p : \langle \theta, \Delta(\hat{s}_i, \hat{x}_i, x)\rangle \leq 0, \ \forall x \in \mathbb{X}(\hat{s}_i), \ \forall i \in [N]\}$, where we define $\Delta(\hat{s}_i, \hat{x}_i, x) \coloneqq (\phi(\hat{s}_i, \hat{x}_i) - \phi(\hat{s}_i, x_i))/\|\phi(\hat{s}_i, \hat{x}_i) - \phi(\hat{s}_i, x_i)\|_2$ if $\phi(\hat{s}_i, \hat{x}_i) \neq \phi(\hat{s}_i, x_i)$, and $\Delta(\hat{s}_i, \hat{x}_i, x) \coloneqq 0$ otherwise. Differently from the definition in Eq. (4), here we simply normalize the vectors $\phi(\hat{s}_i, \hat{x}_i) - \phi(\hat{s}_i, x_i)$, which does not change the set $\mathbb{C}$. Notice that the vectors $\Delta(\hat{s}_i, \hat{x}_i, x)$ are defined by the data $\{(\hat{s}_i, \hat{x}_i)\}_{i=1}^N$, and in turn define the set $\mathbb{C}$. Thus, the vector $\theta \in \mathbb{C}$ most robust to perturbations in the data, i.e., perturbation of $\Delta(\hat{s}_i, \hat{x}_i, x)$, can be found by solving the minimax problem

(9)
$$\max_{\substack{\|\theta\|_2 = 1 \\ \theta \in \mathbb{C}}} \min_{\substack{w \in \mathbb{R}^p \\ (\hat{s}_k, \hat{x}_k) \in \{(\hat{s}_i, \hat{x}_i)\}_{i=1}^N \\ x \in \mathbb{X}(\hat{s}_k)}} \|w\|_2^2$$
$$\text{s.t.} \qquad \langle \theta, \Delta(\hat{s}_k, \hat{x}_k, x) + w\rangle = 0.$$

In problem (9), the "max player" optimizes for the vector $\theta \in \mathbb{C}$ that requires the largest perturbation $w$ so that it lies in the hyperplane $\langle \theta, \Delta(\hat{s}_k, \hat{x}_k, x) + w\rangle = 0$ (i.e., a perturbed facet of $\mathbb{C}$). The "min player" tries to find the "most vunerable" $\Delta(\hat{s}_k, \hat{x}_k, x)$, that is, the facet of $\mathbb{C}$ that requires the smallest perturbation vector $w$ to make $\langle \theta, \Delta(\hat{s}_k, \hat{x}_k, x) + w\rangle = 0$. It can be shown that the optimal perturbation is $w = -(\langle \theta, \Delta(\hat{s}_k, \hat{x}_k, x)\rangle / \|\theta\|_2^2)\theta$, and substituting it in the objective function of (9), one can show that the optimization problem (9) is equivalent to the incenter reformulation of Theorem 3.5.

As a final comment, even ignoring the computational cost of computing the circumcenter vector versus computing the incenter vector, for cases when the set of consistent vectors $\mathbb{C}$ is not empty, the incenter approach may perform better in practice (e.g., see the results in Section 6.1). In light
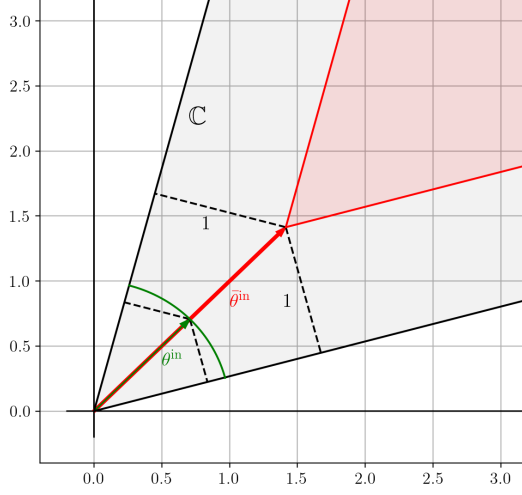
Figure 2. Geometrical illustration of Corollary 3.6 in a simple 2D example. The grey region represents $\mathbb{C}$, the green region represents the feasible set of (10), and the red region represents the feasible set of (11). As can be seen, the optimal solution of (11) can be interpreted as the smallest norm vector inside an inner cone with boundaries 1 unit away from the boundaries of $\mathbb{C}$. Also, it can be seen that by normalizing $\bar{\theta}^{\text{in}}$, we retrieve $\theta^{\text{in}}$, an optimal solution of (10).

of the discussion in Remark 3.4, this suggests that approaches robust to perturbations in the data (i.e., incenter) might be preferred to the one that is robust to an adversarial true data-generating model (i.e., circumcenter).

### 3.2. Reformulations

Next, we present a reformulation of (8). This reformulation will be used to derive tractable incenter-based approaches to IO problems, and we show that if $\text{int}(\mathbb{C}) \neq \emptyset$, one can compute the incenter vector by solving an optimization problem without a norm equality constraint.

**Theorem 3.5** (Incenter reformulation). *Problem* (8) *can be reformulated as*

(10)
$$(\theta^{\text{in}}, r^{\text{in}}) \in \arg \max_{\theta, r} \ r$$
$$\text{s.t. } \langle \theta, \phi(\hat{s}_i, \hat{x}_i) - \phi(\hat{s}_i, x_i) \rangle + r\|\phi(\hat{s}_i, x_i) - \phi(\hat{s}_i, \hat{x}_i)\|_2 \leq 0 \ \forall x_i \in \mathbb{X}(\hat{s}_i), \forall i \in [N]$$
$$\|\theta\|_2 = 1.$$

**Corollary 3.6** (Incenter convex characterization). *Assume* $\text{int}(\mathbb{C}) \neq \emptyset$. *Then, problem* (10) *can be solved as*

(11)
$$\bar{\theta}^{\text{in}} \in \arg \min_{\theta} \ \|\theta\|_2$$
$$\text{s.t. } \langle \theta, \phi(\hat{s}_i, \hat{x}_i) - \phi(\hat{s}_i, x_i) \rangle + \|\phi(\hat{s}_i, x_i) - \phi(\hat{s}_i, \hat{x}_i)\|_2 \leq 0 \quad \forall x_i \in \mathbb{X}(\hat{s}_i), \ \forall i \in [N],$$

*where* $\theta^{\text{in}} = \bar{\theta}^{\text{in}}/\|\bar{\theta}^{\text{in}}\|_2$ *is an optimal solution of problem* (10).

Figure 2 presents a geometrical intuition behind the alternative reformulation of Corollary 3.6 in a simple 2D example. Exploiting the nonempty interior assumption $\text{int}(\mathbb{C}) \neq \emptyset$, we were able to formulate problem (11) without a norm equality constraint, making it a convex optimization problem. In particular, it is tractable whenever $\mathbb{X}(\hat{s}_i)$ has finitely many elements for all $i \in [N]$. In sections 4.2 and 5, we discuss approaches to deal with cases when $\mathbb{X}(\hat{s}_i)$ may have infinitely many elements. The tractability results in Theorem 3.2 and Corollary 3.6 build on a connection to well-known problems in the optimization literature: computing extremal volume balls.

**Remark 3.7** (Connection with extremal volume balls). *Interestingly, the circumcenter (resp. incenter) problem is closely related to the problem of finding the minimum (resp. maximum) volume ball that covers (resp. lies inside) a nonempty polyhedron. The difference is that, instead of optimizing the volume of a ball so that it covers (resp. lies inside) a nonempty polyhedron, we optimize the aperture angle of a circular cone, so that it covers (resp. lies inside) a polyhedral cone (see Figure 1). It is known that computing the minimum volume ball that contains a polyhedron defined by linear inequalities is an intractable convex optimization problem [41, Section 10.5.1], and Theorem 3.2 indicates that the circumcenter has the same property. On the other hand, finding the maximum volume ball that lies inside a polyhedron defined by linear inequalities (a.k.a. the Chebyshev center of the polyhedron) is known to be equivalent to a convex optimization problem [14, Section 8.5.1], and Corollary 3.6 also confirms that the incenter has the same property.*

Since the problem of extremal volume balls is a special case of the more general problem of extremal volume ellipsoids [14, Section 8.4], one natural generalization of the incenter and circumcenter concepts is to use *ellipsoidal cones*. For instance, an ellipsoidal generalization of the circumcenter concept is used by [12] for online IO problems. An ellipsoidal generalization of the incenter reformulation of Theorem 3.5 is

$$
\begin{aligned}
& \max_{\theta, A} \ \log\left(\det\left(A\right)\right) \\
(12) \quad & \text{s.t.} \ \langle \theta, \phi(\hat{s}_i, \hat{x}_i) - \phi(\hat{s}_i, x_i) \rangle + \|A(\phi(\hat{s}_i, x_i) - \phi(\hat{s}_i, \hat{x}_i))\|_2 \leq 0 \quad \forall x_i \in \mathbb{X}(\hat{s}_i), \forall i \in [N] \\
& \|\theta\|_2 = 1, \quad A \succeq 0,
\end{aligned}
$$

which is based on [14, Section 8.4.2]. Even though, to the best of our knowledge, there are no theoretical results for the performance of ellipsoidal incenter and circumcenter approaches for offline IO problems, their performance in our numerical experiments in Section 6 is indeed promising.

We end this section presenting a generalization of the program (11). This generalization will give us the flexibility to exploit structures specific to different IO problems. We generalize (11) as

$$
\begin{aligned}
& \min_{\theta} \ \mathcal{R}(\theta) \\
(13) \quad & \text{s.t.} \ \langle \theta, \phi(\hat{s}_i, \hat{x}_i) - \phi(\hat{s}_i, x_i) \rangle + d(\hat{x}_i, x_i) \leq 0 \quad \forall x_i \in \mathbb{X}(\hat{s}_i), \ \forall i \in [N] \\
& \theta \in \Theta.
\end{aligned}
$$

This generalization occurs on three levels:

(i) **Regularizer**: We generalize the objective function to a general regularization function $\mathcal{R} : \mathbb{R}^p \to \mathbb{R}$. For example, we could have $\mathcal{R}(\theta) = \|\theta - \hat{\theta}\|_2^2$, where $\hat{\theta}$ is an a priory belief or estimate of the true cost vector, or $\mathcal{R}(\theta) = \|\theta\|$ for a general norm.

(ii) **Distance in the response space**: Instead of computing the distance between vectors using the Euclidean distance, we consider a general distance function $d : \mathbb{X} \times \mathbb{X} \to \mathbb{R}_+$. For instance, $d(\hat{x}, x) = \|\hat{x} - x\|$ on continuous spaces, or $d(\hat{x}, x) = I(\hat{x}, x)$ on discrete spaces, where $I(\hat{x}, x) = 0$

if $\hat{x} = x$, otherwise $I(\hat{x}, x) = 1$. We note that $d$ could also be a function of $\hat{s}$. For instance, we could use it to penalize the distance between $\hat{x}$ and $x$ in the *feature space*, by choosing $d(\hat{x}, x) = \|\phi(\hat{s}, \hat{x}) - \phi(\hat{s}, x)\|$. For simplicity, we omit this dependency.

(iii) **Prior information constraint**: We add the additional constraint $\theta \in \Theta$. The set $\Theta$ is used to encode any prior information or assumption we may have on the expert's true cost function, e.g., nonnegativity of the cost vector.

Notice that as long as $\mathcal{R}(\theta)$ is convex in $\theta$ and the set $\Theta$ is convex, Problem (13) is a convex optimization problem.

## 4. Augmented Suboptimality Loss

In the previous section, we have shown how one can tackle IO problems when the dataset $\widehat{\mathcal{D}}$ is consistent with some cost $F(s, x) = \langle \theta^\star, \phi(s, x) \rangle \in \mathcal{F}_\phi$. Moreover, to use the proposed approaches in practice, we also need to be able to list all elements of $\mathbb{X}(\hat{s}_i)$ for all $i \in [N]$, since each of these elements induces one inequality constraint (e.g., see (13)). In this section, we present approaches to tackle an IO problem based on a *loss function*, which will allow us to handle problems with possibly inconsistent data and when the cardinality of $\mathbb{X}(\hat{s}_i)$ is very large (or even infinite). We first introduce a novel loss function for IO problems and show how solving the IO problem using this loss can be interpreted as a relaxation of the optimization program (13).

Recall that in IO problems, our goal is to find a parameter vector $\theta$, such that when solving the optimization problem $\min_{x \in \mathbb{X}(s)} \langle \theta, \phi(s, x) \rangle$, we can reproduce (or approximate) the action the expert would have taken given the same signal $s \in \mathbb{S}$ (see Section 2). In the data-driven IO literature, this problem is usually posed as a (regularized) empirical loss minimization problem

$$(14) \qquad \min_{\theta \in \Theta} \; \kappa \mathcal{R}(\theta) + \frac{1}{N} \sum_{i=1}^{N} \ell_\theta(\hat{s}_i, \hat{x}_i),$$

where $\mathcal{R} : \mathbb{R}^p \to \mathbb{R}$ is a regularization function, $\kappa$ is a nonnegative regularization parameter, and $\ell_\theta : \mathbb{S} \times \mathbb{X} \to \mathbb{R}$ is some loss function, designed in a way that, by solving (14), we find a cost vector $\theta$ such that the function $\langle \theta, \phi(s, x) \rangle$ achieves our IO goal. Similar to supervised learning methods in machine learning, the regularization parameter $\kappa$ should be chosen to improve the out-of-sample performance of the learned model, e.g., using cross-validation techniques. In this work, we propose a new loss function for IO problems, which we name *Augmented Suboptimality Loss* (ASL).

**Definition 4.1** (Augmented Suboptimality Loss). *Given a signal-response pair $(\hat{s}, \hat{x})$, the Augmented Suboptimality Loss of a cost vector $\theta$ is*

$$(\text{ASL}) \qquad \ell_\theta(\hat{s}, \hat{x}) := \max_{x \in \mathbb{X}(\hat{s})} \left\{ \langle \theta, \phi(\hat{s}, \hat{x}) - \phi(\hat{s}, x) \rangle + d(\hat{x}, x) \right\},$$

*where $\phi$ is a feature mapping and $d$ is a distance function.*

To simplify the notation, we omit the dependence of the ASL on $\phi$ and $d$. We note that the ASL can be the well-known *Suboptimality Loss* [39] for the special case with no distance penalization, i.e., $d(\hat{x}, x) = 0$.

### 4.1. **Connections with incenter**

Next, we show how to solve problem (14) when the loss function is the ASL. This reformulation is useful to tackle IO problems with inconsistent data, where there is no cost vector $\theta^\star$ that describes

the dataset $\widehat{\mathcal{D}} = \{(\hat{s}_i, \hat{x}_i)\}_{i=1}^N$ perfectly. Using a standard epigraph reformulation, one can see that the optimization problem (14) with the ASL yields the program

$$
\begin{aligned}
(15) \qquad & \min_{\theta, \beta_1, \ldots, \beta_N} \quad \kappa \mathcal{R}(\theta) + \frac{1}{N} \sum_{i=1}^N \beta_i \\
& \text{s.t.} \quad \langle \theta, \phi(\hat{s}_i, \hat{x}_i) - \phi(\hat{s}_i, x_i) \rangle + d(\hat{x}_i, x_i) \leq \beta_i \qquad \forall x_i \in \mathbb{X}(\hat{s}_i), \ \forall i \in [N] \\
& \qquad \theta \in \Theta.
\end{aligned}
$$

Problem (15) is closely related to the incenter reformulation (13). More precisely, (15) can be interpreted as a *relaxation* of (13) to handle IO problems with inconsistent data. To see this, simply notice that we can arrive at (15) by adding *slack variables* to the constraints of (13) and using $\kappa$ as a trade-off parameter between the sum of slack variables (i.e., the total violation of the constraints of (13)) and the regularization term $\mathcal{R}(\theta)$. Another way to see (14) as a relaxation of (13), is to write (13) in a "loss minimization" form. Namely, notice that (13) can be written as

$$
(16) \qquad \min_{\theta \in \Theta} \ \mathcal{R}(\theta) + \mathbb{I} \left( \frac{1}{N} \sum_{i=1}^N \ell_\theta(\hat{s}_i, \hat{x}_i) \right),
$$

where the indicator function $\mathbb{I}(a) = 0$ if $a \leq 0$, otherwise $\mathbb{I}(a) = \infty$. Thus, (14) can also be interpreted as Lagrangian relaxation of (16). In Appendix A, we present a different way to motivate the ASL, namely, as a convex surrogate of the so-called *predictability loss*. There, we also discuss several properties of the ASL which are relevant to IO problems.

So far, we have only considered the case when the data $\{\hat{x}_i\}_{i=1}^N$ is feasible, i.e., $\hat{x}_i \in \mathbb{X}(\hat{s}_i)$. However, in practice, it may be the case that $\hat{x}_i \notin \mathbb{X}(\hat{s}_i)$ for some $i \in [N]$. This may happen due to noise in the data acquisition, or simply due to a mismatch in the choice of the set mapping $\mathbb{X}$ compared with the true constraint set used by the expert agent. For example, consider the problem where the signal $\hat{s}_i$ represents a graph, and $\mathbb{X}(\hat{s}_i)$ is the set of *Hamiltonian cycles* over this graph (i.e., a graph cycle that visits each node exactly once). This scenario is common when modeling vehicle routing problems, where the nodes of the graph represent a set of customers, and the Hamiltonian cycle represents the sequence of customers a driver visited (for example, to deliver packages). In this scenario, infeasible data could reflect, for example, missing data on the complete cycle driven by the driver (i.e., noise in the data acquisition), or the fact that in reality, the driver can visit the same customer multiple times to deliver missing packages (i.e., mismatch in the choice of the set mapping $\mathbb{X}(\hat{s}_i)$).

**Remark 4.2** (Handling infeasible data)**.** *One way to handle infeasible data is just trying to ignore these cases, treating them as outliers in the data. Numerically, since the nonnegativity of the ASL depends on the assumption that $\hat{x}_i \in \mathbb{X}(\hat{s}_i) \ \forall i \in [N]$ (see Proposition A.3), having infeasible data may lead to negative loss values. Thus, a possible way to handle these cases is to use a modified loss function $\tilde{\ell}_\theta(\hat{s}, \hat{x}) = \max\{0, \ell_\theta(\hat{s}, \hat{x})\}$, which simply ignores negative loss values (i.e., infeasible data). This technique shares the same principle as the bounded rationality loss proposed in [39]. Importantly, this modification preserves the convexity of the loss minimization problem. For instance, this modification is equivalent to simply adding the constraints $\beta_i \geq 0, \ \forall i \in [N]$ to (15).*

The IO problem shares some similarities with *structured prediction* problems, which commonly refer to the class of supervised learning problems, where the output space consists of a finite set of structured objects (e.g., graphs), instead of a simple set of labels, like in standard multiclass

classification problems [54]. Interestingly, the IO formulation in Eq. (15) is closely related to some formulations proposed to solve structured prediction problems.

**Remark 4.3** (Connections with structured prediction). *In the incenter program* (15)*, when $\mathcal{R}$ is the Euclidean norm, $d$ is the Hamming distance, and feature function $\phi$ is linear in $x$, learning the incenter cost vector is equivalent to the Maximum Margin Planning problem presented in [47], which is a type of structured prediction problem. More generally, formulation* (15) *is closely related to the so-called Structured Support Vector Machine (SSVM) approach for structured prediction problems* [59, 44].

The connection between IO problems with discrete feasible sets and the SSVM approach to structured prediction problems is related to the fact that when the constraint set $\mathbb{X}(\hat{s})$ has finitely many elements, each of these elements can be interpreted as a class in a multiclass classification problem. Under this interpretation, the optimization problem (15) can be viewed as a generalization of the soft-margin SVM problem, where the "classes" $x \in \mathbb{X}(\hat{s})$ can be complex structured objects, which is precisely the SSVM problem. Moreover, the concept of "margin" in the SVM framework is related to the distance function $d$ in (15), which comes from the fact that in our incenter-based IO formulation, we want to maximize the angle between the incenter vector and the boundaries of the set $\mathbb{C}$ (see Figure 1b). These observations reveal an interesting connection between incenter-based approaches for IO problems with discrete constraint sets and SSVM approaches for structured prediction problems.

In the last part of this section, we revisit this relation between the incenter and circumcenter concepts through the lens of the regret $R$ defined in (6). In fact, given two cost vectors $(\theta, \theta^\star)$, there is an asymmetry $R(\theta, \theta^\star) \neq R(\theta^\star, \theta)$, interestingly, each of which relating to one of these concepts.

**Remark 4.4** (Connections with regret). *Consider the cost vectors $(\theta, \theta^\star)$ and the regret* (6b).

  (i) **SPO vs Suboptimality losses:** *The regret $R(\theta, \theta^\star)$ is indeed the Smart "Predict, then Optimize" loss (SPO) studied in [26], with respect to which the circumcenter in Definition 3.1 is shown to be the worst case optimal. However, the symmetric counterpart $R(\theta^\star, \theta)$ coincides with the Suboptimality loss from [39], which, as discussed above, is a special case of the ASL connected to the incenter in Definition 3.3.*

 (ii) **Convexity and tractability:** *The regret $R$ is convex in the second argument [39] (note the connection to the Suboptimality loss in the previous point), whereas inherently nonconvex in the first argument; see [26] for the connection of the latter to the 0-1 loss in classification problems. This observation is indeed aligned with the intractability results of circumcenter (Theorem 3.2) and the tractability of incenter (Corollary 3.6). It is worth noting that the authors in [26] also propose a convexified version of the SPO loss (i.e., the regret in the first argument), reminiscent of the hinge loss in classification problems.*

(iii) **Additional required data measurements**: *Given the cost vector $\theta$ and the IO data pair $(\hat{s}, x^\star)$ (for the definition of $x^\star$ see* (6a)*), the regret $R(\theta^\star, \theta)$ can be computed without the knowledge of the ground truth $\theta^\star$, whereas its symmetric counterpart $R(\theta, \theta^\star)$ does depend explicitly on $\theta^\star$ (or its projection through a feature function as in [26]).*

## 4.2. General reformulation for mixed-integer feasible sets

In this section, we present a way to reformulate the IO problem using the ASL for problems when $\mathbb{X}(\hat{s})$ is a mixed-integer set, which generalizes many formulations from the literature. For this

purpose, we consider the mixed-integer feasible set

$$(17) \qquad \mathbb{X}(\hat{s}) := \left\{ (y, z) \in \mathbb{R}^u \times \mathbb{Z}^v : \hat{A}y + \hat{B}z \le \hat{c}, \ z \in \mathbb{Z}(\hat{w}) \right\},$$

where $\hat{s} := (\hat{A}, \hat{B}, \hat{c}, \hat{w})$, and $\mathbb{Z}(\hat{w})$ is a bounded set that may depend on the signal $\hat{w}$. Since $y$ is a continuous variable, $\mathbb{X}(\hat{s})$ has infinitely many elements and we cannot use (15) for this IO problem in practice. To solve this problem, one could use first-order iterative methods to directly optimize the IO loss minimization problem (14) (we discuss such approaches in Section 5). In this section, we leverage classical tools from convex duality to reformulate (14) with the ASL and constraint set (17) as a tractable finite convex optimization.

For this reformulation, we use the following hypothesis function

$$(18) \qquad \langle \theta, \phi(\hat{s}, x) \rangle = \langle y, Q_{yy}y \rangle + \langle y, Q\phi_1(\hat{w}, z) \rangle + \langle q, \phi_2(\hat{w}, z) \rangle,$$

where $x := (y, z)$, $\theta := (\text{vec}(Q_{yy}), \text{vec}(Q), q) \in \mathbb{R}^{u^2 + um + r}$, $Q_{yy} \succcurlyeq 0$, and $\phi_1 : \mathbb{W} \times \mathbb{Z}^v \to \mathbb{R}^m$ and $\phi_2 : \mathbb{W} \times \mathbb{Z}^v \to \mathbb{R}^r$ are feature functions ($\phi$ can be written in terms of $\phi_1$ and $\phi_2$). The choice of a quadratic hypothesis and $\mathbb{X}(\hat{s})$ with linear inequality constraints is chosen because it generalizes the linear hypothesis case, and also for simplicity of exposition. In general, as long as the hypothesis function and constraint set are convex w.r.t. the continuous part of the decision vector (e.g., *conic representable* problems [39]), similar results could be derived. Continuing, we use $d(\hat{x}, x) = \|\hat{y} - y\|_\infty + d_z(\hat{z}, z)$, that is, we use the $\infty$-norm for the continuous part of the decision vector and a general distance function for the integer part of the decision vector. We use the $\infty$-norm for the continuous part of the decision vector because, to use reformulation techniques based on convex duality, we need the inner maximization problem of the ASL to be concave in $y$, and by introducing auxiliary integer variables, we can reformulate the $\infty$-norm as a concave function of $y$. More precisely, we exploit the identity $\|\hat{y} - y\|_\infty = \max_{h \in \{-1,0,1\}^u, \|h\|_1 = 1} \langle h, \hat{y} - y \rangle$.

**Theorem 4.5** (ASL and mixed-integer feasible set). *For the mixed-integer feasible set* (17), *hypothesis function* (18), *and distance function* $d(\hat{x}, x) = \|\hat{y} - y\|_\infty + d_z(\hat{z}, z)$, *problem* (14) *can be reformulated as*

$$(19) \qquad \begin{aligned} & \min \ \kappa \mathcal{R}(\theta) + \frac{1}{N} \sum_{i=1}^N \beta_i \\ & \text{s.t. } \theta = (\text{vec}(Q_{yy}), \text{vec}(Q), q) \in \Theta, \quad \lambda_{ijk} \ge 0, \quad \alpha_{ijk}, \beta_i \in \mathbb{R} \\ & \quad \langle \theta, \phi(\hat{s}_i, \hat{x}_i) \rangle + \alpha_{ijk} + \langle \lambda_{ijk}, \hat{c}_i - \hat{B}_i z_{ij} \rangle - \langle q, \phi_2(\hat{w}_i, z_{ij}) \rangle + \langle h_k, \hat{y}_i \rangle + d_z(\hat{z}_i, z_{ij}) \le \beta_i \\ & \quad \begin{bmatrix} Q_{yy} & Q\phi_1(\hat{w}_i, z_{ij}) + h_k + \hat{A}_i^\top \lambda_{ijk} \\ * & 4\alpha_{ijk} \end{bmatrix} \succcurlyeq 0, \end{aligned}$$

*where the constraints are for all* $(i, j, k) \in [N] \times [M_i] \times [2u]$, $\mathbb{Z}(\hat{w}_i) := \{z_{i1}, \ldots, z_{ij}, \ldots, z_{iM_i}\}$, $M_i := |\mathbb{Z}(\hat{w}_i)|$, *and if* $k \le u$ *(resp.* $k > u$*),* $h_k$ *is the vector of zeros except for the* $k$'*th (resp.* $(k - u)$'*th) element, which is equal to 1 (resp. -1).*

In case we use a linear hypothesis function instead of a quadratic one, the matrix inequality constraints of (19) reduce to much simpler linear equality constraints.

**Corollary 4.6** (LP reformulation for linear hypotheses). *For the mixed-integer feasible set* (17), *linear hypothesis function (i.e.,* (18) *with* $Q_{yy} = 0$*), and distance function* $d(\hat{x}, x) = \|\hat{y} - y\|_\infty +$

$d_z(\hat{z}, z)$, *problem* (14) *can be reformulated as*

$$\min \ \kappa \mathcal{R}(\theta) + \frac{1}{N} \sum_{i=1}^{N} \beta_i$$

(20)     s.t. $\theta = (\text{vec}(Q), q) \in \Theta, \quad \lambda_{ijk} \geq 0, \quad \beta_i \in \mathbb{R}$

$$\langle \theta, \phi(\hat{s}_i, \hat{x}_i) \rangle + \langle \lambda_{ijk}, \hat{c}_i - \hat{B}_i z_{ij} \rangle - \langle q, \phi_2(\hat{w}_i, z_{ij}) \rangle + \langle h_k, \hat{y}_i \rangle + d_z(\hat{z}_i, z_{ij}) \leq \beta_i$$

$$Q \phi_1(\hat{w}_i, z_{ij}) + h_k + \hat{A}_i^\top \lambda_{ijk} = 0,$$

*where the constraints are for all* $(i, j, k) \in [N] \times [M_i] \times [2u]$, $\mathbb{Z}(\hat{w}_i) := \{z_{i1}, \ldots, z_{ij}, \ldots, z_{iM_i}\}$, $M_i := |\mathbb{Z}(\hat{w}_i)|$, *and if* $k \leq u$ *(resp.* $k > u$*),* $h_k$ *is the vector of zeros except for the* $k$*'th (resp.* $(k - u)$*'th) element, which is equal to 1 (resp. -1).*

We note that one can reduce the number of constraints in these reformulations by a factor of $2u$ by using $d(\hat{x}, x) = d_z(\hat{z}, z)$, i.e., only penalizing the integer part of the decision vector, which is equivalent to setting $h_k = 0 \ \forall k \in [2u]$.

**Remark 4.7** (Generality of Theorem 4.5). *Reformulation* (19) *in Theorem 4.5, for IO problem with mixed-integer decision sets, generalizes several reformulations from the literature. For instance, for an IO problem with purely continuous decision sets and linear hypothesis functions, reformulation* (19) *with* $d_z(\hat{z}, z) = 0$ *and* $h_k = 0 \ \forall k \in [2u]$ *reduces to classical Inverse Linear Optimization reformulations [2, 19]. Similarly, if the IO problem has a purely continuous decision set and a quadratic hypothesis function, then the program* (19) *with* $d_z(\hat{z}, z) = 0$ *and* $h_k = 0 \ \forall k \in [2u]$ *reduces to the LMI reformulation in [3]. If the decision set is purely discrete, that is,* $\mathbb{X}(\hat{s})$ *has finitely many elements, then* (19) *and* (20) *recover* (15).*

In conclusion, by dualizing the continuous part of the problem, Theorem 4.5 and Corollary 4.6 present a way to deal with a case when the feasible set $\mathbb{X}(\hat{s})$ is a mixed-integer set. It is worth noting that when the feasible set $\mathbb{X}(\hat{s})$ is a mixed-integer set, the size of reformulations (19) and (20) grow linearly in the number of integer variables, in the dimension of the continuous variables, and in the size of the dataset. That means, for Inverse Optimization problems with large datasets and/or mixed-integer sets with a large number of decision variables, these reformulations can be computationally challenging to solve. This issue is the main focus of our tailored first-order algorithm in the next section.

## 5. Tailored Algorithm: Stochastic Approximate Mirror Descent

All approaches to solving IO problems presented so far depend on optimization programs with possibly a large number of constraints. Given a dataset $\{(\hat{s}_i, \hat{x}_i)\}_{i=1}^{N}$, these optimization problems have at least $\sum_{i=1}^{N} |\mathbb{X}(\hat{s}_i)|$ (or $\sum_{i=1}^{N} |\mathbb{Z}(\hat{w}_i)|$ for the mixed-integer case) constraints. When $N$ is too large (i.e., we have too many signal-response examples) or $|\mathbb{X}(\hat{s}_i)|$ is too large, solving these optimization programs may be intractable in practice. Thus, the focus of this section is to develop a tailored first-order algorithm to tackle such IO problems in a provably efficient way, opening doors to a wider range of real-world applications. A straightforward first-order approach to solve (14) is the standard *Project Subgradient method* [52]. This is a general first-order optimization algorithm, and it converges to an optimal solution of the problem under mild convexity conditions on the objective function and constraint set. For example, this was the approach used in [47] to solve the Maximum Margin Planning problem (see Remark 4.3). Another idea is to solve (14) as a minimax

problem. The paper [55] showed that in some special cases, this problem can be formulated as a bilinear saddle point problem, and then used Nesterov's dual extragradient method [43] to solve it. However, other than only being applicable to a restrictive class of problems, this approach also requires $2N$ projections onto $\mathbb{X}(\hat{s}_i)$ to be computed per iteration of the algorithm, which may be prohibitive in many applications. Other algorithms for minimax problems, such as Nemirovski's Mirror-Prox algorithm [42], would suffer from similar drawbacks. In this section, we introduce an efficient optimization algorithm that can be applied to general IO problems, while also exploiting the specific structures that stem from IO problems. To this end, we define the following notion of a stochastic approximate subgradient.

**Definition 5.1** (Stochastic approximate subgradient). *Let $f : \Theta \to \mathbb{R}$ be a convex function. We say that the random vector $\tilde{g}_\varepsilon(\theta)$ is a* stochastic approximate subgradient *of $f$ at $\theta$ if $\mathbb{E}\left[\tilde{g}_\varepsilon(\theta) \mid \theta\right] = g_\varepsilon(\theta)$ and $g_\varepsilon(\theta)$ is an $\varepsilon$-subgradient of $f$ for some $\varepsilon \geq 0$, that is,*

$$f(\theta) - f(\nu) \leq \langle g_\varepsilon(\theta), \theta - \nu \rangle + \varepsilon, \quad \forall \nu \in \Theta.$$

For different notions of approximate subgradients, see [48, 56]. Next, we propose to solve IO problems with a novel algorithm, the *Stochastic Approximate Mirror Descent* (SAMD). We first define the SAMD as an algorithm to optimize general convex programs and prove convergence rates for it. After that, we discuss how we can use the SAMD algorithm to exploit the structure of IO problems. For a function $f$ and set $\Theta$, the SAMD updates are

(SAMD) $$\theta_{t+1} = \operatorname*{argmin}_{\theta \in \Theta} \left\{ \eta_t \langle \tilde{g}_{\varepsilon_t}(\theta_t), \theta \rangle + \mathcal{B}_\omega(\theta, \theta_t) \right\},$$

where $\eta_t$ is the step-size, the function $\mathcal{B}_\omega$ is the Bregman divergence w.r.t. $\omega : \Theta \to \mathbb{R}$ [15, Section 4], and $\tilde{g}_{\varepsilon_t}(\theta_t)$ is a *stochastic approximate subgradient* of $f$ as defined in Definition 5.1. The SAMD algorithm can be interpreted as a combination of a stochastic mirror descent algorithm [15, Section 6.1] and an $\varepsilon$-subgradient method [9, Section 3.3]. Next, we prove a convergence rate for the SAMD algorithm, where we use the concept of *relative strong convexity*, which is a generalization of the standard strong convexity property [38]. Namely, if $\mathcal{R}$ is $\alpha$-strongly convex relative to $\mathcal{B}_\omega$, then $\mathcal{R}(x) - \mathcal{R}(y) \leq \langle g(x), x - y \rangle - \alpha \mathcal{B}_\omega(y, x)$, where $g(x) \in \partial \mathcal{R}(x)$.

**Proposition 5.2** (SAMD convergence rate). *Let $f : \Theta \to \mathbb{R}$ be a convex function, and $\Theta$ be a convex set. Assume $\mathcal{B}_\omega(\theta, \nu) \leq R^2$, for some $R > 0$, and $\mathbb{E}\left[\|\tilde{g}_{\varepsilon_t}(\theta)\|_*^2 \mid \theta\right] \leq G^2$, $\forall \theta, \nu \in \Theta$. Using $\eta_t = c/\sqrt{t}$ for some constant $c > 0$, the SAMD algorithm guarantees*

$$\mathbb{E}\left[f\left(\frac{1}{T}\sum_{t=1}^T \theta_t\right)\right] - \min_{\theta \in \Theta} f(\theta) \leq \left(\frac{R^2}{c} + cG^2\right)\frac{1}{\sqrt{T}} + \frac{1}{T}\sum_{t=1}^T \varepsilon_t.$$

*Moreover, if we assume $f = h + \mathcal{R}$, where $h$ is convex and $\mathcal{R}$ is $\alpha$-strongly convex relative to $\mathcal{B}_\omega$, then using $\eta_t = 2/\alpha(t+1)$, the SAMD algorithm guarantees*

$$\mathbb{E}\left[f\left(\frac{2}{T(T+1)}\sum_{t=1}^T t\theta_t\right)\right] - \min_{\theta \in \Theta} f(\theta) \leq \frac{2G^2}{\alpha(T+1)} + \frac{2}{T(T+1)}\sum_{t=1}^T t\varepsilon_t.$$

The convergence rate of the SAMD algorithm is a combination of the $O(1/\sqrt{T})$ (or $O(1/T)$) rate of the stochastic mirror descent algorithm plus a term that depends on $\{\varepsilon_t\}_{t=1}^T$ due to the use of approximate subgradients. Notice that, if we use the SAMD algorithm with errors $\varepsilon_t$ diminishing at a rate $O(1/t)$ (i.e., as $t$ increases, we use increasingly more precise approximate subgradients), then the convergence rates of Proposition 5.2 reduce to $O(1/\sqrt{T})$ and $O(1/T)$. The SAMD algorithm

differs from a simple projected subgradient method in three major ways: it uses (i) mirror descent updates, (ii) stochastic subgradients, and (iii) approximate subgradients. Next, we discuss how each of these properties can be used to exploit the structure of the IO problem (14).

## 5.1. **Mirror descent updates**

For instance, consider (14) with $\mathcal{R}(\theta) = \|\theta\|_1$ and $\Theta = \mathbb{R}^p$. This problem can be equivalently written as

$$
(21) \qquad
\begin{aligned}
&\min_{\theta \in \mathbb{R}^p} \quad \frac{1}{N} \sum_{i=1}^N \ell_\theta(\hat{s}_i, \hat{x}_i) \\
&\text{s.t.} \quad \tilde{\kappa}\|\theta\|_1 \leq 1,
\end{aligned}
$$

for some $\tilde{\kappa}$ that depends on $\kappa$ and the data $\{(\hat{s}_i, \hat{x}_i)\}_{i=1}^N$. Then, by introducing the nonnegative variables $\theta^+$ and $\theta^-$, defining $\tilde{\theta} := (\theta^+, \theta^-)$ (i.e., the concatenation of $\theta^+$ and $\theta-$), and setting $\theta = \theta^+ - \theta^- = [I \ -I]\tilde{\theta}$ (here, $I$ is the identity matrix and $[I \ -I]$ is a block matrix), (21) can be written as

$$
(22) \qquad \min_{\tilde{\theta} \in \Delta_{\tilde{\kappa}}} \frac{1}{N} \sum_{i=1}^N \ell_{[I \ -I]\tilde{\theta}}(\hat{s}_i, \hat{x}_i),
$$

where $\Delta_{\tilde{\kappa}} := \{\tilde{\theta} \in \mathbb{R}^{2p} : \tilde{\kappa}\|\tilde{\theta}\|_1 \leq 1, \tilde{\theta} \geq 0\}$ [58]. In other words, (21) can be recast as an optimization problem over a simplex. Next, choosing $\omega(\theta) = \sum_{i=1}^p \theta_i \log(\theta_i)$, the SAMD updates applied to (22) can be written as "exponentiated updates"

$$
\tilde{\theta}_{t+1} =
\begin{cases}
\tilde{\theta}_t \odot \exp(-\eta_t \tilde{g}_{\varepsilon_t}(\tilde{\theta}_t)) & \text{if} \quad \tilde{\kappa}\|\tilde{\theta}_t \odot \exp(-\eta_t \tilde{g}_{\varepsilon_t}(\tilde{\theta}_t))\|_1 \leq 1 \\
\frac{\tilde{\theta}_t \odot \exp(-\eta_t \tilde{g}_{\varepsilon_t}(\tilde{\theta}_t))}{\tilde{\kappa}\|\tilde{\theta}_t \odot \exp(-\eta_t \tilde{g}_{\varepsilon_t}(\tilde{\theta}_t))\|_1} & \text{otherwise.}
\end{cases}
$$

Exponentiated updates are known to have better convergence properties compared to standard subgradient descent updates for "simplex constrained" problems, as well as not requiring solving optimization problems to project onto the simplex. For a detailed discussion on the advantages of mirror descent updates for different settings, see [34, Section 5.7]. In general, by choosing $\omega(\theta) = \frac{1}{2}\|\theta\|_2^2$, the SAMD updates reduce to standard project subgradient updates

$$
\theta_{t+1} = \Pi_\Theta(\theta_t - \eta_t \tilde{g}_{\varepsilon_t}(\theta_t)),
$$

that is, the subgradient method can be seen as a special case of the mirror descent algorithm.

## 5.2. **Stochastic subgradients**

Using stochastic subgradients is advantageous when computing a stochastic subgradient is computationally cheaper than computing a deterministic subgradient. This observation is supported by the fact that, in expectation, the stochastic subgradient method guarantees the same convergence rate as its deterministic version [15, Theorem 6.1]. For the IO loss minimization problem (14), computing a stochastic subgradient can be significantly cheaper than computing the full subgradient, due to the finite sum structure of (14). Namely, by Danskin's theorem [8, Section B.5], we have that the subdifferential of the ASL w.r.t. $\theta$ is

$$
(23) \qquad \partial \ell_\theta(\hat{s}, \hat{x}) = \text{conv}\left\{ \phi(\hat{s}, \hat{x}) - \phi(\hat{s}, x^\star(\hat{s})) \,\middle|\, x^\star(\hat{s}) \in \underset{x \in \mathbb{X}(\hat{s})}{\arg\max} \left\{ d(\hat{x}, x) - \langle \theta, \phi(\hat{s}, x) \rangle \right\} \right\}.
$$

Thus, to compute a subgradient of $\frac{1}{N}\sum_{i=1}^{N}\ell_\theta(\hat{s}_i, \hat{x}_i)$, we need to solve $N$ maximization problems, one for each signal-response pair $\{(\hat{s}_i, \hat{x}_i)\}_{i=1}^{N}$. On the other hand, by sampling an index $j$ uniformly from $[N]$, we have that $g_j(\theta) \in \partial\ell_\theta(\hat{s}_j, \hat{x}_j)$ is a stochastic subgradient of $\frac{1}{N}\sum_{i=1}^{N}\ell_\theta(\hat{s}_i, \hat{x}_i)$, that is, $\mathbb{E}_{j\sim[N]}[g_j(\theta) \mid \theta] = (1/N)\sum_{i=1}^{N}g_i(\theta)$. This is a standard method for computing stochastic (sub)gradients for empirical risk minimization-type problems (e.g., [15, Chapter 6]). In summary, to compute an unbiased stochastic subgradient of (14), instead of $N$, we need to solve only *one* maximization problem.

## 5.3. Approximate subgradients

As shown in (23), we need to solve the optimization problem $\max_{x\in\mathbb{X}(\hat{s})}\{d(\hat{x}, x) - \langle\theta, \phi(\hat{s}, x)\rangle\}$ in order to compute a subgradient of the ASL. However, in practice, it may be too costly to solve this optimization problem to optimality at each iteration of the algorithm. Thus, it would be useful if we could use an approximate solution to this problem, instead of an optimal one. Turns out that, indeed, given an $\varepsilon$-*approximate solution* to the maximization problem, that is, a feasible point $x_\varepsilon$ such that

$$d(\hat{x}, x_\varepsilon) - \langle\theta, \phi(\hat{s}, x_\varepsilon)\rangle \geq \max_{x\in\mathbb{X}(\hat{s})}\{d(\hat{x}, x) - \langle\theta, \phi(\hat{s}, x)\rangle\} - \varepsilon,$$

we can construct an $\varepsilon$-subgradient of the ASL.

**Lemma 5.3** (Approximate solutions and $\varepsilon$-subgradients)**.** *Let $x_\varepsilon$ be a feasible, $\varepsilon$-suboptimal solution of $\max_{x\in\mathbb{X}(\hat{s})}\{d(\hat{x}, x) - \langle\theta, \phi(\hat{s}, x)\rangle\}$. Thus, the vector $g_\varepsilon(\theta) = \phi(\hat{s}, \hat{x}) - \phi(\hat{s}, x_\varepsilon)$ is an $\varepsilon$-subgradient of $\ell_\theta(\hat{s}, \hat{x})$ with respect to $\theta$, i.e.,*

$$\ell_\theta(\hat{s}, \hat{x}) - \ell_\nu(\hat{s}, \hat{x}) \leq \langle\phi(\hat{s}, \hat{x}) - \phi(\hat{s}, x_\varepsilon), \theta - \nu\rangle + \varepsilon.$$

In summary, we can use an approximate solution to the inner maximization problem of the ASL to construct $\varepsilon$-subgradients, and as proved in Proposition 5.2, these can be effectively used to compute an approximate solution of convex optimization problems. Finally, it is not difficult to show that by combining stochastic subgradients (i.e., by using only one random signal-response pair) and approximate subgradients (i.e., solving the respective maximization problem approximately), one gets a stochastic approximate subgradient (Definition 5.1). Putting these ideas together, Algorithm 1 shows the pseudocode of the SAMD algorithm applied to problem (14), where we denote the gradient (or a subgradient) of $\mathcal{R}$ as $\nabla\mathcal{R}$. In line 3 of Algorithm 1, one example is sampled from the dataset for each iteration of the algorithm. However, in practice, it may be advantageous to instead sample a batch of $1 \leq B \leq N$ examples, and use this batch of data to compute the approximate stochastic subgradient. By changing the size $B$ of the batch, we can control the trade-off between having more precise subgradients (large $B$) versus faster subgradient computations (small $B$). This idea is explored in the numerical experiments of Section 6.4.

We end this section by briefly discussing the case of *online IO*. In this scenario, instead of having a dataset of signal-response data upfront, we receive one signal-response pair at a time. After receiving each signal-response data pair, we must choose a cost vector $\theta_t$ using all the information gathered so far, and evaluate the loss of this cost vector. The final performance of the algorithm is measured using *regret* performance metrics.

**Remark 5.4** (Regret bounds and online IO)**.** *The authors in [7] use an online Multiplicative Weights Updates (MWU) algorithm to prove an $O(\sqrt{T})$ regret bound, and more recently, the authors in [12] use an online adaptation of the circumcenter concept to prove an $O(\log(T))$ regret bound. In*

---

**Algorithm 1** SAMD algorithm for (14)

---

1: **Input:** Step-size sequence $\{\eta_t\}_{t=1}^T$, $\kappa$, $\theta_1$, $d$, $\phi$, $\omega$, $\nabla\mathcal{R}$ and dataset $\{(\hat{s}_i, \hat{x}_i)\}_{i=1}^N$.

2: **for** $t = 1, \ldots, T$ **do**

3:     Sample $j$ uniformly from $\{1, \ldots, N\}$

4:     Compute $x_t$, a (possibly approximate) solution of $\max_{x \in \mathbb{X}(\hat{s}_j)} \{d(\hat{x}_j, x) - \langle\theta_t, \phi(\hat{s}_j, x)\rangle\}$

5:     Approximate stochastic subgradient: $\tilde{g}_t(\theta_t) = \kappa\nabla\mathcal{R}(\theta_t) + \phi(\hat{s}_j, \hat{x}_j) - \phi(\hat{s}_j, x_t)$

6:     Mirror descent step: $\theta_{t+1} = \text{argmin}_{\theta \in \Theta} \{\eta_t\langle\tilde{g}_t(\theta_t), \theta\rangle + \mathcal{B}_\omega(\theta, \theta_t)\}$

7: **end for**

8: **Output:** $\{\theta_t\}_{t=1}^T$

---

*Algorithm 1, line 3, if instead of sampling a new signal-response example we use the online data received at that iteration, Algorithm 1 can be readily adapted to online IO problems. Moreover, the convergence bound of Proposition 5.2 may also be straightforwardly converted to an $O(\sqrt{T})$ regret bound (or $O(\log(T))$ for the strongly convex case) [45, Chapter 3]. In particular, Algorithm 1 can be interpreted as a generalization of the MWU algorithm [4, Appendix A.2].*

## 6. NUMERICAL EXPERIMENTS

In this section, we numerically evaluate the approaches to inverse optimization proposed in this paper. All linear and quadratic programs were solved using Gurobi. All semidefinite programs were solved using CVXPY with MOSEK as the solver. For all results presented in this section, we learn the expert's cost function using a training dataset of expert signal-response data and evaluate its performance using a test dataset (i.e., out-of-sample performance). Moreover, in every plot, we report the average performance value for 10 randomly generated true cost vectors, as well as the 5th and 95th percentile bounds. In Appendix D.2, the in-sample results are reported, complementing the ones from this section. All of our experiments are reproducible and are part of the examples in the InvOpt Python package [63].

### 6.1. **Consistent data**

In this section, we numerically evaluate the approaches discussed in Section 3 for IO problems with consistent data.

**Decision problem of the expert.** To generate its decisions, the expert solves the binary linear program

$$(24) \quad \begin{aligned} \min_x \quad & \langle\theta, x\rangle \\ \text{s.t.} \quad & Ax \leq b \\ & x \in \{0, 1\}^n, \end{aligned}$$

where $\theta \in \mathbb{R}_+^n$ is the cost vector, $A \in \mathbb{R}^{t \times n}$, $b \in \mathbb{R}^t$, and $x$ is the decision vector. Although the decision problem (24) is presented as a general binary linear program, many real-world problems can be modeled within this class of optimization problems. As a motivating example, we briefly discuss the problem of modeling consumer behavior, and how it fits (24). In this problem, the expert is a consumer, who given some contextual information, decides which products to buy from a set of $n$ products. That is, each component of the decision vector $x \in \{0, 1\}^n$ corresponds to one of the $n$ products and equals 1 if the consumer buys it, and 0 otherwise. The signal with contextual

information could be, for instance, $\hat{s} = (A, b)$, where each component of $A \in \mathbb{R}^n$ corresponds to the price of each product, and $b \in \mathbb{R}$ corresponds to the total budget of the consumer. In this case, the constraint $Ax \leq b$ simply represents the budget constraint of the consumer. Finally, each component of the cost vector $\theta \in \mathbb{R}^n$ represents (the negative of) the utility the consumer assigns to each product, and by minimizing $\langle \theta, x \rangle$, the goal of the consumer is to buy the set of products that maximizes its utility, while respecting its budget constraint. Thus, in this context, solving the IO problem translates to learning the utility function underlying the actions of a consumer agent.

**Data generation.** To generate training and test data, we sample cost vectors uniformly from $\{\theta \in \mathbb{R}^n : 0 \leq \theta \leq \mathbb{1}\}$, we sample $\hat{A}$ uniformly from $\{A \in \mathbb{R}^{t \times n} : -1 \leq A \leq 0\}$ and $\hat{b}$ uniformly from $\{b \in \mathbb{R}^t : -\mathbb{1} \leq b \leq 0\}$. To make sure the problem instances are feasible, we check if the sum of each row of $\hat{A}$ is larger than the respective component of $\hat{b}$, which ensures that $x = (1, \dots, 1)$ is a feasible solution of (24). Given a signal $\hat{s} = (\hat{A}, \hat{b})$, we generate a response $\hat{x}$ by solving (24), i.e., $\hat{x} \in \operatorname{argmin}_{x \in \mathbb{X}(\hat{s})} \langle \theta, x \rangle$. To evaluate the IO approaches, we generate 10 random cost vectors $\theta_{\text{true}}$. For each of these cost vectors, we generate a training dataset $\widehat{\mathcal{D}}_{\text{train}} = \{(\hat{s}_i, \hat{x}_i)\}_{i=1}^N$ and a test dataset $\widehat{\mathcal{D}}_{\text{test}} = \{(\hat{s}_i, \hat{x}_i)\}_{i=1}^N$, with $N = 100$, $n = 6$ and $t = 4$.

**IO approaches.** We compare seven approaches to choose a vector from the set of consistent cost vectors (4):

- **Feasibility**: we use (5) with $\|\theta\|_1 = 1$, $s = (A, b)$, $\mathbb{X}(s) = \{x \in \{0, 1\}^n : Ax \leq b\}$, $\phi(x, s) = x$, augmented with the constraint $\theta \in \Theta = \{\theta \in \mathbb{R}^n : \theta \geq 0\}$;
- **Incenter**: we use (13) with $\mathcal{R}(\theta) = \frac{1}{2}\|\theta\|_2^2$, $s = (A, b)$, $\mathbb{X}(s) = \{x \in \{0, 1\}^n : Ax \leq b\}$, $\phi(x, s) = x$, $d(\hat{x}_i, x_i) = \|\hat{x}_i - x_i\|_2$, and $\Theta = \{\theta \in \mathbb{R}^n : \theta \geq 0\}$;
- **Circumcenter**: we solve the following optimization problem:

(25)
$$\begin{aligned} \min_{\|\theta\|_2 = 1} \quad & r \\ \text{s.t.} \quad & \max_{\substack{\tilde{\theta} \in \mathbb{C} \\ \|\tilde{\theta}\|_2 = 1}} \|\theta - \tilde{\theta}\|_2^2 \leq r, \end{aligned}$$

which is an epigraph reformulation of the circumcenter problem (see the proof of Theorem 3.2). To solve this optimization problem, we substitute the max constraint by the constraints $\|\theta_E - \tilde{\theta}\|_2^2 \leq r$, $\forall \theta_E \in E$, where $E$ is the set of normalized extreme points of $\mathbb{C}$;

- **Ellip. incenter**: we solve (12), the ellipsoidal version of the incenter concept. In our implementation, we use the convex constraint $\|\theta\| \leq 1$ instead of $\|\theta\| = 1$, since one can show that the constraint $\|\theta\| \leq 1$ will always be tight at an optimum when $\operatorname{int}(\mathbb{C}) \neq \emptyset$;
- **Ellip. circumcenter**: we solve

$$\begin{aligned} \max_{A \succcurlyeq 0, \theta} \quad & \log \det(A) \\ \text{s.t.} \quad & \max_{\substack{\tilde{\theta} \in \mathbb{C} \\ \|\tilde{\theta}\|_2 = 1}} \|A(\tilde{\theta} - \theta)\|_2^2 \leq 1, \end{aligned}$$

which is an ellipsoidal generalization of the circumcenter reformulation (25);
- **Cutting plane**: we use the cutting plane algorithm of [61];
- **Predictability loss**: we use the predictability loss of [6] (Appendix A).

**Results.** Figure 3 shows the results for this scenario. To make the comparisons consistent, before evaluating the results, all cost vectors are normalized. For all the plots, the x-axis refers to the number of training examples used to compute the results. The idea is to evaluate how efficient

| Approach | Feasibility | Incenter | Circumcenter | Ellip. incenter |
|---|---|---|---|---|
| **Time (seconds)** | 69 | 98 | 1323 | 915 |
| **Approach** | Ellip. circumcenter | Cutting plane | Predictability loss | |
| **Time (seconds)** | 1345 | 206 | 236 | |

Table 1. Computational time to generate the results of Figure 3.

these approaches are with respect to the amount of data fed to them. Figure 3a shows the difference between the cost vector returned by the IO approaches (which we name $\theta_{\text{IO}}$) and the cost vector used to generate the data ($\theta_{\text{true}}$). As expected, the more data we feed to the approaches, the closer $\theta_{\text{IO}}$ tends to get from $\theta_{\text{true}}$. Comparing them, the incenter, ellipsoidal incenter, and ellipsoidal circumcenter approaches have similar performance and clearly outperform the other approaches. Figure 3b shows the average difference between the optimal decision of (24) using $\theta_{\text{IO}}$ (which we name $x_{\text{IO}}$) and the decision in the test dataset (which we name $x_{\text{true}}$). Again, we see that the incenter, ellipsoidal incenter, and ellipsoidal circumcenter approaches present the best performance. Figure 3c shows the normalized difference between the cost of the expert decisions and the cost of the decisions using $\theta_{\text{IO}}$. More precisely, we define $\text{Cost}_{\text{IO}} \coloneqq \sum_{i=1}^{N} \langle \theta_{\text{true}}, x_{\text{IO},i} \rangle$ and $\text{Cost}_{\text{true}} \coloneqq \sum_{i=1}^{N} \langle \theta_{\text{true}}, \hat{x}_i \rangle$ and compare the relative difference between them. Notice that this difference will always be nonnegative by the optimality of $\hat{x}_i$. Once again, the incenter, ellipsoidal incenter, and ellipsoidal circumcenter approaches outperform the other approaches.

Moreover, Table 1 shows the time it took to generate the results of this section for each approach. As can be seen, even though the incenter, ellipsoidal incenter, and ellipsoidal circumcenter approaches show similar performance, the incenter approach is at least one order of magnitude faster to compute. Although these numbers depend on the actual implementation of each IO approach, such a difference in solving time is expected, since the incenter problem can be formulated as a quadratic program, whereas the ellipsoidal approaches involve semidefinite constraints. Finally, regarding the main driver for improved out-of-sample performance of the Incenter and the ellipsoidal approaches, one possible explanation is that these approaches optimize for a vector in the interior of $\mathbb{C}$, that is, away from the boundaries of this set. This conclusion is based on the following observations: recall the intuition for the incenter vector provided in Remark 3.4, also visualized in Figure 1b. That is, the idea behind the incenter is to find the vector furthest away from the boundaries of the set $\mathbb{C}$. Also, recall that the authors in [12] showed that the circumcenter concept fails for online IO problems precisely because, in the worst case, the circumcenter vector can lie exactly in the boundary of the set $\mathbb{C}$ [12, Figure 2], and the ellipsoidal circumcenter addresses this problem since its circumcenter lies inside the set $\mathbb{C}$ [12, Figure 3]. Therefore, the incenter, ellipsoidal incenter, and ellipsoidal circumcenter in some sense optimize for vector in the interior of $\mathbb{C}$, and as shown in Figure 3, achieve similar out-of-sample performance in this offline IO experiment (although the incenter vector is computationally cheaper to compute).

## 6.2. Inconsistent data

In this section, we numerically evaluate the approaches discussed in Section 4 for IO problems with inconsistent data.

**Decision problem of the expert.** To generate its decisions, the expert solves the binary linear program (24), where $\theta \in \mathbb{R}^n$ is the cost vector, $A \in \mathbb{R}^{t \times n}$ and $b \in \mathbb{R}^t$, and $x$ is the decision vector. Notice that different from the previous section, we do not assume the cost vector $\theta$ is nonnegative.
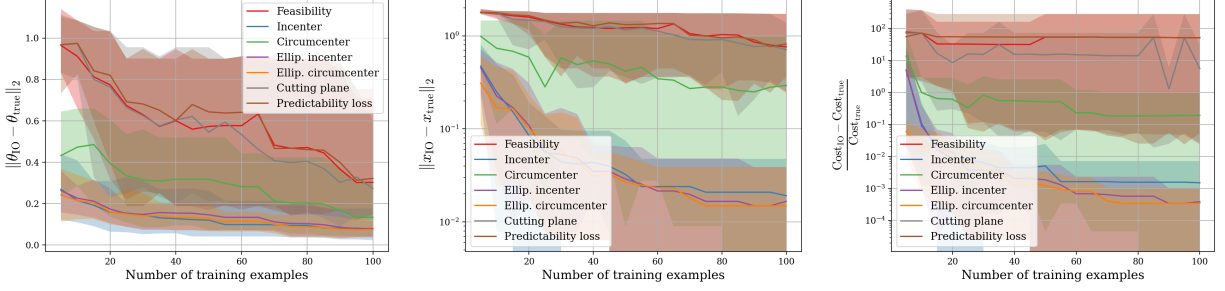
(a) Difference between the true cost vector $(\theta_{\text{true}})$ and the one learned using IO $(\theta_{\text{IO}})$.

(b) Average error between the decision generated by $\theta_{\text{true}}$ and $\theta_{\text{IO}}$.

(c) Relative difference between the cost of the decisions generated using $\theta_{\text{true}}$ and $\theta_{\text{IO}}$.

Figure 3. Out-of-sample results for consistent data scenario.

**Data generation.** To generate training and test data, we sample cost vectors uniformly from $\{\theta \in \mathbb{R}^n : -\mathbb{1} \leq \theta \leq \mathbb{1}\}$, we sample $\hat{A}$ uniformly from $\{A \in \mathbb{R}^{t \times n} : -1 \leq A \leq 1\}$ and $\hat{b}$ uniformly from $\{b \in \mathbb{R}^t : -\mathbb{1} \leq b \leq 0\}$. After generating a signal $\hat{s} = (\hat{A}, \hat{b})$, we check if $\mathbb{X}(\hat{s})$ is nonempty to ensure the problem instance is feasible. To generate the response vectors $\hat{x}$ for the training datasets, we solve problem (24) with noise added to the cost vector, i.e., $\hat{x} \in \arg\min_{x \in \mathbb{X}(\hat{s})} \langle \theta + w, x \rangle$, where $w \in \mathbb{R}^n$ is a random vector with components sampled from a normal distribution with zero mean and standard deviation equal to 0.05. This means that different from the consistent data case, (most probably) there will be no single cost vector consistent with the entire training dataset. To evaluate the IO approaches, we generate 10 random cost vectors $\theta_{\text{true}}$. For each of these cost vectors, we generate a training dataset $\widehat{\mathcal{D}}_{\text{train}} = \{(\hat{s}_i, \hat{x}_i)\}_{i=1}^N$ (by solving the noisy version of (24), with a different noise vector $w$ for each signal-response pair) and a test dataset $\widehat{\mathcal{D}}_{\text{test}} = \{(\hat{s}_i, \hat{x}_i)\}_{i=1}^N$ (noiseless), with $N = 100$, $n = 10$ and $t = 8$.

**IO approaches.** We compare five IO approaches for this problem:

- **Suboptimality Loss (SL)**: we use (14) with $\kappa = 0$, $\Theta = \mathbb{R}^n$, $s = (A, b)$, $\mathbb{X}(s) = \{x \in \{0, 1\}^n : Ax \leq b\}$, $\phi(x, s) = x$, and the suboptimality loss $\ell_\theta(\hat{s}, \hat{x}) = \max_{x \in \mathbb{X}(\hat{s})} \{\langle \theta, \phi(\hat{s}, \hat{x}) - \phi(\hat{s}, x) \rangle\}$. To prevent the trivial solution $\theta = 0$, we add a norm equality constraint $\|\theta\|_\infty = 1$, and solve $2n$ linear programs, one for each facet of the $\infty$-norm unit sphere [39];

- **Augmented Suboptimality Loss (ASL)**: we use (14) with the ASL, $\kappa = 0.001$, $\mathcal{R}(\theta) = \frac{1}{2}\|\theta\|_2^2$, $s = (A, b)$, $\mathbb{X}(s) = \{x \in \{0, 1\}^n : Ax \leq b\}$, $\phi(x, s) = x$, $d(\hat{x}_i, x_i) = \|\hat{x}_i - x_i\|_2$, and $\Theta = \mathbb{R}^n$;

- **Ellipsoidal ASL**: we solve

$$
(26) \quad \begin{aligned}
\min_{A, \theta, \beta_1, \ldots, \beta_N} \quad & -\kappa \log\left(\det\left(A\right)\right) + \frac{1}{N} \sum_{i=1}^N \beta_i \\
\text{s.t.} \quad & \langle \theta, \phi(\hat{s}_i, \hat{x}_i) - \phi(\hat{s}_i, x_i) \rangle + \|A(\phi(\hat{s}_i, x_i) - \phi(\hat{s}_i, \hat{x}_i))\|_2 \leq \beta_i \quad \forall x_i \in \mathbb{X}(\hat{s}_i), \; \forall i \in [N] \\
& \|\theta\|_2 \leq 1, \quad A \succeq 0,
\end{aligned}
$$

which is an ellipsoidal generalization of (15) (see Remark 3.7).

- **Cutting plane**: we use the cutting plane method of [13], which is a extension of the cutting-plane algorithm of [61] for IO problems with inconsistent data;

- **Predictability loss**: we use the predictability loss of [6] (Appendix A).

| Approach | SL | ASL | Ellipsoidal ASL | Cutting plane | Predictability loss |
|---|---|---|---|---|---|
| Time (seconds) | 230 | 241 | 351 | 440 | 6192 |

Table 2. Computational time to generate the results of Figure 4.
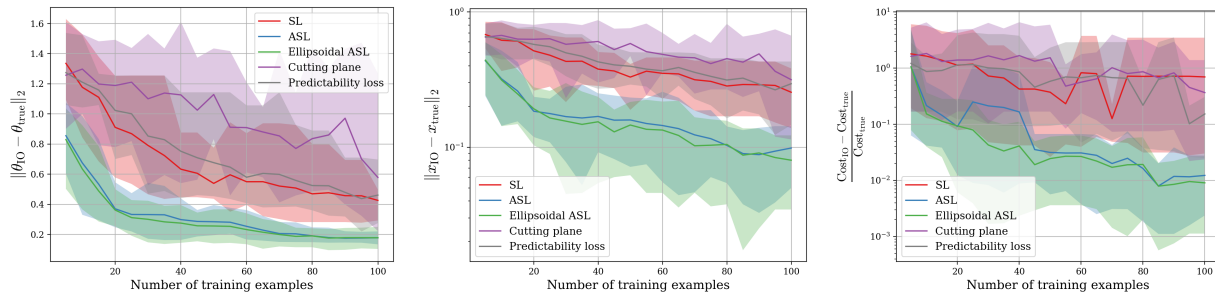


(a) Difference between the true cost vector ($\theta_{\text{true}}$) and the one learned using IO ($\theta_{\text{IO}}$).

(b) Average error between the decision generated by $\theta_{\text{true}}$ and $\theta_{\text{IO}}$.

(c) Relative difference between the cost of the decisions generated using $\theta_{\text{true}}$ and $\theta_{\text{IO}}$.

Figure 4. Out-of-sample results for inconsistent data scenario.

Notice that we do not test circumcenter-based approaches since they are not defined for the inconsistent data case, that is, when $\mathbb{C} \setminus \{0\} = \emptyset$.

**Results.** Figure 4 shows the results this scenario. The discussion on the interpretation of each performance metric presented in Figure 4 mirrors the one of Figure 3. Importantly, the approach based on the ASL and its ellipsoidal version outperforms the other approaches. Moreover, as expected, due to the noise in the training data, one can see that $\theta_{\text{IO}}$ was not able to reproduce the behavior of the expert as well as in the consistent (i.e., noiseless) case. In Table 1 we show the time it took to generate the results of this section for each approach. As can be seen, even though the ASL and Ellipsoidal ASL show similar performance, the ASL approach is more computationally efficient. Again, this difference is because the Ellipsoidal ASL involves semidefinite constraints (Eq. (26)), which is not the case for the standard ASL (Eq. (15)).

## 6.3. Mixed-integer feasible set

In this section, we numerically evaluate the approach from Section 4.2 for IO problems with mixed-integer feasible sets.

**Decision problem of the expert.** To generate its decisions, the expert solves a mixed-integer linear program of the form

$$
(27) \quad \begin{aligned} \min_{y,z} \quad & \langle q_y, y \rangle + \langle q_z, z \rangle \\ \text{s.t.} \quad & Ay + Bz \leq c \\ & 0 \leq y \leq \mathbb{1}, \ z \in \{0,1\}^v, \end{aligned}
$$

where $\theta := (q_y, q_z) \in \mathbb{R}_+^{u+v}$ is the cost vector $A \in \mathbb{R}^{t \times u}$, $B \in \mathbb{R}^{t \times v}$ and $b \in \mathbb{R}^t$.

**Data generation.** To generate training and test data, we sample the cost vector uniformly from $\{\theta \in \mathbb{R}^{u+v} : 0 \leq \theta \leq \mathbb{1}\}$, we sample $\hat{A}$ uniformly from $\{A \in \mathbb{R}^{t \times u} : -1 \leq A \leq 0\}$, $\hat{B}$ uniformly from $\{B \in \mathbb{R}^{t \times v} : -1 \leq B \leq 0\}$ and $\hat{c}$ uniformly from $\{c \in \mathbb{R}^t : -2 \leq c \leq 0\}$. To make sure the problem instances are feasible, we checked if the sum of each row of $[\hat{A} \ \hat{B}]$ is smaller than the respective

| Approach | SL | ASL-z | ASL-yz | Circumcenter | Cutting plane | Predictability loss |
|---|---|---|---|---|---|---|
| **Time (seconds)** | 607 | 694 | 7060 | 5954 | 981 | 1336 |

Table 3. Computational time to generate the results of Figure 5.
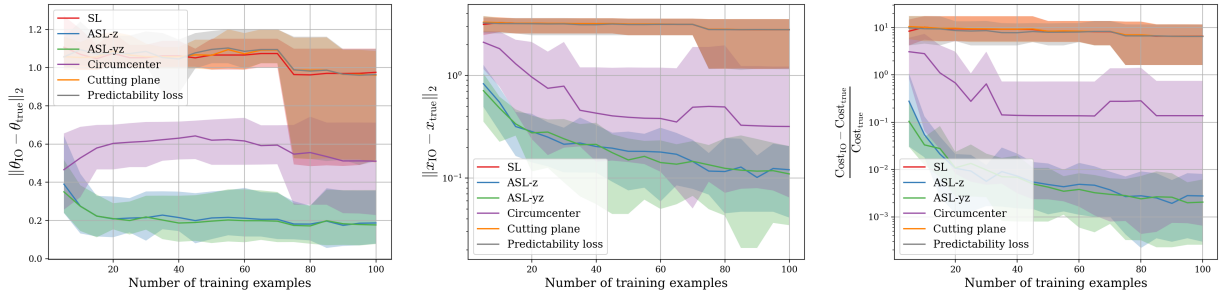


(a) Difference between the true cost vector ($\theta_{\text{true}}$) and the one learned using IO ($\theta_{\text{IO}}$).

(b) Average error between the decision generated by $\theta_{\text{true}}$ and $\theta_{\text{IO}}$.

(c) Relative difference between the cost of the decisions generated using $\theta_{\text{true}}$ and $\theta_{\text{IO}}$.

Figure 5. Out-of-sample results for the mixed-integer feasible set scenario.

component of $\hat{c}$. Given a tuple $(\hat{A}, \hat{B}, \hat{c})$, we generate a response $\hat{x} = (\hat{y}, \hat{z})$ by solving problem (27). To evaluate the IO approaches, we generate 10 random cost vectors $\theta_{\text{true}}$. For each of these cost vectors, we generate a training dataset $\widehat{\mathcal{D}}_{\text{train}} = \{(\hat{s}_i, \hat{x}_i)\}_{i=1}^N$ and a test dataset $\widehat{\mathcal{D}}_{\text{test}} = \{(\hat{s}_i, \hat{x}_i)\}_{i=1}^N$, with $N = 100$, $u = v = 6$ and $t = 4$.

**IO approaches.** We compare six IO approaches for this problem:

- **Suboptimality Loss (SL)**: we use (20) with no regularization $\mathcal{R}$ and no distance function (i.e., $\kappa = 0$ and $d(\hat{x}_i, x_i) = 0$). To avoid the trivial solution $\theta = 0$, we add the constraint $\|\theta\|_1 = 1$. We call this approach SL since it is the result of performing the reformulation steps of Corollary 4.6 using the Suboptimality loss instead of the ASL;
- **ASL-yz**: we use (20) with $\theta = (q_y, q_z)$, $s = (A, B, c, 0)$ ($A$ and $c$ have to be augmented to account for the constraints $0 \le y \le \mathbb{1}$), $\mathbb{Z}(w) = \{0, 1\}^v$, $Q_{yy} = 0$, $\phi_1(w, z) = 1$, $\phi_2(w, z) = z$, $\kappa = 0$, $\Theta = \{\theta \in \mathbb{R}^{u+v} : \theta \ge 0\}$, and $d_z(\hat{z}_i, z_i) = \|\hat{z}_i - z_i\|_2$.
- **ASL-z**: same as **ASL-yz**, but with $h_k = 0$ $\forall k \in [2u]$ (see discussion in the paragraph after Corollary 4.6);
- **Circumcenter**: we solve (25);
- **Cutting plane**: we use the cutting plane algorithm of [61];
- **Predictability loss**: we use the predictability loss of [6] (Appendix A).

**Results.** Figure 5 shows the results for this scenario. The discussions on the interpretations of the results of this section mirror the ones of Figure 3 from Section 6.1. Once again, the approach based on the ASL outperforms the other approaches on all three performance metrics. In particular, the ASL-z and ASL-yz show similar performance, with the ASL-yz being slightly better overall and when there are very few training examples. However, because the ASL-yz optimization problem has $2N$ times more constraints than the ASL-z optimization problem, it is much more costly to solve, as can be seen from the computational times in Table 3.

## 6.4. Stochastic approximate mirror descent

In this section, we numerically evaluate the SAMD algorithm proposed in Section 5.

**Decision problem of the expert.** To generate its decisions, the expert solves the binary linear program (24), where $\theta \in \mathbb{R}_+^n$ is the cost vector, $A \in \mathbb{R}^{t \times n}$, $b \in \mathbb{R}^t$, and $x$ is the decision vector. We can write this optimization problem as $\min_{x \in \mathbb{X}(s)} F(s, x)$ by defining $s := (A, b)$, $F(s, x) := \langle \theta, x \rangle$ and $\mathbb{X}(s) := \{x \in \{0, 1\}^n : Ax \geq b\}$.

**Data generation.** To generate training and test data, we sample cost vectors uniformly from $\{\theta \in \mathbb{R}^n : 0 \leq \theta \leq \mathbb{1}\}$, we sample $\hat{A}$ uniformly from $\{A \in \mathbb{R}^{t \times n} : -1 \leq A \leq 0\}$ and $\hat{b}$ uniformly from $\{b \in \mathbb{R}^t : -\frac{n}{3} \leq b \leq 0\}$. To make sure the problem instances are feasible, we checked if the sum of each row of $\hat{A}$ is larger than the respective component of $\hat{b}$. Given a signal $\hat{s}$, we generate a response $\hat{x}$ by solving (24), i.e., $\hat{x} \in \operatorname{argmin}_{x \in \mathbb{X}(\hat{s})} \langle \theta, x \rangle$. To evaluate the IO approaches, we generate 10 random cost vectors $\theta_{\text{true}}$. For each of these cost vectors, we generate a training dataset $\widehat{\mathcal{D}}_{\text{train}} = \{(\hat{s}_i, \hat{x}_i)\}_{i=1}^N$ and a test dataset $\widehat{\mathcal{D}}_{\text{test}} = \{(\hat{s}_i, \hat{x}_i)\}_{i=1}^N$, with $N = 50$, $n = 20$ and $t = 15$.

**IO approaches.** We tackle this IO problem using (14) with $\kappa = 0.01$, $\mathcal{R}(\theta) = \|\theta\|_1$, $\Theta = \{\theta \in \mathbb{R}^n : \theta \geq 0\}$ and the ASL with $s = (A, b)$, $\mathbb{X}(s) = \{x \in \{0, 1\}^n : Ax \leq b\}$, $\phi(s, x) = x$ and $d(\hat{x}, x) = \|x - \hat{x}\|_1$. When testing algorithms with exponentiated updates (i.e., mirror descent updates with $\omega(\theta) = \sum_{i=1}^n \theta_i \log(\theta_i)$), we solve the reformulation (21), with $\tilde{\kappa}$ chosen such that (14) and (21) have the same optimal solution. We compare eight algorithms, which result from all possible combinations of standard or mirror descent updates, deterministic or stochastic subgradients, and exact or approximate subgradients.

(i) **Subgradient method (SM)**: Algorithm 1 with $\omega(\theta) = \frac{1}{2}\|\theta\|_2^2$ and exact subgradients computed using the entire dataset;

(ii) **Mirror descent (MD)** with exponentiated updates: Algorithm 1 with $\omega(\theta) = \sum_{i=1}^n \theta_i \log(\theta_i)$ and exact subgradients computed using the entire dataset;

(iii) **Stochastic subgradient method (SSM)**: Algorithm 1 with $\omega(\theta) = \frac{1}{2}\|\theta\|_2^2$ and exact stochastic subgradients.

(iv) **Approximated subgradient method (ASM)**: Algorithm 1 with $\omega(\theta) = \frac{1}{2}\|\theta\|_2^2$ and approximate subgradients computed using the entire dataset.

(v) **Stochastic mirror descent (SMD)** with exponentiated updates: Algorithm 1 with $\omega(\theta) = \sum_{i=1}^n \theta_i \log(\theta_i)$ and exact stochastic subgradients.

(vi) **Approximate mirror descent (AMD)** with exponentiated updates: Algorithm 1 with $\omega(\theta) = \sum_{i=1}^n \theta_i \log(\theta_i)$ and and approximate subgradients computed using the entire dataset.

(vii) **Stochastic approximate subgradient method (SASM)**: Algorithm 1 with $\omega(\theta) = \frac{1}{2}\|\theta\|_2^2$ and approximate stochastic subgradients.

(viii) **Stochastic Approximate Mirror Descent (SAMD)** with exponentiated updates, i.e., Algorithm 1 with $\omega(\theta) = \sum_{i=1}^n \theta_i \log(\theta_i)$ and approximate stochastic subgradients.

For all algorithms, we use $\eta_t = 1/(\|\tilde{g}_{\varepsilon_t}(\theta_t)\|_* \sqrt{t})$. To compute approximate subgradients for the ASM and SAMD algorithms, we give the solver (in our case, Gurobi) a time limit of 0.03 seconds to solve the optimization problem in line 4 of Algorithm 1. If the solver is not able to find an optimal solution within this time limit, it returns the best feasible solution found.

**Results.** To compare the performance of the algorithms, we report their results in terms of running time instead of the number of iterations. Figure 6a shows the convergence of the proposed algorithms in terms of the training loss gap. More precisely, defining $f(\theta) := \kappa \mathcal{R}(\theta) + \frac{1}{N} \sum_{i=1}^N \ell_\theta(\hat{s}_i, \hat{x}_i)$,
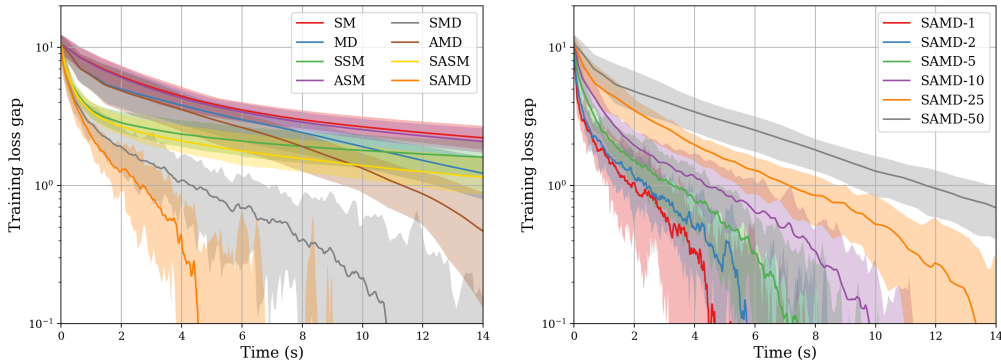
(a) Convergence of the first-order optimization algorithms.

(b) Convergence of the SAMD algorithms for different sizes of batch sizes.

Figure 6. Convergence time of different first-order optimization algorithms and for different batch sizes.

the training loss gap of some $\theta_t$ is defined as $f(\theta_t) - \min_{\theta \in \Theta} f(\theta)$. From this plot, it is clear that each modification of the standard subgradient method (i.e., mirror descent updates, stochastic subgradients, and approximate subgradients) contributes to improving the convergence speed of the algorithms, with the fastest convergence achieved by the combination of all of these improvements, i.e., the SAMD algorithm. In Figure 6b, we show the convergence of the SAMD using different batch sizes to compute stochastic subgradients. That is, instead of using only one sampled example at each iteration (line 3 of Algorithm 1), we sample a batch of $B$ examples and use this batch of data to compute the stochastic subgradient, which we call SAMD-$B$. From Figure 6b we can see that even though smaller batches lead to more variance in the convergence of the algorithm (as expected) it also leads to a faster empirical convergence for this experiment.

## 7. Further Discussions and Potential Applications

In this work, we present new approaches to tackle inverse optimization problems. Based on the geometry of the set of consistent cost vectors, we propose the use of an incenter vector of this set as the solution to the IO problem. Moreover, we propose a new loss function for IO problems: the Augmented Suboptimality Loss. This loss can be interpreted as a relaxation of the incenter approach to tackle problems with inconsistent data. Moreover, this loss can be used to derive IO approaches tailored for problems with mixed-integer feasible sets and can be optimized directly using first-order methods. For the latter case, we propose new algorithms that combine stochastic, approximate, and mirror descent updates, all of which can be used to exploit the structure of IO losses. Finally, we numerically evaluate the approaches proposed in this paper and show that they can outperform state-of-the-art methods in a variety of scenarios.

The work presented in this paper opens several avenues for future research. For instance, one could try different ways to choose a vector from the set of consistent costs, for example, using ellipsoidal cones instead of circular cones. This idea was used in online IO algorithms (see [12, Section 4.3]), but perhaps it can also be leveraged for better empirical performance in the offline case. Also, one could try to adapt and further develop the incenter approaches presented in this paper to the online IO scenario, perhaps leveraging the geometry of the IO problem to prove tighter regret bounds in some specific scenarios, akin to [12]. Another idea is to come up with different algorithms to tackle the

optimization problems discussed in this paper, for instance, using cutting-plane/bundle methods, extending the related literature of structured prediction problems [61, 33, 57]. Moreover, given the wealth of different first-order methods in the literature, we feel like designing specialized first-order (or even proximal or second-order) algorithms tailored for inverse optimization problems is an under-explored research direction. Given the newly established bridge between IO and structured prediction problems (see Remark 4.3), we foresee that many interesting results can be achieved by combining and extending the literature of these two, mostly disjoint, communities. Finally, it would be interesting to extend the numerical experiments of this paper using real-world data, for instance, data from transportation/routing problems, e.g., as was recently done in [64] in the context of the Amazon Challenge [5].

## Appendix A. Theoretical Properties of the Augmented Suboptimality Loss

Define $F_{\theta,\phi}(s,x) \coloneqq \langle \theta, \phi(s,x) \rangle$. In this section, we discuss several theoretical properties of the Augmented Suboptimality Loss (ASL), and show how it can be interpreted as a convex surrogate of the *Generalized Predictability Loss* (GPL). The GPL is a natural loss function for IO problems and is defined as follows.

**Definition A.1** (Generalized Predictability Loss). *Given a signal-response pair* $(\hat{s}, \hat{x})$, *the Generalized Predictability Loss of a cost vector* $\theta$ *is given by*

(28)
$$\ell_{\theta}^{\mathrm{pred}}(\hat{s}, \hat{x}) \coloneqq \min_{x} \ d(\hat{x}, x)$$
$$\text{s.t.} \quad x \in \operatorname*{argmin}_{y \in \mathbb{X}(\hat{s})} F_{\theta,\phi}(\hat{s}, y),$$

*where* $d : \mathbb{X} \times \mathbb{X} \to \mathbb{R}_+$ *is a distance function.*

The GPL has an intuitive interpretation for IO problems: given a signal $s$, it computes how close (in terms of $d$) are $\hat{x}$ (the expert's response) and $x$ (an optimal response according to $F_{\theta,\phi}$). Consequently, by solving (14) using the GPL, we optimize for a cost $F_{\theta,\phi}$ that best reproduces the responses taken by the expert for each signal $s$. For the special case when $d(x,y) = \|x - y\|_2^2$, the GPL reduces to the so-called *predictability loss*. Unfortunately, when using the GPL, problem (14) is an NP-hard bi-level optimization problem in general [6].

To come up with losses that are convex w.r.t. $\theta$, and still meaningful for IO problems, we can use the concept of *convex surrogate functions*. Informally, given a nonconvex function $g$, a convex surrogate function $h$ is a convex upper bound of $g$. The idea is then to minimize $h$ instead of $g$. The hope is that by minimizing $h$, we also minimize (at least to some extent) the original nonconvex function $g$. Interestingly, we can show that the ASL is a convex surrogate for the GPL and that it possesses several properties attractive for IO loss functions. In particular, the properties of the ASL shown in Proposition A.3 are similar to (and in some sense generalizations of) the properties of the Suboptimality Loss presented in [39].

**Assumption A.2** (Optimizer condition). *Let the distance function* $d : \mathbb{X} \times \mathbb{X} \to \mathbb{R}_+$ *be such that* $d(x,y) = 0$ *if and only if* $x = y$. *Then,* $\forall x \in \mathbb{X}(s)$ *and* $\forall s \in \mathbb{S}$, *there exists a* $\theta \in \Theta$ *such that*

$$y^\star \in \operatorname*{argmin}_{y \in \mathbb{X}(s)} F_{\theta,\phi}(s,y) \implies F_{\theta,\phi}(s,x) - F_{\theta,\phi}(s,y^\star) \geq d(x, y^\star).$$

Assumption A.2 can be interpreted as a "unique minimizer" condition and it is necessary to prove the "consistency" property of Proposition A.3. This is a strong assumption and does not hold for

general IO problems. However, this assumption is not critical for the algorithm design and analysis presented in this paper. Moreover, empirically, IO algorithms based on the ASL perform well even for IO problems that do not possess the "unique minimizer" property (for example, see the numerical results of Section 6). Finally, we note that Assumption A.2 can be interpreted as a generalization of other assumptions from the literature, which are also used to prove consistency-like properties of loss functions. We list three such examples, where we use the notation $y^\star \in \operatorname{argmin}_{y \in \mathbb{X}(s)} F_{\theta,\phi}(s, y)$.

- Let $d(x, y) = I(x, y)$, the 0-1 distance. In this case, Assumption A.2 reads as

$$F_{\theta,\phi}(s, x) - F_{\theta,\phi}(s, y^\star) \geq 1 \quad \forall (s, x) \in \mathbb{S} \times \mathbb{X}(s) \setminus \{y^\star\}.$$

The inequality above holds, for instance, in the case $F_{\theta,\phi}(s, x) = \langle \theta, x \rangle$ with $\theta \in \mathbb{Z}_+^n$ (a vector of positive integers) and $\mathbb{X}(x) = \{0, 1\}^n$ [7, Corollary 3.6].
- Let $d(x, y) = \|x - y\|_2^2$. In this case, Assumption A.2 reads as

$$F_{\theta,\phi}(s, x) - F_{\theta,\phi}(s, y^\star) \geq \|x - y^\star\|_2^2, \quad \forall (s, x) \in \mathbb{S} \times \mathbb{X}(s).$$

Let the set $\mathbb{X}(s)$ be $\mu$-strongly convex with $\mu = 2/\|\nabla_x F_{\theta,\phi}(s, y^\star)\|_2$. Then, one can show that Assumption A.2 holds [25, Proposition 1]. Particularly, notice that the inequality holds even when $F_{\theta,\phi}(s, x) = \langle \theta, x \rangle$.
- Let $d(x, y) = \|x - y\|_2^2$. In this case, Assumption A.2 reads as

$$F_{\theta,\phi}(s, x) - F_{\theta,\phi}(s, y^\star) \geq \|x - y^\star\|_2^2, \quad \forall (s, x) \in \mathbb{S} \times \mathbb{X}(s).$$

One can show that this inequality holds when $F_{\theta,\phi}(s, x)$ is 2-strongly convex w.r.t. $x$. Thus, one can interpret Assumption A.2 as a generalization of the strong-convexity assumption used in [39, Proposition 2.5].

**Proposition A.3** (Properties of the ASL). *The ASL $\ell_\theta$ has the following properties:*

- *(**Convex surrogate**) $\ell_\theta^{\mathrm{pred}}(s, x) \leq \ell_\theta(s, x) \quad \forall (s, x) \in \mathbb{S} \times \mathbb{X}(s)$;*
- *(**Convexity**) $\ell_\theta$ is convex w.r.t. $\theta$;*
- *(**Nonnegativity**) $\ell_\theta(s, x) \geq 0, \quad \forall (s, x) \in \mathbb{S} \times \mathbb{X}(s)$;*
- *(**Consistency**) $\ell_\theta(s, x) = 0 \implies x \in \operatorname{argmin}_{y \in \mathbb{X}(s)} F_{\theta,\phi}(s, y), \quad \forall s \in \mathbb{S}$. If Assumption A.2 holds, then $\ell_\theta(s, x) = 0 \impliedby x \in \operatorname{argmin}_{y \in \mathbb{X}(s)} F_{\theta,\phi}(s, y), \quad \forall s \in \mathbb{S}$.*

*Proof.* Let $y^\star \in \operatorname{argmin}_{y \in \mathbb{X}(s)} F_{\theta,\phi}(s, y)$.

(**Convex surrogate**) By the definition of the generalized predictability loss and $y^\star$, we have $\ell_\theta^{\mathrm{pred}}(s, x) \leq d(x, y^\star)$ and

$$(29) \qquad\qquad\qquad F_{\theta,\phi}(s, x) - F_{\theta,\phi}(s, y^\star) \geq 0,$$

for all for all $x \in \mathbb{X}(s)$, for all $s \in \mathbb{S}$. Thus,

$$
\begin{aligned}
\ell_\theta^{\mathrm{pred}}(s, x) &\leq d(x, y^\star) \\
\text{(Eq. (29))} \qquad &\leq d(x, y^\star) + F_{\theta,\phi}(s, x) - F_{\theta,\phi}(s, y^\star) \\
&\leq \max_{y \in \mathbb{X}(s)} \{d(x, y) + F_{\theta,\phi}(s, x) - F_{\theta,\phi}(s, y)\} = \ell_\theta(s, x).
\end{aligned}
$$

(**Convexity**) Using the fact that $F_{\theta,\phi}(s, x) = \langle \theta, \phi(s, x) \rangle$, we conclude that $\ell_\theta(s, x)$ is the point-wise maximum of linear functions of $\theta$, which is convex w.r.t. $\theta$.

(**Nonnegative**) Since $\ell_\theta^{\mathrm{pred}}(s, x)$ is nonnegative, and we have shown that the ASL upper bounds the GPL, this implies $\ell_\theta(s, x) \geq 0$.

**(Consistency $\implies$ )**

$$\ell_\theta(s,x) = 0 \implies \max_{y \in \mathbb{X}(s)} \{F_{\theta,\phi}(s,x) - F_{\theta,\phi}(s,y) + d(x,y)\} = 0$$

$$\implies F_{\theta,\phi}(s,x) = \min_{y \in \mathbb{X}(s)} \{F_{\theta,\phi}(s,y) - d(x,y)\}$$

$$\implies F_{\theta,\phi}(s,x) \leq F_{\theta,\phi}(s,y) - d(x,y), \quad \text{for any } y \in \mathbb{X}(s)$$

$$\implies F_{\theta,\phi}(s,x) \leq F_{\theta,\phi}(s,y^\star) - d(x,y^\star)$$

(30) $$\implies F_{\theta,\phi}(s,x) - F_{\theta,\phi}(s,y^\star) \leq 0,$$

where the last inequality follows from the facts that $d(x,y^\star)$ is nonnegative. From (29) and (30) and the definition of $y^\star$, we conclude $x \in \operatorname{argmin}_{y \in \mathbb{X}(s)} F_{\theta,\phi}(s,y)$.

**(Consistency $\impliedby$ )** By Assumption A.2, we have that

$$x \in \operatorname*{argmin}_{y \in \mathbb{X}(s)} F_{\theta,\phi}(s,y) \implies F_{\theta,\phi}(s,y) - F_{\theta,\phi}(s,x) \geq d(y,x), \quad \text{for any } y \in \mathbb{X}(s)$$

$$\implies F_{\theta,\phi}(s,x) - F_{\theta,\phi}(s,y) + d(y,x) \leq 0, \quad \text{for any } y \in \mathbb{X}(s)$$

$$\implies \max_{y \in \mathbb{X}(s)} \{F_{\theta,\phi}(s,x) - F_{\theta,\phi}(s,y) + d(y,x)\} \leq 0 \implies \ell_\theta(s,x) \leq 0.$$

Since we have already shown that $\ell_\theta(s,x) \geq 0$, this implies $\ell_\theta(s,x) = 0$. $\qquad\square$

## Appendix B. Continuous problems: special cases

A key step in the derivation of the reformulation presented in Theorem 4.5 is dualizing the maximization problem in the ASL with respect to the continuous part of the decision vector. For the reformulation to be exact, strong duality needs to hold, i.e., the maximization problem needs to be concave w.r.t. the continuous variable. As mentioned in Section 4.2, using a general $\infty$-norm penalization comes at the expense of increasing the number of constraints in the reformulation by a factor of $2u$. However, for some special cases, one can dualize the distance-augmented problem without increasing the number of constraints of the final problem. Here we briefly discuss two such examples: *linear programs with totally unimodular constraint matrix* and *quadratically constrained quadratic programs*.

**LPs with totally unimodular constraint matrix.** Let us consider IO problems with linear hypothesis $\langle \theta, \phi(s,x) \rangle = \langle \theta, x \rangle$ and linear constraints $Ax \leq b, 0 \leq x \leq \mathbb{1}$. For this problem, the inner maximization problem of the ASL with $d(\hat{x}, x) = \|\hat{x} - x\|_1$ is of the form

(31)
$$\max_{x \in \mathbb{R}^n} \quad \langle \theta, \hat{x} - x \rangle + \|\hat{x} - x\|_1$$
$$\text{s.t.} \quad Ax \leq b$$
$$0 \leq x \leq \mathbb{1}.$$

Due to the norm in the objective function, this is a nonconcave optimization problem in general. However, consider the case when the constraint matrix $A$ in (31) is *totally unimodular* and $b$ is an integer vector. Totally unimodular constraint matrices appear in many classical optimization problems, for instance, shortest path problems, bipartite graph matching, and maximum flow problems [23]. In this case, it is a well-known result that every extreme point of the polyhedron $\{x : Ax \leq b, \ 0 \leq x \leq \mathbb{1}\}$ is integral [31], thus, (31) has integral (in this particular case, binary) optima. Combined with the fact that, if $\hat{x}$ and $x$ are binary vectors, we have that

$\|\hat{x} - x\|_1 = \langle \mathbb{1} - 2\hat{x}, x \rangle + \langle \mathbb{1}, \hat{x} \rangle$, the optimization problem (31) is equivalent to

$$\max_{x \in \mathbb{R}^n} \quad \langle \theta, \hat{x} - x \rangle + \langle \mathbb{1} - 2\hat{x}, x \rangle + \langle \mathbb{1}, \hat{x} \rangle$$
$$\text{s.t.} \quad Ax \leq b$$
$$0 \leq x \leq \mathbb{1},$$

that is, problem (31) is equivalent to an LP and, thus, can be dualized without increasing the number of constraints in the final IO reformulation.

**Quadratically constrained quadratic programs.** Next, consider IO problems with quadratic hypothesis $\langle \theta, \phi(s, x) \rangle = \langle x, Qx \rangle + 2\langle x, q \rangle$ and one quadratic constraint $\langle x, Ax \rangle + 2\langle x, b \rangle + c \leq 0$. For this problem, the inner maximization problem of the ASL with $d(\hat{x}, x) = \|\hat{x} - x\|_2^2$ is of the form

$$(32) \qquad \begin{aligned} \max_{x \in \mathbb{R}^n} \quad & \langle \hat{x}, Q\hat{x} \rangle + 2\langle \hat{x}, q \rangle - \langle x, Qx \rangle - 2\langle x, q \rangle + \|\hat{x} - x\|_2^2 \\ \text{s.t.} \quad & \langle x, Ax \rangle + 2\langle x, b \rangle + c \leq 0. \end{aligned}$$

Problem (32) is sometimes called a *Generalized Trust Region Problem*, and has numerous applications to, for example, robust optimization, signal processing, and compressed sensing [40, 46, 60]. Moreover, (32) is not a concave maximization problem unless $Q - I \succcurlyeq 0$. However, it is a well-known result that strong duality holds for (32) and its dual program is

$$\begin{aligned} \min_{t, \lambda} \quad & \langle \hat{x}, Q\hat{x} \rangle + 2\langle \hat{x}, q \rangle + \langle \hat{x}, \hat{x} \rangle - t \\ \text{s.t.} \quad & \lambda \geq 0 \\ & \begin{bmatrix} Q - I + \lambda A & q + \hat{x} + \lambda b \\ * & \lambda c - t \end{bmatrix} \succcurlyeq 0, \end{aligned}$$

provided Slater's constraint qualification is satisfied, i.e., there exists an $x$ with $\langle x, Ax \rangle + \langle x, b \rangle + c \leq 0$ [14, Appendix B].

## Appendix C. Proofs

### C.1. **Proof of Theorem 3.2**

Using the facts that $\|\tilde{\theta}\|_2 = \|\theta\|_2 = 1$, the range of the arccos is $[0, \pi]$, $-\cos(\gamma)$ is monotone increasing for $\gamma \in [0, \pi]$, and $-\langle \tilde{\theta}, \theta \rangle = \frac{1}{2}\|\tilde{\theta} - \theta\|_2^2 - 1$, evaluating the cost function of (7) is equivalent to

$$(33) \qquad \begin{aligned} \max_{\tilde{\theta}} \quad & \|\theta - \tilde{\theta}\|_2^2 \\ \text{s.t.} \quad & \tilde{\theta} \in \mathbb{C}, \quad \|\tilde{\theta}\|_2 = 1. \end{aligned}$$

Next, applying the change of variables $\tilde{\theta} \mapsto R\tilde{\theta}$, where $R \in \mathbb{R}^{n \times n}$ is an orthonormal matrix with its first column equal to $\theta$, (33) is equivalent to

$$(34) \qquad \begin{aligned} \max_{\tilde{\theta}} \quad & \|e_1 - \tilde{\theta}\|_2^2 \\ \text{s.t.} \quad & R\tilde{\theta} \in \mathbb{C}, \quad \|\tilde{\theta}\|_2 = 1. \end{aligned}$$

Then, defining $\mathbb{C} := \{R\tilde{\theta} = (x, y) \in \mathbb{R} \times \mathbb{R}^{n-1} : a_i x + \langle b_i, y \rangle \leq 0 \; \forall i \in [N], \; -x + \langle e_j, y \rangle \leq 0 \text{ and } -x - \langle e_j, y \rangle \leq 0 \; \forall j \in [n-1]\}$, we can rewrite (34) as

$$\begin{aligned}
\max_{x,y} \quad & (x-1)^2 + \|y\|_2^2 \\
\text{s.t.} \quad & a_i x + \langle b_i, y \rangle \le 0 && \forall i \in [N] \\
& -x + \langle e_j, y \rangle \le 0 && \forall j \in [n-1] \\
& -x - \langle e_j, y \rangle \le 0 && \forall j \in [n-1] \\
& x^2 + \|y\|_2^2 = 1.
\end{aligned}$$

(35)

Next, we change the objective function of (35) from $f(x,y) = (x-1)^2 + \|y\|_2^2$ to $g(x,y) = \|(1/x)y\|_2^2$. The resulting optimization problem is equivalent to (35) because $g$ is a strictly positive monotonic transformation of $f$, that is, for any feasible $(x_1, y_1)$ and $(x_2, y_2)$, we have that

$$\begin{aligned}
& f(x_1, y_1) < f(x_2, y_2) \\
&\iff (x_1-1)^2 + \|y_1\|_2^2 < (x_2-1)^2 + \|y_2\|_2^2 \\
&\iff -2x_1 < -2x_2 \\
&\iff \frac{1}{x_1^2} < \frac{1}{x_2^2} \\
&\iff \frac{1 - x_1^2}{x_1^2} < \frac{1 - x_2^2}{x_2^2} \\
&\iff \left\| \frac{1}{x_1} y_1 \right\|_2^2 < \left\| \frac{1}{x_2} y_2 \right\|_2^2 \\
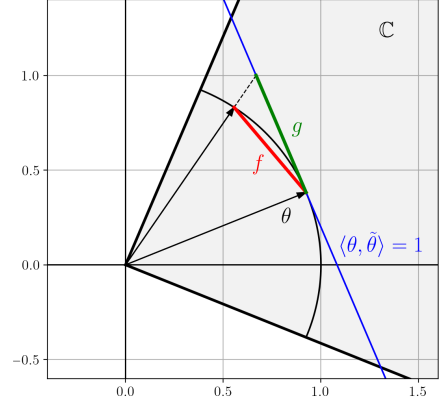&\iff g(x_1, y_1) < g(x_2, y_2),
\end{aligned}$$



Illustration of the monotonicity of the transformation from $f$ to $g$.

where we used the fact that from the inequalities $-x \pm \langle e_j, y \rangle \le 0$ and the equality $x^2 + \|y\|_2^2 = 1$, any feasible $x$ is strictly larger than 0. Thus, (35) is equivalent to

$$\begin{aligned}
\max_{x,y,z} \quad & \|z\|_2^2 \\
\text{s.t.} \quad & a_i x + \langle b_i, y \rangle \le 0 && \forall i \in [N] \\
& -x + \langle e_j, y \rangle \le 0 && \forall j \in [n-1] \\
& -x - \langle e_j, y \rangle \le 0 && \forall j \in [n-1] \\
& x^2 + \|y\|_2^2 = 1, \quad z = \frac{1}{x} y.
\end{aligned}$$

(36)

Next, substituting $y = xz$ in the inequality constraints, and using the fact that $x > 0$ to rearrange the constraints, we arrive at

$$\begin{aligned}
\max_{z} \quad & \|z\|_2^2 \\
\text{s.t.} \quad & \langle b_i, z \rangle \le -a_i \quad \forall i \in [N] \\
& -1 \le z \le 1,
\end{aligned}$$

(37)

where we dropped the equality constraints $x = 1/\sqrt{1 + \|z\|_2^2}$ and $y = \left(1/\sqrt{1 + \|z\|_2^2}\right) z$, since they are the only ones that depend on $x$ and $y$. Therefore, we have shown that evaluating the cost function of the circumcenter problem is equivalent to a quadratic maximization problem over a polytope, which is NP-hard [50].

## C.2. **Proof of Theorem 3.5**

First, the non-zero constraints can be enforced via $\|\theta\|_2 = \|\tilde{\theta}\|_2 = 1$, and similar to the proof of Theorem 3.2, one can show that

$$\underset{\|\theta\|_2=1}{\operatorname{argmax}} \underset{\substack{\tilde{\theta}\in\overline{\operatorname{int}(\mathbb{C})} \\ \|\tilde{\theta}\|_2=1}}{\min}\ a(\theta,\tilde{\theta}) = \underset{\|\theta\|_2=1}{\operatorname{argmax}} \underset{\substack{\tilde{\theta}\in\overline{\operatorname{int}(\mathbb{C})} \\ \|\tilde{\theta}\|_2=1}}{\min}\ \|\theta-\tilde{\theta}\|_2^2.$$

Since for any $\theta \in \overline{\operatorname{int}(\mathbb{C})}$, we can always choose $\tilde{\theta} = \theta$ and have $\|\theta - \tilde{\theta}\|_2^2 = 0$, the $\theta$ that maximizes this minimax problem will always be in $\mathbb{C}$. Consequently, to minimize the distance to any $\theta \in \mathbb{C}$, we have that the optimal $\tilde{\theta}$ will always be in the boundary of $\mathbb{C}$. Using these observations, one can show the following equivalence for the inner minimization problem:

$$(38) \qquad \underset{\substack{\tilde{\theta}\in\overline{\operatorname{int}(\mathbb{C})} \\ \|\tilde{\theta}\|_2=1}}{\min}\ \|\theta-\tilde{\theta}\|_2^2 = \underset{\substack{x_i\in\mathbb{X}(\hat{s}_i) \\ i\in[N]}}{\min}\left\{ \underset{\substack{\langle\tilde{\theta},\phi(\hat{s}_i,x_i)-\phi(\hat{s}_i,\hat{x}_i)\rangle=0 \\ \|\tilde{\theta}\|_2=1}}{\min}\ \|\theta-\tilde{\theta}\|_2^2 \right\},$$

which follows from the fact that the optimal $\tilde{\theta}$ being in boundary of $\mathbb{C}$ implies $\langle\tilde{\theta}, \phi(\hat{s}_i, x_i) - \phi(\hat{s}_i, \hat{x}_i)\rangle = 0$ for some $i \in [N]$ and $x_i \in \mathbb{X}(\hat{s}_i)$ (see Definition 2.2).

Next, we derive a closed-form solution to the minimization problems inside curly brackets in (38). Namely, we show that

$$(39) \qquad \underset{\substack{\langle\tilde{\theta},w\rangle=0 \\ \|\tilde{\theta}\|_2=1}}{\operatorname{argmin}}\ \|\theta-\tilde{\theta}\|_2^2 = \frac{p(\theta,w)}{\|p(\theta,w)\|_2},$$

where

$$(40) \qquad p(\theta,w) := \underset{\langle\tilde{\theta},w\rangle=0}{\operatorname{argmin}}\ \|\theta-\tilde{\theta}\|_2^2 = \theta - \frac{\langle\theta,w\rangle}{\|w\|_2^2}w$$

is the euclidean projection of $\theta$ onto the hyperplane defined by $w$. We prove (39) by contradiction. Namely, we show that if there exists a vector $\alpha$ with lower cost value than $p(\theta,w)/\|p(\theta,w)\|_2$ for (39), this contradicts the optimality of $p(\theta,w)$ for problem (40). Thus, let $\alpha \in \mathbb{R}^p$ be such that $\langle\alpha,w\rangle = 0$ and $\|\alpha\|_2 = 1$, and assume

$$\left\|\theta - \frac{p(\theta,w)}{\|p(\theta,w)\|_2}\right\|_2^2 > \|\theta-\alpha\|_2^2.$$

Consequently, we have that

$$\left\|\theta - \frac{p(\theta,w)}{\|p(\theta,w)\|_2}\right\|_2^2 > \|\theta-\alpha\|_2^2 \iff 1 + 1 - \frac{2}{\|p(\theta,w)\|_2}\langle\theta,p(\theta,w)\rangle > 1 + 1 - 2\langle\theta,\alpha\rangle$$

$$\iff -2\langle\theta,p(\theta,w)\rangle > -2\|p(\theta,w)\|_2\langle\theta,\alpha\rangle,$$

which follows by expanding the norm squared and the facts that $\|\theta\|_2 = \|\alpha\|_2 = 1$. Adding $\|\theta\|_2^2 + \|p(\theta,w)\|_2^2$ to both sides and using the identity $\|\theta-p(\theta,w)\|_2^2 = \|\theta\|_2^2 + \|p(\theta,w)\|_2^2 - 2\langle\theta,p(\theta,w)\rangle$, we get that

$$\left\|\theta - \frac{p(\theta,w)}{\|p(\theta,w)\|_2}\right\|_2^2 > \|\theta-\alpha\|_2^2 \iff \|\theta-p(\theta,w)\|_2^2 > \|\theta\|_2^2 + \|p(\theta,w)\|_2^2 - 2\|p(\theta,w)\|_2\langle\theta,\alpha\rangle.$$

Next, define $\beta := \|p(\theta, w)\|_2 \alpha$, and notice that $\langle \beta, w \rangle = 0$ and $\|\beta\|_2^2 = \|p(\theta, w)\|_2^2$. Thus, we arrive at

$$\left\| \theta - \frac{p(\theta, w)}{\|p(\theta, w)\|_2} \right\|_2^2 > \|\theta - \alpha\|_2^2 \iff \|\theta - p(\theta, w)\|_2^2 > \|\theta\|_2^2 + \|\beta\|_2^2 - 2\langle \theta, \beta \rangle = \|\theta - \beta\|_2^2.$$

Therefore, since $\|\theta - p(\theta, w)\|_2^2 > \|\theta - \beta\|_2^2$ contradicts the optimality of $p(\theta, w)$ in (40), (39) follows by contradiction.

Combining (38) and (39), we have that

$$\min_{\substack{\tilde{\theta} \in \text{int}(\mathbb{C}) \\ \|\tilde{\theta}\|_2 = 1}} \|\theta - \tilde{\theta}\|_2^2 = \min_{\substack{x_i \in \mathbb{X}(\hat{s}_i) \\ i \in [N]}} \left\| \theta - \frac{p(\theta, \phi(\hat{s}_i, x_i) - \phi(\hat{s}_i, \hat{x}_i))}{\|p(\theta, \phi(\hat{s}_i, x_i) - \phi(\hat{s}_i, \hat{x}_i))\|_2} \right\|_2^2.$$

To conclude the proof, notice that

$$(41a) \qquad \underset{w}{\operatorname{argmin}} \left\| \theta - \frac{p(\theta, w)}{\|p(\theta, w)\|_2} \right\|_2^2 = \underset{w}{\operatorname{argmin}} \; a\left( \theta, \frac{p(\theta, w)}{\|p(\theta, w)\|_2} \right)$$

$$= \underset{w}{\operatorname{argmin}} \; a\left( \theta, p(\theta, w) \right)$$

$$(41b) \qquad = \underset{w}{\operatorname{argmin}} \; \sin\left( a(\theta, p(\theta, w)) \right)$$

$$(41c) \qquad = \underset{w}{\operatorname{argmin}} \; \|\theta - p(\theta, w)\|_2^2$$

$$(41d) \qquad = \underset{w}{\operatorname{argmin}} \; \frac{|\langle \theta, w \rangle|}{\|w\|_2},$$

where (41a) follows from the definition of angle, (41b) follows from $a(\theta, p(\theta, w)) \leq \pi/2$, (41c) follows from $p(\theta, w) \perp \theta - p(\theta, w)$ and (41d) follows from Eq. (40). Putting it together, we conclude that

$$\theta^{\text{in}} \in \underset{\|\theta\|_2 = 1}{\operatorname{argmax}} \min_{\substack{\tilde{\theta} \in \text{int}(\mathbb{C}) \\ \|\tilde{\theta}\|_2 = 1}} a(\theta, \tilde{\theta}) = \underset{\|\theta\|_2 = 1}{\operatorname{argmax}} \min_{x_i \in \mathbb{X}(\hat{s}_i), i \in [N]} \frac{|\langle \theta, \phi(\hat{s}_i, x_i) - \phi(\hat{s}_i, \hat{x}_i) \rangle|}{\|\phi(\hat{s}_i, x_i) - \phi(\hat{s}_i, \hat{x}_i)\|_2}.$$

Applying an epigraph reformulation and rearranging the resulting inequality constraints ends the proof.

## C.3. Proof of Corollary 3.6

First, we show that the assumption that $\text{int}(\mathbb{C}) \neq \emptyset$ implies that the optimal $r$ of problem (10) is positive. To see this, notice that $\text{int}(\mathbb{C}) \neq \emptyset$ implies that there exists some $\theta \in \mathbb{R}^p$ such that $\langle \theta, \phi(\hat{s}_i, x_i) - \phi(\hat{s}_i, \hat{x}_i) \rangle > 0$, $\forall x_i \in \mathbb{X}(\hat{s}_i)$, $\forall i \in [N]$, which follows Definition 2.2. Next, simply notice that

$$r = \frac{\min_{x_i \in \mathbb{X}(\hat{s}_i), i \in [N]} \langle \theta, \phi(\hat{s}_i, x_i) - \phi(\hat{s}_i, \hat{x}_i) \rangle}{\max_{x_i \in \mathbb{X}(\hat{s}_i), i \in [N]} \|\phi(\hat{s}_i, x_i) - \phi(\hat{s}_i, \hat{x}_i)\|_2} > 0$$

is a feasible $r$ for problem (10), which shows that the optimal $r > 0$. Thus, problem (10) can be written as

$$\begin{aligned} \min_{\theta, r} \quad & \|\theta\|_2 / r \\ \text{s.t.} \quad & \langle \theta, \phi(\hat{s}_i, \hat{x}_i) - \phi(\hat{s}_i, x_i) \rangle + r \|\phi(\hat{s}_i, x_i) - \phi(\hat{s}_i, \hat{x}_i)\|_2 \leq 0 \qquad \forall x_i \in \mathbb{X}(\hat{s}_i), \; \forall i \in [N] \\ & \|\theta\|_2 = 1, \end{aligned}$$

where we used the facts that $\|\theta\|_2 = 1$ and the optimal $r > 0$. Next, applying the change of variables $\theta/r \to \bar{\theta}$, we get

$$\min_{\bar{\theta},r} \quad \|\bar{\theta}\|_2$$

$$\text{s.t.} \quad \langle \bar{\theta}, \phi(\hat{s}_i, \hat{x}_i) - \phi(\hat{s}_i, x_i)\rangle + \|\phi(\hat{s}_i, x_i) - \phi(\hat{s}_i, \hat{x}_i)\|_2 \leq 0 \qquad \forall x_i \in \mathbb{X}(\hat{s}_i), \ \forall i \in [N]$$

$$\|\bar{\theta}\|_2 = 1/r.$$

Finally, notice that since the optimization variable $r$ only appears in the constraint $\|\bar{\theta}\|_2 = 1/r$, and for any $\bar{\theta}$ we can set $r = 1/\|\bar{\theta}\|_2$, the optimization variable $r$ does not affect the optimal value of $\bar{\theta}$, and we can simply drop the constraint $\|\bar{\theta}\|_2 = 1/r$.

## C.4. **Proof of Theorem 4.5**

For the feasible set (17) and hypothesis function (18), the ASL with $d(\hat{x}, x) = \|\hat{y} - y\|_\infty + d_z(\hat{z}, z)$ can be written as

$$\max_{\substack{z \in \mathbb{Z}(\hat{w}) \\ h \in \mathbb{H}}} \max_y \quad \langle \theta, \phi(\hat{s}, \hat{x})\rangle - \langle y, Q_{yy} y\rangle - \langle y, Q\phi_1(\hat{w}, z)\rangle - \langle q, \phi_2(\hat{w}, z)\rangle + \langle h, \hat{y} - y\rangle + d_z(\hat{z}, z)$$

$$\text{s.t.} \quad \hat{A}y + \hat{B}z \leq \hat{c},$$

where we use the identity $\|\hat{y} - y\|_\infty = \max_{h \in \mathbb{H}} \langle h, \hat{y} - y\rangle$ with $\mathbb{H} := \{h \in \{-1, 0, 1\}^u : \|h\|_1 = 1\}$. Since we assume $Q_{yy} \succcurlyeq 0$, the inner maximization problem is convex, and by strong duality, we can write its dual as

$$\max_{\substack{z \in \mathbb{Z}(\hat{w}) \\ h \in \mathbb{H}}} \min_{\alpha, \lambda} \quad \langle \theta, \phi(\hat{s}, \hat{x})\rangle + \alpha + \langle \lambda, \hat{c} - \hat{B}z\rangle - \langle q, \phi_2(\hat{w}, z)\rangle + \langle h, \hat{y}\rangle + d_z(\hat{z}, z)$$

$$\text{s.t.} \quad \|Q\phi_1(\hat{w}, z) + h + \hat{A}^\top \lambda\|_{Q_{yy}^\dagger}^2 \leq 4\alpha$$

$$Q\phi_1(\hat{w}, z) + h + \hat{A}^\top \lambda \in \mathcal{R}(Q_{yy})$$

$$\lambda \geq 0.$$

Next, applying a Schur complement transformation (e.g., [14, Section A.5.5]) to the constraints of the optimization problem above, followed by an epigraph transformation, we have

$$\max_{\substack{z \in \mathbb{Z}(\hat{w}) \\ h \in \mathbb{H}}} \min_{\alpha, \beta, \lambda} \quad \beta$$

(42)
$$\text{s.t.} \quad \langle \theta, \phi(\hat{s}, \hat{x})\rangle + \alpha + \langle \lambda, \hat{c} - \hat{B}z\rangle - \langle q, \phi_2(\hat{w}, z)\rangle + \langle h, \hat{y}\rangle + d_z(\hat{z}, z) \leq \beta$$

$$\begin{bmatrix} Q_{yy} & Q\phi_1(\hat{w}, z) + h + \hat{A}^\top \lambda \\ * & 4\alpha \end{bmatrix} \succcurlyeq 0$$

$$\lambda \geq 0.$$

To reformulate (42) into a single minimization problem, we use the fact that

(43)
$$\left\{ \begin{aligned} \max_{x \in \{x_1, \ldots, x_M\}} \min_{t, \delta} \quad & t \\ \text{s.t.} \quad & g(\delta, x) \leq t \end{aligned} \right\} = \left\{ \begin{aligned} \min_{t, \delta_1, \ldots, \delta_M} \quad & t \\ \text{s.t.} \quad & g(\delta_j, x_j) \leq t, \quad \forall j \in [M] \end{aligned} \right\},$$

for a general function $g$, which follows from the observation that

$$\max_{x \in \{x_1, \ldots, x_M\}} \min_\delta g(\delta, x) = \max\{\min_{\delta_1} g(\delta_1, x_1), \ldots, \min_{\delta_M} g(\delta_M, x_M)\}$$

$$= \min_{\delta_1, \ldots, \delta_M} \max\{g(\delta_1, x_1), \ldots, g(\delta_M, x_M)\}.$$

Combining (42) with (43), we arrive at

$$
\begin{aligned}
\min \quad & \beta \\
\text{s.t.} \quad & \theta = (\text{vec}(Q_{yy}), \text{vec}(Q), q) \in \Theta, \quad \lambda_{jk} \geq 0, \quad \alpha_{jk}, \beta \in \mathbb{R} \\
& \langle \theta, \phi(\hat{s}, \hat{x}) \rangle + \alpha_{jk} + \langle \lambda_{jk}, \hat{c} - \hat{B} z_j \rangle - \langle q, \phi_2(\hat{w}, z_j) \rangle + \langle h_k, \hat{y} \rangle + d_z(\hat{z}, z_j) \leq \beta \\
& \begin{bmatrix} Q_{yy} & Q\phi_1(\hat{w}, z_j) + h_k + \hat{A}^\top \lambda_{jk} \\ * & 4\alpha_{jk} \end{bmatrix} \succcurlyeq 0,
\end{aligned}
\tag{44}
$$

where the constraints are $\forall (j, k) \in [M] \times [2u]$, $\mathbb{Z}(\hat{w}) := \{z_1, \ldots, z_{M_i}\}$, $M := |\mathbb{Z}(\hat{w})|$, and if $k \leq u$ ($k > u$), $h_k$ is the vector of zeros except for the $k$'th $k$'th (($k - u$)'th) element, which is equal to 1 (-1). Finally, plugging (44) into problem (14), we arrive at

$$
\begin{aligned}
\min \quad & \kappa \mathcal{R}(\theta) + \frac{1}{N} \sum_{i=1}^{N} \beta_i \\
\text{s.t.} \quad & \theta = (\text{vec}(Q_{yy}), \text{vec}(Q), q) \in \Theta, \quad \lambda_{ijk} \geq 0, \quad \alpha_{ijk}, \beta_i \in \mathbb{R} \\
& \langle \theta, \phi(\hat{s}_i, \hat{x}_i) \rangle + \alpha_{ijk} + \langle \lambda_{ijk}, \hat{c}_i - \hat{B}_i z_{ij} \rangle - \langle q, \phi_2(\hat{w}_i, z_{ij}) \rangle + \langle h_k, \hat{y}_i \rangle + d_z(\hat{z}_i, z_{ij}) \leq \beta_i \\
& \begin{bmatrix} Q_{yy} & Q\phi_1(\hat{w}_i, z_{ij}) + h_k + \hat{A}_i^\top \lambda_{ijk} \\ * & 4\alpha_{ijk} \end{bmatrix} \succcurlyeq 0,
\end{aligned}
$$

where the constraints are $\forall (i, j, k) \in [N] \times [M_i] \times [2u]$.

## C.5. Proof of Proposition 5.2

Let $\theta^\star \in \operatorname{argmin}_{\theta \in \Theta} f(\theta)$. By the definition of an $\varepsilon$-subgradient, we have

$$
\begin{aligned}
\mathbb{E}\left[f(\theta_t) - f(\theta^\star)\right] &\leq \mathbb{E}\left[\langle g_{\varepsilon_t}(\theta_t), \theta_t - \theta^\star \rangle + \varepsilon_t\right] \\
&= \mathbb{E}\left[\langle \mathbb{E}\left[\tilde{g}_{\varepsilon_t}(\theta_t) \mid \theta_t\right], \theta_t - \theta^\star \rangle + \varepsilon_t\right] \\
&= \mathbb{E}\left[\langle \tilde{g}_{\varepsilon_t}(\theta_t), \theta_t - \theta^\star \rangle + \varepsilon_t\right],
\end{aligned}
\tag{45}
$$

where the first equality follows from the definition of a stochastic approximate subgradient and the second equality follows from the law of total expectation. Next, we use the standard inequality for the analysis of mirror-descent algorithms [34, Proposition 5.1]

$$
\langle \tilde{g}_{\varepsilon_t}(\theta_t), \theta_t - \theta^\star \rangle \leq \frac{1}{\eta_t}(\mathcal{B}_\omega(\theta^\star, \theta_t) - \mathcal{B}_\omega(\theta^\star, \theta_{t+1})) + \frac{\eta_t}{2} \|\tilde{g}_{\varepsilon_t}(\theta_t)\|_*^2.
\tag{46}
$$

Summing (46) from $t = 1$ to $T$ and taking its expectation, we arrive at

$$
\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^{T} \langle \tilde{g}_{\varepsilon_t}(\theta_t), \theta_t - \theta^\star \rangle\right] &\leq \mathbb{E}\left[\sum_{t=1}^{T} \frac{1}{\eta_t}(\mathcal{B}_\omega(\theta^\star, \theta_t) - \mathcal{B}_\omega(\theta^\star, \theta_{t+1})) + \frac{1}{2} \sum_{t=1}^{T} \eta_t \|\tilde{g}_{\varepsilon_t}\|_*^2\right] \\
&\leq \frac{R^2}{\eta_1} + R^2 \sum_{t=2}^{T} \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}\right) + \frac{G^2}{2} \sum_{t=1}^{T} \eta_t \\
&= \frac{R^2}{\eta_T} + \frac{G^2}{2} \sum_{t=1}^{T} \eta_t \leq \left(\frac{R^2}{c} + cG^2\right) \sqrt{T},
\end{aligned}
\tag{47}
$$

where the second line follows from rearranging the sum and the boundedness assumptions, the third line follows from telescoping the sum, and the fourth line follows from the definition of $\eta_t$ and the

inequality $\sum_{t=1}^{T} 1/\sqrt{t} \le 2\sqrt{T}$. Finally, using Jensen's inequality, we arrive at

$$
\begin{aligned}
\mathbb{E}\left[f\left(\frac{1}{T}\sum_{t=1}^{T}\theta_t\right) - f(\theta^\star)\right] &\le \frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T}(f(\theta_t) - f(\theta^\star))\right] \\
&\le \frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T}\langle \tilde{g}_{\varepsilon_t}(\theta_t), \theta_t - \theta^\star\rangle + \sum_{t=1}^{T}\varepsilon_t\right] \\
&\le \left(\frac{R^2}{c} + cG^2\right)\frac{1}{\sqrt{T}} + \frac{1}{T}\sum_{t=1}^{T}\varepsilon_t,
\end{aligned}
$$

where the second inequality follows from (45) and the third inequality follows from (47). Next, we prove an improved convergence bound when $f = h + \mathcal{R}$, and $\mathcal{R}$ is relative $\alpha$-strongly convex. The proof is based on [36]. In this case, similar to (45), we have

$$
\mathbb{E}\left[f(\theta_t) - f(\theta^\star)\right] \le \mathbb{E}\left[\langle \tilde{g}_{\varepsilon_t}(\theta_t), \theta_t - \theta^\star\rangle + \varepsilon_t - \alpha\mathcal{B}_w(\theta^\star, \theta_t)\right].
$$

Invoking (46), multiplying both sides by $t$, and summing the resulting inequality from $t = 1$ to $T$, we get

$$
\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^{T} t(f(\theta_t) - f(\theta^\star))\right] &\le \mathbb{E}\left[\sum_{t=1}^{T}\left(\frac{t}{\eta_t}(\mathcal{B}_\omega(\theta^\star, \theta_t) - \mathcal{B}_\omega(\theta^\star, \theta_{t+1})) + \frac{t\eta_t}{2}\|\tilde{g}_{\varepsilon_t}\|_*^2 + t\varepsilon_t - \alpha t\mathcal{B}_w(\theta^\star, \theta_t)\right)\right] \\
&\le \left(\frac{1}{\eta_1} - \alpha\right)\mathcal{B}_\omega(\theta^\star, \theta_1) + \sum_{t=2}^{T}\left(\frac{t}{\eta_t} - \alpha t - \frac{t}{\eta_{t-1}}\right)\mathcal{B}_\omega(\theta^\star, \theta_t) + \frac{G^2}{2}\sum_{t=1}^{T}t\eta_t + \sum_{t=1}^{T}t\varepsilon_t.
\end{aligned}
$$

Next, by setting $\eta_t = 2/\alpha(t+1)$, it is easy to show that the sums between parenthesis become nonpositive, and we arrive at

$$
(48) \qquad \mathbb{E}\left[\sum_{t=1}^{T} t(f(\theta_t) - f(\theta^\star))\right] \le \frac{G^2}{\alpha}\sum_{t=1}^{T}\frac{t}{(t+1)} + \sum_{t=1}^{T}t\varepsilon_t \le \frac{G^2 T}{\alpha} + \sum_{t=1}^{T}t\varepsilon_t.
$$

Finally, using Jensen's inequality, we have that

$$
\begin{aligned}
\mathbb{E}\left[f\left(\frac{2}{T(T+1)}\sum_{t=1}^{T}t\theta_t\right) - f(\theta^\star)\right] &\le \frac{2}{T(T+1)}\mathbb{E}\left[\sum_{t=1}^{T}t(f(\theta_t) - f(\theta^\star))\right] \\
&\le \frac{2G^2}{\alpha(T+1)} + \frac{2}{T(T+1)}\sum_{t=1}^{T}t\varepsilon_t,
\end{aligned}
$$

where the second inequality follows from (48).

## C.6. **Proof of Lemma 5.3**

This proof is based on [9, Example 3.3.1]. Recall the definition of $\ell_\theta$:

$$
\ell_\theta(\hat{s}, \hat{x}) = \max_{x\in\mathbb{X}(\hat{s})} \left\{\langle\theta, \phi(\hat{s}, \hat{x}) - \phi(\hat{s}, x)\rangle + d(\hat{x}, x)\right\}.
$$

Then, by the definition of $x_\varepsilon$ we have

$$
\begin{aligned}
\ell_\theta(\hat{s}, \hat{x}) - \ell_\nu(\hat{s}, \hat{x}) &= \max_{x\in\mathbb{X}(\hat{s})}\left\{\langle\theta, \phi(\hat{s}, \hat{x}) - \phi(\hat{s}, x)\rangle + d(\hat{x}, x)\right\} - \max_{x\in\mathbb{X}(\hat{s})}\left\{\langle\nu, \phi(\hat{s}, \hat{x}) - \phi(\hat{s}, x)\rangle + d(\hat{x}, x)\right\} \\
&\le \langle\theta, \phi(\hat{s}, \hat{x}) - \phi(\hat{s}, x_\varepsilon)\rangle + d(\hat{x}, x_\varepsilon) + \varepsilon - \max_{x\in\mathbb{X}(\hat{s})}\left\{\langle\nu, \phi(\hat{s}, \hat{x}) - \phi(\hat{s}, x)\rangle + d(\hat{x}, x)\right\}.
\end{aligned}
$$

Next, by the optimality of the max, we arrive at

$$\ell_\theta(\hat{s}, \hat{x}) - \ell_\nu(\hat{s}, \hat{x}) \leq \langle \theta, \phi(\hat{s}, \hat{x}) - \phi(\hat{s}, x_\varepsilon) \rangle + d(\hat{x}, x_\varepsilon) + \varepsilon - \langle \nu, \phi(\hat{s}, \hat{x}) - \phi(\hat{s}, x_\varepsilon) \rangle + d(\hat{x}, x_\varepsilon)$$
$$= \langle \phi(\hat{s}, \hat{x}) - \phi(\hat{s}, x_\varepsilon), \theta - \nu \rangle + \varepsilon.$$

This shows that $\phi(\hat{s}, \hat{x}) - \phi(\hat{s}, x_\varepsilon)$ is an $\varepsilon$-subgradient of $\ell_\theta$ w.r.t. $\theta$.

## Appendix D. Further Numerical Results

### D.1. **Breast cancer Wisconsin prognostic dataset**

In this numerical experiment, we present a way to model a real-world decision problem using our IO approaches. Namely, we use the Breast Cancer Wisconsin Prognostic (BCWP) dataset from the UCI Machine Learning Repository [24]. This dataset consists of real-world data from different breast cancer patients. For each patient, there are 30 features extracted from images of cells taken from breast lumps, plus the tumor size and the number of involved lymph nodes, for a total of 32 numerical features. Moreover, each case is classified as "recurrent" if the disease has recurred after surgery, and "non-recurrent" if the disease did not recur by the time of the patient's last check-up. For recurrent patients, there is also information on the time to recur (TTR), that is, how many months it took for the disease to recur. For non-recurrent patients, we have information on the disease-free survival time (DFST), that is, how many months it is known that the patient was disease-free after the surgery. The dataset consists of 198 distinct cases, 47 of which have recurred. Thus, given the 32 features, the goal is to predict if the disease will recur or not, as well as the TTR (if recurrent) or the DFST (if non-recurrent).

We interpret this problem as an IO problem: given a signal vector $\hat{w} \in \mathbb{R}^{32}$ (numerical vector with the 32 features), an expert agent (e.g., a doctor) returns a decision $(y, z) \in \mathbb{R} \times \{0, 1\}$, where $y$ is the TTR/DFST and $z = 1$ if the expert predicts the disease to return (recurrent patient), and $z = 0$ otherwise (non-recurrent patient). In other words, given the data on some patients, the expert predicts if the disease will recur or not. If recurrent, the expert also predicts the TTR; if not recurrent, the expert predicts the DFST, which can be interpreted as a lower bound on how long the patient is expected to be disease-free. Thus, given the dataset of signal-decision data, we use IO to learn a cost function that when minimized, mimics the behavior of the expert. Since the decision of the expert comprises of continuous and discrete components (i.e., $\mathbb{X}(\hat{s}) = \{(y, z) \in \mathbb{R} \times \{0, 1\} : y \geq 0\}$), we tackle this scenario using the mixed-integer approach proposed in Section 4.2. Here we note that since the BCWP dataset comes from a real-world scenario, the choice of hypothesis function we use for the IO approach (in particular, the feature functions $\phi_1$ and $\phi_2$) is a modeling choice. We use the hypothesis function (18), with $\phi_1(w, z) = \phi_2(w, z) = (w, z, zw, 1)$. Thus, we can use the reformulation (19) to solve the problem, with $\hat{A} = -I$, $\hat{B} = \hat{c} = 0$, $\Theta = \mathbb{R}^n$, $d_z(\hat{z}_i, z_i) = |\hat{z}_i - z_i|$, and $\mathcal{R}(\theta) = \frac{1}{2}\|\theta\|_2^2$. We call this approach **ASL-yz**. Similar to the experiments in Section 4.2, we also test an approach similar to **ASL-yz**, but with $h_k = 0 \ \forall k \in [2u]$ (see discussion in the paragraph after Corollary 4.6). We call this approach **ASL-z**. Finally, as a benchmark, we test a regression+classification approach, where we treat the problem as a separate regression (prediction of the TTR/DFST) and classification (recurrent/non-recurrent patient). For the regression task, we tested scikit-learn's epsilon-support vector regression, ordinary least squares linear regression, kernel ridge regression, multi-layer perceptron regressor, regression based on k-nearest neighbors, gaussian process regression, and a decision tree regressor. For the classification task, we tested

| Approach | ASL-z | ASL-yz | regression+classification |
|---|---|---|---|
| $\|y - y_{\mathbf{true}}\|$ | 51.17 | 27.33 | 27.44 |
| $\|z - z_{\mathbf{true}}\|$ | 20% | 21% | 21.25% |

Table 4. Out-of-sample average error in months of the TTR/DFST prediction and out-of-sample average percentage error in the prognostic of recurrent vs non-recurrent patients.

scikit-learn's support vector classifier, logistic regression classifier, classifier implementing the k-nearest neighbors vote, gaussian process classification based on Laplace approximation, decision tree classifier, random forest classifier, multi-layer perceptron classifier, and an AdaBoost classifier. For this experiment, using scikit-learn's default tuning parameters, the methods with the best out-of-sample performance for the regression and classification tasks were the kernel ridge regression and support vector classifier, respectively.

To evaluate our approach, we generate 20 training and test datasets, where each training dataset is generated by randomly sampling 90% of the original dataset, while the remaining 10% is used as the test dataset, similar to a 20-fold cross-validation procedure. Table 4 shows the out-of-sample results for approaches to this problem. In particular, it shows the average error for the continuous part of the decision variable, i.e., the average error in months of the TTR/DFST prediction, and the percentage error of the discrete part of the decision variable, i.e., the average error in the prognostic of recurrent vs non-recurrent patients. From these results, we can see that the ASL-yz approach has slightly better performance than the regression+classification approach and has a much smaller regression error than the ASL-y approach. Compared to other works that used the BCWP dataset to predict the cancer recurrence or recurrence time, our IO approach differs from them in two major ways: first, our approach requires no pre-processing of the data. For instance, [62] splits the data into different classes according to the TTR/DFST and learns one predictor per class, and [53] pre-processes the dataset using a feature selection procedure. In contrast, our OI approach requires no pre-processing of the dataset, although we believe its results may be improved after an appropriate pre-processing of the dataset. Second, our IO approach can predict the TTR/DFST and recurrence/non-recurrence *simultaneously*. In machine learning terms, our IO approach does classification and regression at the same time. This differs from the approaches in the literature, which focus on either predicting the TTR/DFST or predicting recurrence/non-recurrence. For instance, the authors in [53] were able to achieve an average decision error of 13.9 months using leave-one-out testing and a feature selection procedure, and the authors in [37] were able to achieve a 16.53% prediction error for positive vs negative cases, where a case is considered positive if a recurrence occurred before 24 months, and negative otherwise.

## D.2. In-sample results

In this section, we show the in-sample results for the numerical experiments of Section 6. Namely, Figure 8 shows the in-sample counterpart of Figure 3, Figure 9 shows the in-sample counterpart of Figure 4, Figure 10 shows the in-sample counterpart of Figure 5, Figure 12 shows the in-sample counterpart of Figure 11, and Table 5 shows the in-sample counterpart of Table 4. Notice that since we always learn the IO cost vector using the training data, the difference between the true cost vector and the one learned using IO is independent of in- or out-of-sample results, thus, this plot is not shown in this section. Moreover, for the cases when the training dataset is noisy, we do not

(a) Average error between the decision generated by $\theta_{\text{true}}$ and $\theta_{\text{IO}}$.

(b) Relative difference between the cost of the decisions generated using $\theta_{\text{true}}$ and $\theta_{\text{IO}}$.
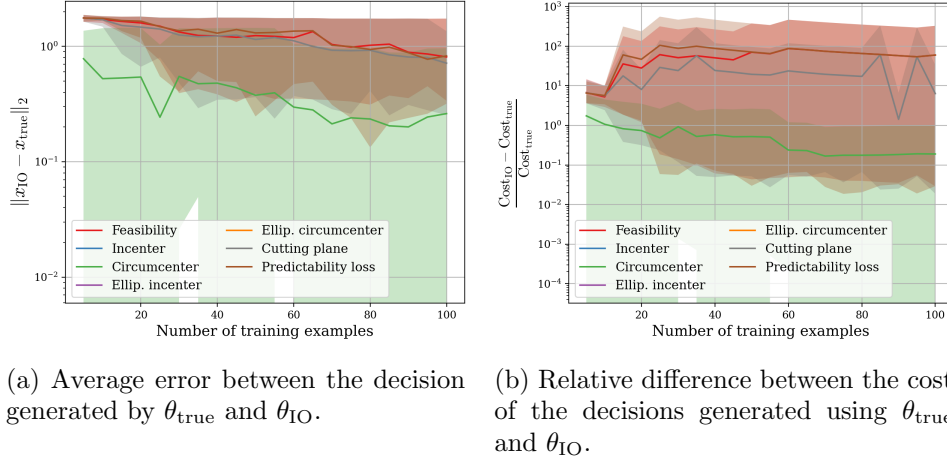
Figure 8. In-sample results for consistent data scenario. The results for the incenter approach do not appear in the plots because they are zero for all number of training examples tested.
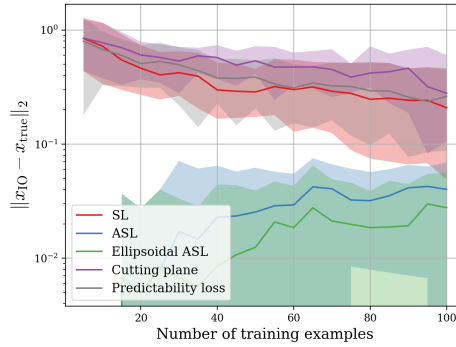


Figure 9. In-sample results for inconsistent data scenario. Average error between the decision generated by $\theta_{\text{true}}$ and $\theta_{\text{IO}}$.

show the plot that compares the difference between the cost of the expert decisions with the cost of the decisions using $\theta_{\text{IO}}$.

| Approach | ASL-z | ASL-yz | regression+classification |
|---|---|---|---|
| $|y - y_{\textbf{true}}|$ | 34.27 | 26.06 | 25.03 |
| $|z - z_{\textbf{true}}|$ | 24.16% | 24.16% | 23.48% |

Table 5. In-sample average error in months of the TTR/DFST prediction and out-of-sample average percentage error in the prognostic of recurrent vs non-recurrent patients.

D.3. **IO results for SAMD**

Figure 11 shows the performance of the algorithms tested in Section 5.3 in terms of the IO performance metrics. The discussion on the interpretation of the results of Figure 11 mirrors the one related to Figure 3 from the previous section, with the difference that now the x-axis of the figures refers to running time instead of the number of training examples. In these plots, we can
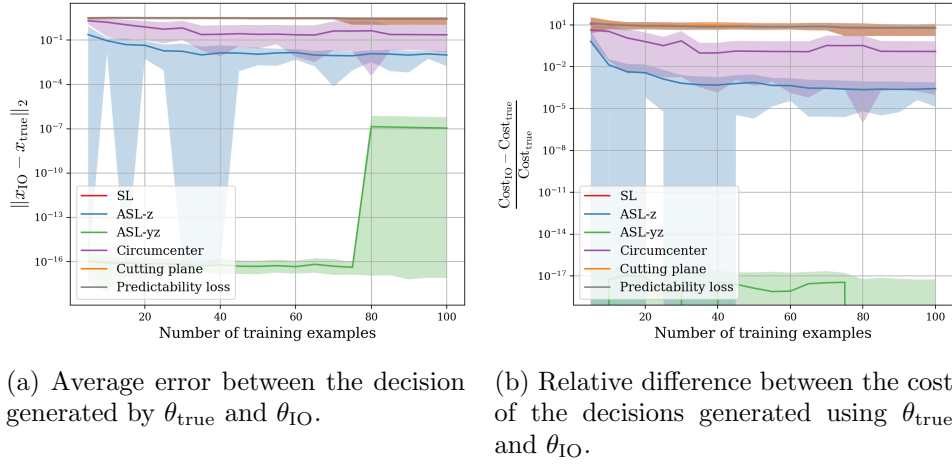
(a) Average error between the decision generated by $\theta_{\text{true}}$ and $\theta_{\text{IO}}$.

(b) Relative difference between the cost of the decisions generated using $\theta_{\text{true}}$ and $\theta_{\text{IO}}$.

Figure 10. In-sample results for the mixed-integer feasible set scenario.



(a) Difference between the true cost vector $(\theta_{\text{true}})$ and the one learned using IO $(\theta_{\text{IO}})$.

(b) Average error between the decision generated by $\theta_{\text{true}}$ and $\theta_{\text{IO}}$.

(c) Relative difference between the cost of the decisions generated using $\theta_{\text{true}}$ and $\theta_{\text{IO}}$.
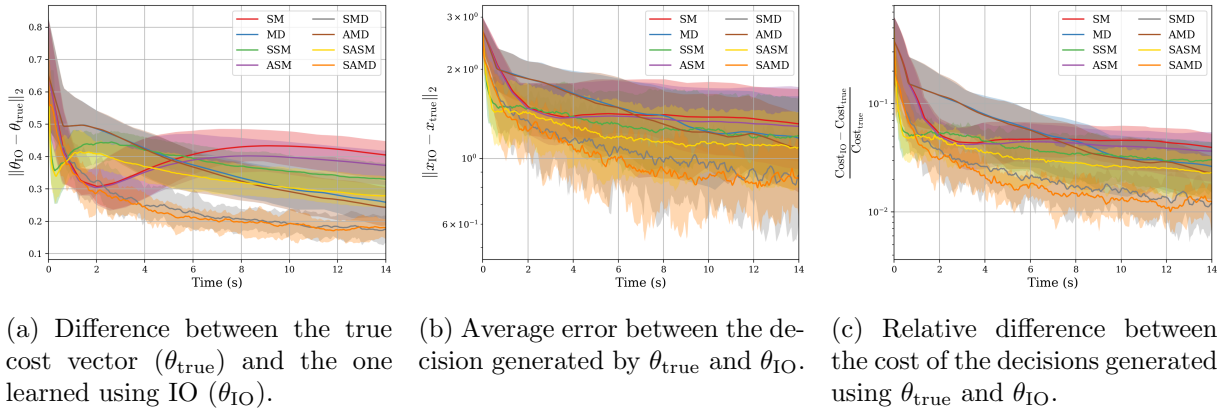
Figure 11. Out-of-sample results using first-order algorithms.



(a) Average error between the decision generated by $\theta_{\text{true}}$ and $\theta_{\text{IO}}$.

(b) Relative difference between the cost of the decisions generated using $\theta_{\text{true}}$ and $\theta_{\text{IO}}$.
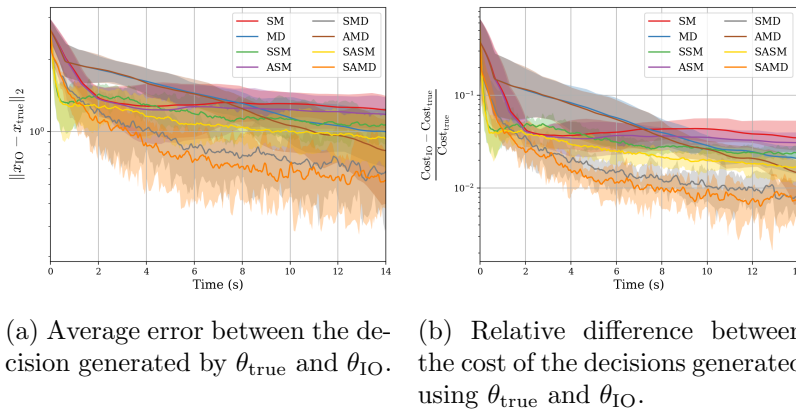
Figure 12. In-sample results using iterative first-order algorithms.

see that the improvements in convergence speed of mirror descent updates, stochastic subgradients, and approximate subgradients are also reflected in the performance of the resulting solution for the IO problem.

## References

[1] Shabbir Ahmed and Yongpei Guan. The inverse optimal value problem. *Mathematical programming*, 102:91–110, 2005.

[2] Ravindra K. Ahuja and James B. Orlin. Inverse optimization. *Operations research*, 49(5):771–783, 2001.

[3] Syed Adnan Akhtar, Arman Sharifi Kolarijani, and Peyman Mohajerin Esfahani. Learning for control: An inverse optimization approach. *IEEE Control Systems Letters*, 2021.

[4] Zeyuan Allen-Zhu and Lorenzo Orecchia. Linear coupling: An ultimate unification of gradient and mirror descent. *arXiv preprint arXiv:1407.1537*, 2014.

[5] Amazon.com, Inc. Last-mile routing challenge team performance and leaderboard, 2021. URL: https://routingchallenge.mit.edu/last-mile-routing-challenge-team-performance-and-leaderboard/.

[6] Anil Aswani, Zuo-Jun Shen, and Auyon Siddiq. Inverse optimization with noisy data. *Operations Research*, 66(3):870–892, 2018.

[7] Andreas Bärmann, Sebastian Pokutta, and Oskar Schneider. Emulating the expert: Inverse optimization through online learning. In *International Conference on Machine Learning*, pages 400–410. PMLR, 2017.

[8] Dimitri P. Bertsekas. *Nonlinear programming*. Athena Scientific, 2008.

[9] Dimitri P. Bertsekas. *Convex optimization algorithms*. Athena Scientific, 2015.

[10] 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.

[11] Dimitris Bertsimas, Vishal Gupta, and Ioannis Ch. Paschalidis. Data-driven estimation in equilibrium using inverse optimization. *Mathematical Programming*, 153(2):595–633, 2015.

[12] Omar Besbes, Yuri Fonseca, and Ilan Lobel. Contextual inverse optimization: Offline and online learning. *Operations Research*, 2023.

[13] Merve Bodur, Timothy C. Y. Chan, and Ian Yihang Zhu. Inverse mixed integer optimization: Polyhedral insights and trust region methods. *INFORMS Journal on Computing*, 34(3):1471–1488, 2022.

[14] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[15] S. Bubeck. *Convex Optimization: Algorithms and Complexity*. Foundations and Trends in Machine Learning. Now Publishers, 2015.

[16] Didier Burton and Ph L. Toint. On an instance of the inverse shortest paths problem. *Mathematical programming*, 53:45–61, 1992.

[17] Timothy C. Y. 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.

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

[19] Timothy C. Y. Chan, Taewoo Lee, and Daria Terekhov. Inverse optimization: Closed-form solutions, geometry, and goodness of fit. *Management Science*, 65(3):1115–1135, 2019.

[20] Timothy C. Y. Chan, Rafid Mahmood, and Ian Yihang Zhu. Inverse optimization: Theory and applications. *Operations Research*, 2023.

[21] Violet Xinying Chen and Fatma Kılınç-Karzan. Online convex optimization perspective for learning from dynamically revealed preferences. *arXiv preprint arXiv:2008.10460*, 2020.

[22] Joseph Y. J. Chow and Will W. Recker. Inverse optimization with endogenous arrival time constraints to calibrate the household activity pattern problem. *Transportation Research Part B: Methodological*, 46(3):463–479, 2012.

[23] M. Conforti, G. Cornuéjols, and G. Zambelli. *Integer Programming*. Graduate Texts in Mathematics. Springer International Publishing, 2014.

[24] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[25] Othman El Balghiti, Adam N. Elmachtoub, Paul Grigas, and Ambuj Tewari. Generalization bounds in the predict-then-optimize framework. *Advances in neural information processing systems*, 2019.

[26] Adam N. Elmachtoub and Paul Grigas. Smart "predict, then optimize". *Management Science*, 68(1):9–26, 2022.

[27] András Faragó, Áron Szentesi, and Balázs Szviatovszki. Inverse optimization in high-speed networks. *Discrete Applied Mathematics*, 129(1):83–98, 2003.

[28] Kimia Ghobadi, Taewoo Lee, Houra Mahmoudzadeh, and Daria Terekhov. Robust inverse optimization. *Operations Research Letters*, 46(3):339–344, 2018.

[29] Kimia Ghobadi and Houra Mahmoudzadeh. Inferring linear feasible regions using inverse optimization. *European Journal of Operational Research*, 290(3):829–843, 2021.

[30] Clemens Heuberger. Inverse combinatorial optimization: A survey on problems, methods, and results. *Journal of combinatorial optimization*, 8:329–361, 2004.

[31] Alan J. Hoffman and Joseph B. Kruskal. Integral boundary points of convex polyhedra. *50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-Art*, pages 49–76, 2010.

[32] Garud Iyengar and Wanmo Kang. Inverse conic programming with applications. *Operations Research Letters*, 33(3):319–330, 2005.

[33] Thorsten Joachims, Thomas Finley, and Chun-Nam John Yu. Cutting-plane training of structural SVMs. *Machine learning*, 77(1):27–59, 2009.

[34] Anatoli Juditsky and Arkadi Nemirovski. First order methods for nonsmooth convex large-scale optimization, i: general purpose methods. *Optimization for Machine Learning*, 30(9):121–148, 2011.

[35] Arezou Keshavarz, Yang Wang, and Stephen Boyd. Imputing a convex objective function. In *2011 IEEE International Symposium on Intelligent Control*, pages 613–619, 2011.

[36] Simon Lacoste-Julien, Mark Schmidt, and Francis Bach. A simpler approach to obtaining an o (1/t) convergence rate for the projected stochastic subgradient method. *arXiv preprint arXiv:1212.2002*, 2012.

[37] Yuh-Jye Lee and Olvi L. Mangasarian. Ssvm: A smooth support vector machine for classification. *Computational optimization and Applications*, 20(1):5–22, 2001.

[38] Haihao Lu, Robert M. Freund, and Yurii Nesterov. Relatively smooth convex optimization by first-order methods, and applications. *SIAM Journal on Optimization*, 28(1):333–354, 2018.

[39] 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.

[40] Jorge J. Moré. Generalizations of the trust region problem. *Optimization methods and Software*, 2(3-4):189–209, 1993.

[41] Arkadi Nemirovski. Lecture notes: Interior point polynomial methods in convex programming, 1996. URL: https://www2.isye.gatech.edu/~nemirovs/Lect_IPM.pdf.

[42] Arkadi Nemirovski. Prox-method with rate of convergence O(1/t) for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM Journal on Optimization*, 15(1):229–251, 2004.

[43] Yurii Nesterov. Dual extrapolation and its applications to solving variational inequalities and related problems. *Mathematical Programming*, 109(2-3):319–344, 2007.

[44] Sebastian Nowozin, Christoph H. Lampert, et al. Structured learning and prediction in computer vision. *Foundations and Trends® in Computer Graphics and Vision*, 6(3–4):185–365, 2011.

[45] Francesco Orabona. A modern introduction to online learning. *arXiv preprint arXiv:1912.13213*, 2019.

[46] Ting Kei Pong and Henry Wolkowicz. The generalized trust region subproblem. *Computational optimization and applications*, 58(2):273–322, 2014.

[47] Nathan D. Ratliff, J. Andrew Bagnell, and Martin A. Zinkevich. Maximum margin planning. In *Proceedings of the 23rd international conference on Machine learning*, pages 729–736, 2006.

[48] Nathan D. Ratliff, J. Andrew Bagnell, and Martin A. Zinkevich. (approximate) subgradient methods for structured prediction. In *Artificial Intelligence and Statistics*, pages 380–387. PMLR, 2007.

[49] Javier Saez-Gallego and Juan M. Morales. Short-term forecasting of price-responsive loads using inverse optimization. *IEEE Transactions on Smart Grid*, 9(5):4805–4814, 2017.

[50] Sartaj Sahni. Computationally related problems. *SIAM Journal on computing*, 3(4):262–279, 1974.

[51] Andrew J. Schaefer. Inverse integer programming. *Optimization Letters*, 3:483–489, 2009.

[52] Naum Zuselevich Shor. *Minimization methods for non-differentiable functions*. Springer Series in Computational Mathematics. Springer, 1985.

[53] W. Nick Street, Olvi L. Mangasarian, and William H. Wolberg. An inductive learning approach to prognostic prediction. In *Machine Learning Proceedings 1995*, pages 522–530. Elsevier, 1995.

[54] Ben Taskar, Vassil Chatalbashev, Daphne Koller, and Carlos Guestrin. Learning structured prediction models: A large margin approach. In *Proceedings of the 22nd international conference on Machine learning*, pages 896–903, 2005.

[55] Ben Taskar, Simon Lacoste-Julien, Michael I. Jordan, Kristin P. Bennett, and Emilio Parrado-Hernández. Structured prediction, dual extragradient and bregman projections. *Journal of Machine Learning Research*, 7(7), 2006.

[56] Bahar Taşkesen, Soroosh Shafieezadeh-Abadeh, and Daniel Kuhn. Semi-discrete optimal transport: Hardness, regularization and numerical solution. *Mathematical Programming*, pages 1–74, 2022.

[57] Choon Hui Teo, S. V. N. Vishwanathan, Alex Smola, and Quoc V. Le. Bundle methods for regularized risk minimization. *Journal of Machine Learning Research*, 11(1), 2010.

[58] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[59] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, Yasemin Altun, and Yoram Singer. Large margin methods for structured and interdependent output variables. *Journal of machine learning research*, 6(9), 2005.

[60] Alex L. Wang and Fatma Kılınç-Karzan. The generalized trust region subproblem: solution complexity and convex hull results. *Mathematical Programming*, 191(2):445–486, 2022.

[61] Lizhi Wang. Cutting plane algorithms for the inverse mixed integer linear programming problem. *Operations research letters*, 37(2):114–116, 2009.

[62] Elias Zafiropoulos, Ilias Maglogiannis, and Ioannis Anagnostopoulos. A support vector machine approach to breast cancer diagnosis and prognosis. In *IFIP international conference on artificial intelligence applications and innovations*, pages 500–507. Springer, 2006.

[63] Pedro Zattoni Scroccaro. InvOpt: An open-source Python package to solve Inverse Optimization problems. https://github.com/pedroszattoni/invopt, 2023.

[64] Pedro Zattoni Scroccaro, Piet van Beek, Peyman Mohajerin Esfahani, and Bilge Atasoy. Inverse optimization for routing problems. *Transportation Science*, 2024.