Online Convex Optimization Perspective for Learning from Dynamically Revealed Preferences

Violet (Xinying) Chen[1] and Fatma Kılınç-Karzan[1]

[1]Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA 15232

October 5, 2021

Abstract

We study the problem of online learning (OL) from revealed preferences: a learner wishes to learn a non-strategic agent's private utility function through observing the agent's utility-maximizing actions in a changing environment. We adopt an online inverse optimization setup, where the learner observes a stream of agent's actions in an online fashion and the learning performance is measured by regret associated with a loss function. We first characterize a special but broad class of agent's utility functions, then utilize this structure in designing a new convex loss function. We establish that the regret with respect to our new loss function also bounds the regret with respect to three classical loss functions commonly used in the inverse optimization literature. This allows us to design a flexible OL framework that enables a unified treatment of loss functions and supports a variety of online convex optimization algorithms. We demonstrate with theoretical and empirical evidence that our framework based on the new loss function (in particular online Mirror Descent) has significant advantages in terms of regret performance and solution time over other OL algorithms from the literature and bypasses the previous technical assumptions as well.

1 Introduction

Preferences of an agent implicitly dictates his/her actions, and influence for example what a company should offer as its products or how a company should personalize recommendations to an individual customer (agent). This creates incentives for the company/central decision maker to learn the preferences of their agents. Nevertheless, in reality, the true preferences of the agents are often private to the individual agents and are only implicitly revealed in the form of their behaviors/actions to the central decision maker. Such typical interactions for example include a streaming platform suggesting a number of videos to a user and tracking whether the user watches or likes them. As evident from such scenarios, inferring the agents' preference information through agent interactions and observations of their behaviors is a critical task for the decision makers in such settings.

A common assumption adopted to formalize the problem of learning from revealed preferences is that rational agents are *utility* maximizers, that is, they choose actions to maximize their utility functions subject to a set of restrictions. The central decision maker interacting with the agents is the *learner*. An important learner-centric goal is to design schemes for the learner to extract useful information on the agents' utility functions. This learning point of view of revealed preferences has been explored in a broad range of literature from economics (e.g., Beigman and Vohra (2006), Varian (2006)), machine learning (e.g., Balcan et al. (2014), Dong and Zeng (2020), Dong et al. (2018b)) and operations research (e.g., Ahmadi et al. (2020), Bärmann et al. (2017), Mohajerin Esfahani et al. (2018)). Based on the type of learner-agent interactions and information feedback, such preference learning schemes vary in information requirement, preference elicitation objective and learning complexity.

In this paper, we focus on a specific setup where the learner seeks to learn the utility function of a non-strategic agent while receiving information about the agent's actions in an online fashion. This setup fits naturally in applications where the agents benefit from effective learning of their true preferences. For example this

is the case when a streaming platform interacts with its users to learn their preferences. In this example, in a typical interaction, the platform recommends videos to a user and the user takes actions based on the recommendations. User actions, e.g., clicks, movie streaming, etc., are fully observable to the platform and they reflect the user's true preferences.

1.1 Related Literature

Varian (2006) is one of the earliest and most celebrated work for learning from revealed preferences in the economics literature. They study constructing utility functions of the agent to explain a sequence of her/his observed actions. Nevertheless, this approach has a main shortcoming—a utility function capable of explaining past actions not necessarily also guarantees accurate predictions of the future actions. Consequently, Beigman and Vohra (2006) have initiated a new line of research to learn utility functions capable of predicting future actions with statistical performance guarantees. Beigman and Vohra (2006) examine a statistical setup where the learning algorithm takes as input a batch of observations and is evaluated by its sample complexity guarantees. Zadimoghaddam and Roth (2012) focus on the setting where the agent has a linear or linearly separable concave utility function, and propose learning algorithms with polynomially bounded sample complexity. Balcan et al. (2014) identify a connection between the problem of learning a utility function and the structured prediction problem of D-dimensional linear classes. Through this connection, Balcan et al. (2014) suggest an algorithm for learning utility functions that is superior (in terms of sample complexity) than the method from Zadimoghaddam and Roth (2012) in the case of linear utility functions and is also applicable for learning separable piecewise-linear concave functions and CES functions with explicit sample complexity bounds.

As an alternative to this statistical view, Balcan et al. (2014) study a query-based learning model, where the learner aims to recover the exact utility function by querying an oracle for the agent's optimal actions. The query-based models consider an online feedback mechanism where the learner receives one observation of the agent's action at a time. When the learner has the power to choose which observation to receive from the query oracle, Balcan et al. (2014) give exact learning algorithms for several classes of utility functions. There is a recent research stream on *learning to optimize* the learner's objective function based on information from revealed preferences of the agents. In this stream it is often assumed that the learner has similar power on the selection of the observations. For example, Amin et al. (2015) and Ji et al. (2018) propose algorithms for finding the profit-maximizing prices for a seller, who has price controlling power and learns buyer preferences by observing the buying behavior at different price levels. Roth et al. (2016) and Dong et al. (2018b) consider the learning task as Stackelberg games, where the leader player is the learner and a follower player is a *strategic* agent with incentive to manipulate actions and hide information.

We note two restrictions with the problem setup in these fore-mentioned papers. First, the assumption that the learner can choose observations is not always achievable in practice. A more realistic setup is accommodated by the data-driven inverse optimization view where the learner does not control the sequence of observations. Second, when the learner is optimizing an objective function that does not explicitly measure how well s/he is learning about the agent, the approaches that are effective for choosing the learner's objective-optimizing action provide no guarantees on the quality of the learned agent information.

Inverse optimization generalizes the query-based view and offers a natural abstraction of learning from revealed preferences. This approach is typically used in settings with non-strategic agents, in which the agents have no incentive to hide information from the learner, and thus an agent's decisions reveal her/his true preferences. In this setting, the learner's goal is to recover unknown parameters of an agent's utility function from the observations of her/his true optimal solution. Chan et al. (2021) provides a comprehensive review of inverse optimization. We next summarize key developments in the literature, with an emphasis on two topics that are more relevant to this paper: data-driven inverse optimization and online inverse learning.

Early studies on inverse optimization examine the setting where the agent's optimization problem is fixed, see e.g., Ahuja and Orlin (2001), Heuberger (2004), Iyengar and Kang (2005), Schaefer (2009). Unfortunately, this classical setup is limited in its practical applicability as it ignores uncertainty in the environment. A new thread of research on data-driven inverse optimization studies a flexible setup, where the learner observes the agent's optimal or sub-optimal decisions corresponding to varying external data signals. In the noiseless

case, that is, when observations of optimal solutions/agent actions are available, Keshavarz et al. (2011) show that data-driven inverse optimization of convex programs is polynomial time solvable. In the case of noisy observations, Aswani et al. (2018) proves that such problems are NP-hard in general.

Data-driven inverse optimization is further categorized based on whether observations are given as a batch upfront or in an online manner. In the batch setup, Keshavarz et al. (2011) study the inverse optimization of identifying the unknown affine weights in a convex objective function that is an affine combination of pre-selected basis convex functions. Recent work of Aswani et al. (2018) and Mohajerin Esfahani et al. (2018) in the batch setup investigates the inverse optimization of general convex programs without the basis function structure. Aswani et al. (2018) adopt the *prediction loss* ℓ^{pre}, which measures the difference between the observed agent action and the predicted agent action through squared norm distance, as the inverse optimization objective. They formulate the inverse problem into a bilevel program using Lagrangian duality, and present two heuristic algorithms with approximation guarantees for solving the bilevel formulation. Mohajerin Esfahani et al. (2018) use *suboptimality loss* ℓ^{sub}, which is defined as the difference between objective values at the observed agent action and the predicted action, as their loss function and provide a distributionally robust formulation of the inverse problem. Batch setup requires that the learner receives observations of the agent's actions all at once. However, obtaining a large batch of observations all at once as well as learning from such a batch often presents operational and computational challenges. In practice, such strong batch feedback is rare as the learner often interacts with the agent repetitively in a dynamic environment.

A recent stream of research Bärmann et al. (2017), Dong et al. (2018a) adopts a dynamic information acquisition setup and studies the online data-driven inverse optimization where the learner observes a stream of the agent's actions one by one in an online fashion. Both Bärmann et al. (2017) and Dong et al. (2018a) suggest OL algorithms and measure their performance via the *regret*, i.e., the difference between the losses incurred from online estimates of the unknown parameters in the agent's utility function and the offline optimal estimate. Bärmann et al. (2017) consider the problem of learning the linear utility function of an agent given the noiseless online observations of the agent's actions in a dynamic environment. They propose two specialized OL algorithms with first-order oracles that both achieve a bound of $O(\sqrt{T})$ on the sum of the suboptimality loss ℓ^{sub} and the estimate loss ℓ^{est} after T periods but lacks regret guarantees with respect to the prediction loss ℓ^{pre}. Dong et al. (2018a) consider the setup, where the learner wishes to learn an unknown linear component of an agent's quadratic objective function from noisy observations. By utilizing the implicit OL framework of Kulis and Bartlett (2010) equipped with a Mixed Integer Second Order Cone Program (MISOCP)-based solution oracle, they provide a regret bound of $O(\sqrt{T})$ with respect to the prediction loss ℓ^{pre} after T periods whenever ℓ^{pre} is convex. Dong et al. (2018a) present a number of rather technical assumptions that guarantee the convexity of ℓ^{pre}, however these assumptions are not only difficult to verify but also quite restrictive. In fact, in Dong et al. (2018a), these were shown to hold only for a specific class of convex quadratic problem.

1.2 Contributions and Outline

We propose a novel modeling framework for learning from dynamically revealed preferences via online data-driven inverse optimization. In our setup, the learner monitors a sequence of data signals and observes the respective rational decisions of a non-strategic agent without noise over a finite time horizon of T time steps. The learner operates and receives information in an online fashion, and updates an estimate θ_t of θ_{true} using newly available information at each time step.

Our framework is enabled by the identification of ℓ^{sim}, a loss function that is both simple in structure and has practical connections with conventional loss functions from the inverse optimization literature. Online inverse optimization based on ℓ^{sim} is a online convex optimization (OCO) problem, which enjoys the flexibility to be handled by any deterministic OCO algorithm. A detailed list of our contributions along with an outline of the paper is as follows.

- In Section 2, we present a formal description of our problem setting. Section 2.1 introduces the agent's problem and discusses a rather broad decomposable structure assumption on the agent's utility functions that is capable of representing all of the utility functions studied in the data-driven inverse optimization

literature as well as other key utility functions. Section 2.2 describes the learner's inverse optimization problem that minimizes a given *loss function* $\ell(\cdot)$ to obtain an accurate estimate θ of the hidden parameter θ_{true}. We then state the online inverse optimization framework in Section 2.3: we describe the sequence of events and define regret as the performance measure.

- In Section 3, we utilize our structural assumption on the agent's utility function to design a new convex loss function, namely *simple loss* ℓ^{sim}.
- We establish in Section 4 that in the noiseless setting, a bounded regret with respect to ℓ^{sim} also guarantees a bounded regret with respect to all the other loss functions; see Proposition 1 and Corollary 1. We also briefly discuss the noisy setting.
- Convexity and simplicity of ℓ^{sim} enables us to use an online convex optimization (OCO) framework (see Section 5) that offers the flexibility to use different OL algorithms, such as, online Mirror Descent (MD) utilizing a first-order oracle (Section 5.1) and implicit OL based on a solution oracle (Section 5.2). In the noiseless setup, our framework equipped with online MD covers *all* of the problem classes studied in the online data-driven inverse optimization literature, and matches the corresponding state-of-the-art regret bounds with respect to *all* of the loss functions in a *unified* manner. In particular, our results immediately generalize the customized algorithms from Bärmann et al. (2017) and completely bypass the requirement to verify the rather technical assumptions of Dong et al. (2018a) and the need to use their expensive MISOCP-based solution oracle; see Section 5.3 for a detailed comparison discussion.
- Our numerical study in Section 6 highlights that when compared to the ℓ^{pre}-based implicit OL with an MISOCP solution oracle approach of Dong et al. (2018a), ℓ^{sim}-based OL algorithms equipped with a first-order oracle or a solution oracle, particularly online MD, demonstrate significant advantages in terms of both the learning performance (i.e., regret bounds) and the computation time. This is directly in line with our theoretical results. Moreover, these results seem to be fairly robust with respect to changes in the structure of the agent's domain as well as the noise in observations.

All proofs are given in Appendix D, derivation details on the solution oracles are presented in Appendix E, and Appendices F provide further discussion about the noisy setup.

Notation. We let $\mathbb{R}^n_+$ be the set of nonnegative n-dimensional vectors. For a given vector v, we use v_i to denote its i-th element. We let $[n] \coloneqq \{1, \dots, n\}$, and we use $\{a_i\}_{i\in[n]}$ to represent a collection of entries, such as vectors, functions, etc., indexed with $i \in [n]$. For a differentiable function f, we use $\nabla f(x)$ to denote the gradient of f at x. For a nondifferentiable function f, we use $\partial f(x)$ to denote the subdifferential of f at x.

2 Problem Setting

In our setting, the learner monitors a sequence of external signals $\{u_t\}_{t\in[T]} \subseteq \mathbb{R}^k$ and observations $\{y_t\}_{t\in[T]} \subseteq \mathbb{R}^n$ of the agent's respective optimal decisions over a finite time horizon of T time steps.

2.1 Forward Problem

For a fixed exogenous signal u, the agent's optimal decision $x(\theta_{true}; u)$ is given by the *forward problem*:

$$x(\theta_{true}; u) \in \arg\min_x \{f(x; \theta_{true}, u) : \ g(x; u) \leq 0, \ x \in \mathcal{X}\}, \tag{1}$$

where $\mathcal{X} \subseteq \mathbb{R}^n$ is the static domain of the agent's problem and $\theta_{true} \in \mathbb{R}^p$ is a parameter known only by the agent, f represents the negative of the agent's utility function capturing her/his preferences, and g is a (set of) constraint(s) defining agent's feasible actions in $\mathcal{X}$. We study a special class of objective functions f in the forward problem (1). Note that the agent's feasible domain does not depend on her/his preference parameter, θ_{true} or θ. We argue that this is a reasonable assumption in practice, for instance, on a streaming platform, an agent's preference does not affect what contents and decisions are available.

Assumption 1. The function f has a decomposable structure of the form $f(x; \theta, u) = f_1(x; u) + f_2(\theta; u) + \langle \theta, c(x) \rangle$, where $c(x) = (c_1(x), \dots, c_p(x))$. □

While Assumption 1 may appear to be restrictive, it still allows us to capture all problem classes studied in the literature as well as two other important classes of utility functions that have not been addressed in the online preference learning setting: CES (constant elasticity of substitute) function, and Cobb-Douglas function, which is a limit case of the CES function. See Appendix A for the transformations of CES and Cobb-Douglas functions to satisfy Assumption 1. Unfortunately, the other well-studied limit case known as the Leontief function does not fit into the same framework. In Section 3, we will show how Assumption 1 is advantageous in establishing desirable convexity of the suboptimality loss function used in literature, and designing a new convex loss function.

Remark 1. Assumption 1 allows for the possibility of the function c to obscure information. In most examples of interest in the online inverse optimization or preference learning, the function c will provide direct information on x, such as $c_i(x_i) = x_i$, or $c_i(x_i) = x_i^2$, etc. That said, one can purposefully select this function c to obscure information on x, e.g., $c_i(x_i) = 0$ for almost all values of x_i. In such cases, our framework as well as any other meaningful approach may fail to provide interesting guarantees (i.e., sublinear regret bounds) in an online setup. We further discuss this in Appendix B. □

2.2 Inverse Problem

Given a signal u, under *perfect information* the learner observes the agent's optimal solution without noise, i.e., $y = x(\theta_{true}; u)$; under *imperfect information*, $y = x(\theta_{true}; u) + \epsilon$, where $\epsilon \in \mathbb{R}^n$ is the noise that the learner suffers from when observing agent's action. Consistent with the literature, we assume that the learner has access to an *agent response oracle*, with which the learner can generate the *predicted action* $x(\theta; u)$ at a given signal u and estimate θ by solving the following model obtained from (1) where θ_{true} is replaced with θ

$$x(\theta; u) \in \arg\min_x \{f(x; \theta, u) : \ g(x; u) \leq 0, \ x \in \mathcal{X}\}. \tag{2}$$

Based on external signals, past observations of the agent's respective optimal decisions, and the knowledge of a convex set Θ containing θ_{true}, the learner wishes to predict θ values that will mimic closely the agent's preference-driven action $x(\theta_{true}; u)$. The performance of learner's estimates are measured via *loss function* ℓ: given signal u, the learner incurs $\ell(\theta, x(\theta; u); y, u)$ as the loss for the estimate θ, where y is the learner's observation of $x(\theta_{true}; u)$ and $x(\theta; u)$ is the learner's prediction of the agent's optimal decision as defined in (2). Specifically, $\ell : \mathbb{R}^p \times \mathbb{R}^n \mapsto \mathbb{R}$ takes $(\theta, x(\theta; u)) \in \mathbb{R}^p \times \mathbb{R}^n$ as independent variables and (y, u) as given parameters. We refer to the learner's loss minimization problem as the *inverse problem*. More formally, given the revealed parameters u and y, this *inverse problem* is a bilevel program of the form

$$\min_{\theta, x(\theta;u)} \{\ell(\theta, x(\theta; u); y, u) : \ x(\theta; u) \in \operatorname{argmin}_{x \in \mathcal{X}} \{f(x; \theta, u) : g(x; u) \leq 0\}, \ \theta \in \Theta\}. \tag{3}$$

2.3 Online Inverse Optimization

We consider the dynamic information acquisition setup where the time varying data signals $\{u_t\}_{t\in[T]}$ and the corresponding agent action observations $\{y_t\}_{t\in[T]}$ become available in an online fashion. To formalize, in the online inverse optimization over a finite time horizon T, at each time step $t \in [T]$, the learner generates an estimate $\theta_t \in \Theta$ for the true parameter θ_{true} using the current signal u_t, the past information $\{(y_{t'}, u_{t'})\}_{t'\in[t-1]}$, and the feedback collected on the loss functions $\ell_{t'}(\theta) \coloneqq \ell(\theta, x(\theta; u_{t'}); y_{t'}, u_{t'})$, $t' \in [t-1]$, i.e., from the previous $t-1$ time steps, and then the learner observes y_t. Typical OL algorithms rely on the feedback on the current loss function $\ell_t(\theta)$ such as the first-order information, i.e., the gradient $\nabla\ell_t(\theta_t)$ or a subgradient $\partial\ell_t(\theta_t)$ (indeed we will operate with this type of feedback too). In the case of online inverse optimization, once the observation y_t is revealed, we must demonstrate that such feedback for $\ell_t(\theta)$ is possible based on the information available to the learner at the current iteration t, i.e., $\{(y_{t'}, u_{t'}, \theta_{t'})\}_{t'\in[t]}$, and we will illustrate that for our choice of loss function $\ell_t(\theta)$ this is indeed possible.

The goal of the online learner is to minimize the cumulative loss $\sum_{t\in[T]} \ell_t(\theta_t)$. The performance of an OL algorithm is measured via *regret*, that is, the difference between the cumulative loss incurred from the online decisions $\{\theta_t\}_{t\in[T]}$ and the best fixed decision in hindsight:

$$R_T\left(\{\ell_t\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}\right) \coloneqq \textstyle\sum_{t\in[T]} \ell_t(\theta_t) - \min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t(\theta). \tag{4}$$

When an OL algorithm attains a vanishing average regret overtime, it guarantees a vanishing optimality gap with respect to the offline counterpart where all time-varying information are available in advance.

Remark 2. In standard OL, specially OCO where $\{\ell_t(\theta)\}_{t\in[T]}$ are convex for all t, loss functions follow an adversarial view, i.e., an adversary who tries to hide as much information as possible from the learner generates them and then they are revealed to the learner. In contrast to this standard adversarial view in OCO, as we will later describe, in our online inverse learning framework, we are concurrently designing the loss functions $\{\ell_t(\theta)\}_{t\in[T]}$ and applying deterministic OCO algorithms to these loss functions. This may bring the question of whether the usual guarantees of OCO regret minimizing algorithms will remain valid in our setup or not. To this end, we highlight that the regret guarantees provided by the standard OCO algorithms hold for *arbitrary* families of convex loss functions $\{\ell_t(\theta)\}_{t\in[T]}$. This flexibility in handling loss functions, often adversarial ones, makes OCO a useful and versatile tool in many applications (see Hazan (2019)). In fact, the application of OCO to *non-arbitrary* functions as a tool has been employed previously in the context of designing OCO-based frameworks for solving robust convex optimization in Ben-Tal et al. (2015), Ho-Nguyen and Kılınç-Karzan (2018), Ho-Nguyen and Kılınç-Karzan (2019). For our particular application, in Section 3, we design linear functions $\{\ell_t^{sim}(\theta)\}_{t\in[T]}$ to which we apply OCO algorithms. While using non-arbitrary classes of loss functions does not invalidate any guarantees from deterministic OCO methods such as online MD, more caution is needed for OCO algorithms which involve randomness and provide guarantees on *expected regret* such as stochastic gradient descent. This is due to the fact that the design of the loss functions in specific applications may create undesirable dependence among random variables and invalidate certain steps used in the analysis of stochastic OCO algorithms. Hence, here we will focus on deterministic OCO algorithms. □

3 Loss Functions

Loss function $\ell(\theta)$ plays a key role in the formulation of the inverse problem (3). Since the learner's goal is to mimic the agent's true action with the estimated preference θ, the appropriate loss functions should reflect how close the prediction $x(\theta;u)$ is to $x(\theta_{true};u)$ at a given signal u. The following are common loss functions used in inverse optimization context (recall that f is the agent's forward objective in (1) and $x(\theta;u_t)$ is the optimal solution to (2) for given θ, u_t):

- Prediction loss: $\ell^{pre}(\theta, x(\theta;u_t); y_t, u_t) \coloneqq \|y_t - x(\theta;u_t)\|^2$,
- Suboptimality loss: $\ell^{sub}(\theta, x(\theta;u_t); y_t, u_t) \coloneqq f(y_t;\theta,u_t) - f(x(\theta;u_t);\theta,u_t)$, and
- Estimate loss: $\ell^{est}(\theta, x(\theta;u_t); y_t, u_t) \coloneqq f(x(\theta;u_t);\theta_{true},u_t) - f(y_t;\theta_{true},u_t)$.

These functions use the observation y_t as a proxy of the true action $x(\theta_{true};u_t)$. Under perfect information with $y_t = x(\theta_{true};u_t)$, ℓ^{pre} directly compares the distance between the true action and the predicted action. ℓ^{sub} and ℓ^{est} utilize the agent's objective function: ℓ^{sub} measures how much y_t would affect the agent's optimal objective value at estimate θ, ℓ^{est} measures how much $x(\theta;u_t)$ would change the observed agent's objective value. Under imperfect information, due to the noises present in y_t, the loss values can be split into two components, one reflects the difference between $x(\theta;u_t)$ and $x(\theta_{true};u_t)$, and the other one is dependent on the noise shifting $x(\theta_{true};u_t)$ to y_t.

Within data-driven inverse optimization in a batch setup, ℓ^{pre} is used in Aswani et al. (2018) and ℓ^{sub} is used in Mohajerin Esfahani et al. (2018) (see Section 1.1). In the online inverse optimization setup, ℓ^{sub} and ℓ^{est} are studied by Bärmann et al. (2017) under the assumption that f is linear in x, and ℓ^{pre} by Dong et al. (2018a) when the forward problem is a quadratic program with a special structure. The OL algorithms from these latter two papers are customized for the chosen loss functions and forward problem structure, indicating the lack of a unified general framework.

We introduce the following shorthand notation.

$$\ell_t^{pre}(\theta) \coloneqq \ell^{pre}(\theta, x(\theta;u_t); y_t, u_t),\ \ell_t^{sub}(\theta) \coloneqq \ell^{sub}(\theta, x(\theta;u_t); y_t, u_t),\ \ell_t^{est}(\theta) \coloneqq \ell^{est}(\theta, x(\theta;u_t); y_t, u_t).$$

We first establish that under Assumption 1, $\ell^{sub}(\theta)$ becomes a convex function of θ. To the contrary, $\ell_t^{pre}(\theta)$ and $\ell_t^{est}(\theta)$ are not guaranteed to be convex; see Appendix C.

Lemma 1. *Under Assumption 1, $\ell_t^{sub}(\theta)$ is convex in θ for every u_t, y_t and $t \in [T]$.*

Online inverse learning based on these conventional loss functions has potential shortcomings. The non-convexity of ℓ^{est} and ℓ^{pre} in general cases means additional assumptions are necessary to design OCO algorithms with respect to both functions. ℓ^{sub} has the desirable convexity, but ℓ^{sub}-based OCO algorithms may encounter computational issues due to the inverse term $x(\theta; u)$; see Appendix G for further discussion. Moreover, literature has yet to provide a general online inverse learning framework that can handle all three loss functions simultaneously.

We next utilize the structure of f in Assumption 1 to design a new loss function with practical connections with the conventional losses. This new loss function further enables a more flexible online inverse learning framework.

Definition 1. Suppose Assumption 1 holds. We define the *simple loss* as

$$\ell^{sim}(\theta, x(\theta_t; u_t); y_t, u_t) := \langle \theta, c(y_t) - c(x(\theta_t; u_t)) \rangle + \langle \theta_{true}, c(x(\theta_t; u_t)) - c(y_t) \rangle.$$

□

Let $\ell_t^{sim}(\theta) := \ell^{sim}(\theta, x(\theta_t; u_t), y_t, u_t)$. In $\ell_t^{sim}(\theta)$, the term $x(\theta_t; u_t)$ is the optimal solution to (2) with given θ_t and u_t, and it can be viewed as a prediction of the agent's action at the current estimate θ_t of the true parameter θ_{true}. Hence, when ℓ^{sim} is used as the loss function in online inverse optimization, at each time step t, $\ell_t^{sim}(\theta)$ has an explicit dependence on the revealed signal u_t, the observation y_t of agent's true optimal action, and the predicted action $x(\theta_t; u_t)$ using the estimate θ_t generated based on the information from previous $t-1$ time steps and the signal u_t. Here, it is noteworthy to highlight that since θ_t is determined before the function $\ell_t^{sim}(\theta)$ is revealed, there is no cyclic dependence between them.

Next, we note that $\ell_t^{sim}(\theta)$ is a convex function of θ, which is important for its use in our online inverse optimization framework. We will demonstrate in Section 4 that under Assumption 1, regret minimization based on $\ell^{sim}(\theta)$ also leads to performance guarantees with respect to the all other loss functions.

Lemma 2. *Under Assumption 1, $\ell_t^{sim}(\theta)$ is linear (hence convex) in θ for every $t \in [T]$.*

4 Regret Performance for ℓ^{sim}-based Online Inverse Optimization

We will develop an OCO-based framework for preference learning via online inverse optimization. To this end, we have already introduced a loss function, i.e., ℓ^{sim}, that is convex under Assumption 1. In this section, we show that in the perfect information setup (i.e., when there is no noise on the observations $y_t = x(\theta_{true}, u_t)$ for all t), the regret with respect to ℓ^{sim} indeed bounds the regrets with respect to all other loss functions of interest as well. Although restrictive, perfect information setting is still relevant in practice in settings such as a streaming platform having observations of users' true actions described in Introduction. For completeness, we discuss the *imperfect information* case from both theoretical and empirical (see Section 6.4) aspects, and provide preliminary theoretical explanations of the observed empirical performances in Appendix F.

4.1 Perfect Information

Our result establishes a fundamental guarantee among the regret bounds with respect to $\ell_t^{sim}, \ell_t^{sub}$ and ℓ_t^{est}.

Proposition 1. *Suppose Assumption 1 holds and there is no noise on the observations. Then, for any sequence $\{\theta_t\}_{t\in[T]}$, we have*

$$\begin{aligned}
&(a)\ R_T(\{\ell_t^{sub}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}),\ R_T(\{\ell_t^{est}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}),\ \text{and } R_T(\{\ell_t^{pre}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) \geq 0,\\
&(b)\ R_T(\{\ell_t^{sub}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) + R_T(\{\ell_t^{est}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) = \textstyle\sum_{t=1}^{T} \ell_t^{sim}(\theta_t),\\
&(c)\ R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) \geq R_T(\{\ell_t^{sub}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) + R_T(\{\ell_t^{est}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}).
\end{aligned}$$

As a consequence of (c), $R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$ upper bounds both $R_T(\{\ell_t^{sub}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$ and $R_T(\{\ell_t^{est}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$.

When f is a strongly convex function in x, (Mohajerin Esfahani et al. 2018, Proposition 2.5) shows that $\ell_t^{sub}(\theta) \geq \frac{\gamma}{2}\ell_t^{pre}(\theta)$ for all t and for all $\theta \in \Theta$, with γ being the strong convexity parameter of f. Hence, this result enables us to derive a further regret bound for the loss functions $\{\ell_t^{pre}\}_{t\in[T]}$.

Corollary 1. *Suppose Assumption 1 holds and there is no noise. Assume further that f is a strongly convex function of x for every θ, i.e., there exists $\gamma > 0$ such that $f(x;\theta,u) - f(y;\theta,u) \geq \langle s_y, x-y\rangle + \frac{\gamma}{2}\|x-y\|^2$, where s_y is a subgradient of $f(y;\theta,u)$ with respect to y. Then, for any sequence $\{\theta_t\}_{t\in[T]} \subseteq \Theta$ we have $R_T(\{\ell_t^{sub}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) \geq \frac{\gamma}{2} R_T(\{\ell_t^{pre}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$.*

Assumption 1 ensures that ℓ_t^{sim} is a convex function of θ, and thus any deterministic OCO algorithm will be applicable for regret minimization with respect to $\{\ell_t^{sim}\}_{t\in[T]}$. Then, as a consequence of Proposition 1 (and Corollary 1), such algorithms will also be minimizing regret with respect to the loss functions $\ell_t^{sub}, \ell_t^{est}$ (and ℓ_t^{pre}), as well.

Remark 3. The regret bounds with respect to these loss functions have the following implications under perfect information. A sublinear regret bound with respect to ℓ^{sim} implies that the average loss incurred by the estimates $\{\theta_t\}$ approaches the offline optimal loss over time. Sublinear regret bounds with respect to ℓ^{sub} and ℓ^{est} indicate that the learner is able to generate estimates $\{\theta_t\}$ that lead to vanishing errors in the predicted agent's objective function values. A sublinear regret bound with respect to ℓ^{pre} additionally indicates that the average $\|\cdot\|_2$-norm distance between the predicted agent's action and her/his true action decreases to zero over time. Note that none of these regret guarantees in particular ensures that the $\{\theta_t\}$ generated from the online learning process are good approximations of θ_{true}. In general, this is an overly ambitious task as (Bärmann et al. 2017, Example 3.2) has shown a simple case where the exact recovery of θ_{true} cannot be guaranteed. We note that stronger performance guarantees, such as $\frac{1}{T}\sum_{t\in[T]}\|\theta_t - \theta_{true}\| \to 0$ may be possible for special cases, for example, when $x(\theta; u_t)$ has a closed form expression as a continuous function in θ. In addition, in certain cases the optimal actions from the forward problem may be non-unique, our framework is not aiming to predict the chosen action $x(\theta_{true}; u_t)$, instead, we measure the learning performance with regret values based on the objective value of the agent. □

4.2 Imperfect Information

The case when the learner has access to only imperfect information about the agent's actions is of natural interest as well. Mohajerin Esfahani et al. (2018) identify two types of noisy information as of interest: (i) *measurement noise*, that is, for all $t \in [T]$, the learner observes $y_t = x(\theta_{true}, u_t) + \epsilon_t$ with ϵ_t denoting a random noise, and (ii) *bounded rationality*, which means for all $t \in [T]$, the agent may choose a sub-optimal action instead of $x(\theta_{true}, u_t)$. Such imperfect information does not affect the convexity property of loss functions, and so both ℓ^{sub} and ℓ^{sim} remain convex (see Lemma 1 and 2) still enabling the use of OCO algorithms for regret minimization with respect to these loss functions. For instance, we can still apply a ℓ^{sim}-based OCO algorithm to minimize regret with respect to ℓ^{sim}.

However, since y_t is no longer guaranteed to be a minimizer of (1) with $u = u_t$, Proposition 1 does not hold in general, and consequently $R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$ is not guaranteed to bound the regrets with respect to the other loss functions. In addition, due to the noises in y_t, for any of the four loss definitions, its associated regret $R_T(\{\ell_t\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$ can no longer accurately measure learning performance with respect to the agent's true actions and objective values. In our investigation of the imperfect information setting in Section 6.4 and Appendix F, instead of regrets, we consider different loss-based performance measures.

5 Online Learning Algorithms

Under Assumption 1, both $\ell^{sim}(\theta)$ and $\ell^{sub}(\theta)$ are convex in θ, so online inverse optimization with respect to either loss function can be done in an OCO framework. We equip our framework with two well-known deterministic OL regret minimization algorithms utilizing different oracles: Online *Mirror Descent* (MD), which is a classical OCO algorithm that utilizes a first-order oracle, and the implicit OL algorithm introduced in Kulis and Bartlett (2010) that is based on a solution oracle.

To take advantage of the unifying capability from $R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$, here we will focus on ℓ^{sim} as our loss function. In fact, for OL with a first order oracle, our ℓ^{sim}-based algorithm coincides with an ℓ^{sub}-based method, and when a solution oracle is used, the ℓ^{sim}-based algorithm is arguably superior; see Appendix G. Under Assumption 1 and with perfect information, both algorithms generate regret bounds with respect to ℓ^{sim}, which then imply regret bounds with respect to the other loss functions as well (see Section 3).

For exposition convenience, we state all of these algorithms in the same online setup: the learner receives observations $\{y_t, u_t\}_{t\in[T]}$ and generates $\{\theta_t\}_{t\in[T]} \subseteq \Theta$ to minimize the regret $R_T(\{\ell_t\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$.

5.1 Online Convex Optimization with First-Order Oracle

We review the well-known first-order OCO algorithm, namely the online *Mirror Descent* (MD) algorithm in the proximal setup. We follow the presentation and notation of Juditsky et al. (2011) and define the following standard components of the proximal setup:

- *Norm*: $\|\cdot\|$ on the Euclidean space $\mathbb{E}$ where the domain Θ lives, along with its dual norm $\|\zeta\|_* := \max_{\|\theta\|\le 1} \langle \zeta, \theta\rangle$.
- *Distance-Generating Function* (d.g.f.): A function $\omega(\theta) : \Theta \to \mathbb{R}$, which is convex and continuous on Θ, and admits a selection of subdifferential $\partial\omega(\theta)$ that is continuous on the set $\Theta^\circ := \{\theta \in \Theta : \partial\omega(\theta) \neq \varnothing\}$, and is strongly convex with modulus 1 with respect to $\|\cdot\|$:

$$\forall \theta', \theta'' \in \Theta^\circ : \ \langle \partial\omega(\theta') - \partial\omega(\theta''),\ \theta' - \theta''\rangle \ge \|\theta' - \theta''\|^2.$$

- *Bregman distance*: $V_\theta(\theta') := \omega(\theta') - \omega(\theta) - \langle \partial\omega(\theta), \theta' - \theta\rangle$ for all $\theta \in \Theta^\circ$ and $\theta' \in \Theta$.

  Note the strong convexity of ω implies $V_\theta(\theta') \ge \frac{1}{2}\|\theta - \theta'\|^2 \ge 0$ for all $\theta \in \Theta^\circ$ and $\theta' \in \Theta$.
- *Prox-mapping*: Given a *prox center* $\theta \in \Theta^\circ$,

$$\mathrm{Prox}_\theta(\xi) := \arg\min_{\theta' \in \Theta} \{\langle \xi, \theta'\rangle + V_\theta(\theta')\} : \mathbb{E} \to \Theta^\circ.$$

  When the d.g.f. is taken as the squared ℓ_2-norm, the prox mapping becomes the usual projection operation of the vector $\theta - \xi$ onto Θ.
- *ω-center*: $\theta_\omega := \arg\min_{\theta\in\Theta} \omega(\theta)$.
- *Set width*: $\Omega := \max_{\theta\in\Theta} V_{\theta_\omega}(\theta) \le \max_{\theta\in\Theta} \omega(\theta) - \min_{\theta\in\Theta} \omega(\theta)$.

When functions $\{\ell_t^{sim}(\theta)\}_{t\in[T]}$ are convex in θ, online MD as stated in (Ho-Nguyen and Kılınç-Karzan 2019, Algorithm 1) is applicable to guarantee a sublinear regret bound on $R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$, which further bounds regrets with respect to the other loss functions as discussed in Section 4.

Theorem 1. *(Ho-Nguyen and Kılınç-Karzan 2019, Theorem 1) Suppose Θ is convex and $\ell_t : \Theta \mapsto \mathbb{R}$ is a convex function for $t \in [T]$. Suppose there exists $G \in (0, \infty)$ such that all the subgradients s_t of ℓ_t are bounded, i.e., $\max_{s_t \in \partial\ell_t(\theta)} \|s_t\|_* \le G$ for all $\theta \in \Theta$ and $t \in [T]$. Let the step size η_t be chosen as $\eta_t = \frac{2\Omega}{G^2 T}$. At time step t, using the online Mirror Descent algorithm, we generate θ_{t+1} as*

$$\theta_{t+1} := \mathrm{Prox}_{\theta_t}(\eta_t s_t) = \arg\min_{\theta\in\Theta} \{\langle \eta_t s_t, \theta\rangle + V_{\theta_t}(\theta)\}, \tag{5}$$

where $s_t \in \partial\ell_t(\theta_t)$. Then the sequence $\{\theta_t\}_{t\in[T]}$ satisfies $R_T(\{\ell_t\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) \le \sqrt{2\Omega G^2 T}$.

In applying Theorem 1 to the loss functions $\{\ell_t^{sim}\}_{t\in[T]}$, we have the subgradient $s_t = c(y_t) - c(x(\theta_t; u_t))$. So, it suffices to set $G \ge 2\max\{\|c(x)\|_* : x \in \mathcal{X}\}$. The set width Ω depends on Θ only and can be computed for a given Θ and Bregman distance explicitly.

5.2 Implicit Online Learning with a Solution Oracle

We next review the implicit OL with a solution oracle from Dong et al. (2018a). This algorithm was first introduced in its general form in Kulis and Bartlett (2010).

The *implicit online learning* algorithm computes

$$\theta_{t+1} \coloneqq \operatorname{argmin}_{\theta\in\Theta} L_t(\theta), \tag{6}$$

where $L_t(\theta) = V_{\theta_t}(\theta) + \eta_t \ell_t(\theta)$ and $V_\theta(\theta')$ is the Bregman distance, η_t is a step size. This approach does not rely on the first-order oracle on ℓ_t, but rather assumes the existence of a *solution oracle* to solve (6). Kulis and Bartlett (2010) establish the following regret bound on the OL using implicit update (6).

Theorem 2. *(Kulis and Bartlett 2010, Theorem 3.2.) Suppose Θ is convex, and $\ell_t : \Theta \mapsto \mathbb{R}$ is a convex and differentiable function for $t \in [T]$. Let θ^* be the offline optimal solution to $\min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t(\theta)$. For any $0 < \alpha_t \leq \frac{L_t(\theta_{t+1})}{L_t(\theta_t)}$ for $t \in [T]$, for any step size $\eta_t > 0$, an implicit OL algorithm with the update rule (6) attains*

$$R_T(\{\ell_t\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) \leq \sum_{t\in[T]} \frac{1}{\eta_t} \left[(1-\alpha_t)\eta_t \ell_t(\theta_t) + V_{\theta_t}(\theta^*) - V_{\theta_{t+1}}(\theta^*)\right]. \tag{7}$$

When ℓ_t is a convex and Lipschitz continuous function of θ and the domain Θ has a finite width with respect to the selected Bregman divergence, the regret bound (7) further results in a $O(\sqrt{T})$ bound on $R_T(\{\ell_t\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$.

Theorem 3. *Suppose Θ is convex, and for each $t \in [T]$, $\ell_t : \Theta \mapsto \mathbb{R}$ is a convex function of θ that is uniformly Lipschitz continuous with parameter G, and suppose $\max_{\theta_1,\theta_2\in\Theta} V_{\theta_1}(\theta_2) \leq \widehat{\Omega}$. Then, by choosing $\eta_t = \frac{\sqrt{\widehat{\Omega}}}{G}\frac{1}{\sqrt{t}}$ for $t \in [T]$, an implicit OL algorithm with the update rule (6) attains*

$$R_T(\{\ell_t\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) \leq 2\sqrt{\widehat{\Omega} G^2 T}. \tag{8}$$

To apply Theorem 3, we can choose the Lipschitz parameter G by definition. For instance, with the loss functions $\{\ell_t^{sim}\}_{t\in[T]}$, we have $|\ell_t^{sim}(\theta) - \ell_t^{sim}(\theta')| = |\langle \theta - \theta', c(y_t) - c(x(\theta_t; u_t))\rangle| \leq \|\theta - \theta'\| \|c(y_t) - c(x(\theta_t; u_t))\|$, hence $G \geq 2\max\{\|c(x)\|_* : x \in \mathcal{X}\}$ suffices. Alternatively, with $\{\ell_t^{pre}\}_{t\in[T]}$, $|\ell_t^{pre}(\theta) - \ell_t^{pre}(\theta')| = |\langle x(\theta'; u_t) - x(\theta; u_t), 2y_t - x(\theta'; u_t) - x(\theta; u_t)\rangle| \leq \|x(\theta'; u_t) - x(\theta; u_t)\| \|2y_t - x(\theta'; u_t) - x(\theta; u_t)\|$, so we need additional information about $x(\theta; u_t)$ to decide a suitable G. The set width $\widehat{\Omega}$ only depends on Θ and the Bregman distance definition.

5.3 Comparison with the Existing Approaches

Bärmann et al. (2017) study online inverse optimization under perfect information where the agent's objective f is a bilinear function of θ and x, i.e., $f(x; \theta) = \langle \theta, x\rangle$. They suggest using the online gradient descent and the Multiplicative Weights Update (MWU) algorithms to generate $\{\theta_t\}_{t\in[T]}$ estimates and show via separate analysis that the resulting estimates have vanishing average losses with respect to ℓ^{est} and ℓ^{sub} (at the rate $O(1/\sqrt{T})$) but do not present their regret bounds or analyze ℓ^{pre} loss. Note that both online gradient descent and MWU algorithm are simply special cases of the online MD algorithm customized to the geometry of the problem domain. Moreover, the setting studied in Bärmann et al. (2017) clearly satisfies our Assumption 1 and the perfect information assumption, hence we can utilize our ℓ^{sim}-based OL framework equipped with online MD and directly derive average regret bounds of $O(1/\sqrt{T})$ on ℓ^{sim}, ℓ^{sub} and ℓ^{est}. In addition, as opposed to the simple bilinear form of f considered in Bärmann et al. (2017), our framework can handle more general functions f in the forward problem when $f_1(x; u)$ and/or $f_2(\theta; u)$ are nontrivial. In this respect, it is of interest to study the case of strongly convex $f_1(x; u)$, where through Corollary 1, our framework also leads to regret bound with respect to ℓ^{pre}.

Dong et al. (2018a) study the following problem where f is linear in θ and strongly convex in x

$$\min_x \left\{\tfrac{1}{2} x^\top P x - \langle \theta_{true}, x\rangle : \ x \in \mathcal{X}(u)\right\}. \tag{9}$$

Here, P is a positive definite matrix and $\mathcal{X}(u)$ is the agent's feasible domain determined by the external signal fixed as u. In this setting, Dong et al. (2018a) propose a regret minimization algorithm utilizing the implicit OL method (Kulis and Bartlett (2010)) with a nonconvex MISOCP oracle. They focus on the prediction loss ℓ^{pre}, and establish a $O(\sqrt{T})$ bound on $R_T(\{\ell_t^{pre}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$ whenever $\ell_t^{pre}(\theta)$ is a convex function of θ. A main limitation of their approach is that the convexity of ℓ^{pre} does not hold in general. Although they identify a technical sufficient condition (Dong et al. 2018a, Assumption 3.3) that can guarantee convexity of ℓ^{pre}, they also remark that this condition is restrictive and very hard to verify in practice even for the simplest form of problem classes. In fact, the only example they identify as satisfying their assumption is when the agent's optimization problem is (9) and the set $\mathcal{X}(u)$ must *always* contain the minimizer of the unrestricted objective minimization problem, i.e., $P^{-1}\theta_{true} \in \mathcal{X}(u)$ for all possible u.

When the agent's problem has the specific form of (9), the algorithm from Dong et al. (2018a) updates θ_{t+1} as the optimal solution of the following bilevel program:

$$\theta_{t+1} := \arg\min_{\theta\in\Theta} \left\{ \frac{1}{2}\|\theta - \theta_t\|^2 + \eta_t \|y_t - x(\theta; u_t)\|^2 : x(\theta; u_t) \in \arg\min_x \left\{ \frac{1}{2}x^\top P x - \langle \theta, x\rangle : x \in \mathcal{X}(u_t) \right\} \right\}.$$

It was shown in Dong et al. (2018a) that when the feasible domain $\mathcal{X}(u_t)$ is polyhedral, this bilevel program can be represented as a MISOCP. Consequently, the implicit OL algorithm of Dong et al. (2018a) utilizes an MISOCP based solution oracle to generate $\{\theta_t\}_{t\in[T]}$. The main convergence result (Dong et al. 2018a, Theorem 3.2) proves that under their assumptions by choosing the step size $\eta_t \propto 1/\sqrt{t}$, the sequence of estimates $\{\theta_t\}_{t\in[T]}$ generated with the above update yields a $O(\sqrt{T})$ bound on the regret $R_T(\{\ell_t^{pre}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$.

Note that the format of f in (9) satisfies our Assumption 1, and consequently ℓ_t^{sim} is guaranteed to be convex for any $\mathcal{X}(u)$. Therefore, our OCO framework based on minimizing regret for loss functions $\{\ell_t^{sim}\}_{t\in[T]}$ is applicable to (9). In addition, in the perfect information setting, through Proposition 1 and Corollary 1, our framework can provide regret bounds with respect to *all* of ℓ^{sim}, ℓ^{est}, ℓ^{sub} and ℓ^{pre}, without further structural assumptions on the agent's domain. In contrast, the implicit OL approach of Dong et al. (2018a) for minimizing regret with respect to ℓ^{pre} requires additional conditions on the agent's domain (see (Dong et al. 2018a, Assumptions 3.1, 3.2, 3.3)) in order to guarantee a regret bound. In particular, it is specifically focused on ℓ^{pre} and provides no insight on other performance measures of interest captured by ℓ^{est} and ℓ^{sub} either. Moreover, online MD in our framework uses a much simpler (and computationally faster) first-order oracle in contrast to the expensive MISOCP oracle in the implicit OL approach of Dong et al. (2018a). One aspect that Dong et al. (2018a) emphasize is the noise in observations: they prove theoretical expected regret bounds with respect to ℓ^{pre} when y_t is a noisy observation of $x(\theta_{true}; u_t)$, but the theoretical guarantees of our framework based on ℓ^{sim} do not readily extend to the imperfect information setup. We caution that their analysis for the noisy setup is subject to the restrictive conditions needed to ensure convexity of $\ell_t^{pre}(\theta)$, so it might not provide meaningful performance guarantees in general.

6 Computational Study

We perform numerical experiments on a practical application that is motivated by a company (learner) seeking to learn about its customer's (agent's) preferences in a changing market. We assume the customer is a rational decision maker, and in any given market situation, her/his action reflects accurately her/his optimal preferences. These experiments do not aim to provide structural insights on specific instances, rather, our main purpose is to demonstrate the performance of ℓ^{sim} based OCO algorithms from various aspects and the comparison with an alternative ℓ^{pre}-based approach in Dong et al. (2018a).

We first focus on the case when perfect information is available, i.e., there is no noise in learner's observations of the agent's optimal actions, and address three main questions. First, are there notable performance differences among OL algorithms based on different oracles? Second, how do the algorithm performances vary in terms of different loss functions? Third, does the structure of the agent's feasible region affect complexity of the learning problem and the algorithm performances? While discussing these questions, we also compare against existing algorithms from the literature.

In the second part of our numerical study, we examine the robustness of these OL algorithms under imperfect information, i.e., when there is random noise to the learner's observations of the agent's optimal actions. Recall that in the imperfect information setup, our OL based approach is not guaranteed to provide low regret guarantees, so these experiments essentially shed light to their empirical performance in the noisy setup. Despite the lack of theoretical bounds, we can analyze the OL algorithms to explain the observed empirical trends in losses and regrets (see Appendix F).

All algorithms are coded in Python 3.8, and Gurobi 8.1.1 with default settings is used to solve the mathematical programs needed for the subproblems associated with the corresponding oracles. We limit the solution time of each mathematical program to be at most 3600 seconds. We have not hit this imposed time limit in any of our experiments. All experiments are conducted on a server with 2.8 GHz processor and 64GB memory.

6.1 Problem Instances

We consider a market with n products that evolves over a finite time horizon T, e.g., the product prices change. These changes consequently impact the agent's feasible actions; in this case, agents are customers interested in purchasing the products. For each $t \in [T]$, we let u_t denote the market parameters relevant to the agent's decisions at period t. When constraint parameters are fixed as u_t, an agent's action $x(\theta_{true}; u_t)$ is an optimal solution to an optimization problem parametrized by u_t and θ_{true}, where θ_{true} captures the agent's preferences over the products. We model the agent's optimization problem as a maximization of her/his utility function subject to feasibility constraints. The learner knows the agent's decision problem up to the parameter vector θ_{true}, and the learner's goal is to estimate θ_{true} using observations of the agent's actions y_t in response to the market conditions u_t at each period $t \in [T]$.

We study two different forms for the agent's utility function.

(a) For direct comparison with Dong et al. (2018a), we examine the case where the agent's utility function has the quadratic form (9), i.e., the agent's action $x(\theta_{true}; u_t)$ is given by

$$x(\theta_{true}; u_t) \coloneqq \arg\max_x \left\{ -\frac{1}{2} x^\top P x + \langle \theta_{true}, x \rangle : \ x \in \mathcal{X}(u_t) \right\}, \tag{10}$$

where $P \in \mathbb{S}^n_{++}$ is a fixed positive definite matrix known by both the learner and the agent and $\mathcal{X}(u_t)$ represents the domain for the agent's feasible actions determined by the market parameters u_t.

(b) We also examine a second setup where the agent has a CES utility function with $\rho = 2$. Hence, in period t, the agent's action $x(\theta_{true}; u_t)$ is given by

$$x(\theta_{true}; u_t) \coloneqq \arg\max_x \left\{ \sum_{i \in [n]} -(\theta_{true})_i x_i^2 : \ x \in \mathcal{X}(u_t) \right\}. \tag{11}$$

Note that this setup with a CES utility has not been previously studied in an OL framework.

These particular forms of utility functions in (10) and (11) imply that the dimensions of θ and x are the same, i.e., $p = n$. Moreover, observe that both of the objective functions in (10) and (11) satisfy Assumption 1, and thus in both cases $\ell_t^{sim}(\theta)$ is convex in θ.

To identify the impact of agent's feasible region on the complexity of the problem and on the performance of the learning algorithms, we experiment on a variety of settings for $\mathcal{X}(u_t)$.

(i) *Continuous knapsack* domain: in this setting, we impose only a budget constraint on the agent: $\mathcal{X}(u_t) = \mathcal{X}^{ck}(p_t, b_t) \coloneqq \{x \in \mathbb{R}^n_+ : \langle p_t, x \rangle \leq b_t\}$, where the parameters $p_t \in \mathbb{R}^n_+$ correspond to the product prices and $b_t \in \mathbb{R}_+$ is the budget available to the customer during time period t. Note that both p_t and b_t can vary in each time period $t \in [T]$.

(ii) *Continuous polytope* domain: here, we generalize the continuous knapsack domain and model general resource constraints resulting in a polytope as the feasible region $\mathcal{X}(u_t) = \mathcal{X}^{cp}(A_t, c_t) = \{x \in \mathbb{R}^n_+ : A_t x \leq c_t\}$, where all the parameters are nonnegative.

(iii) *Binary knapsack* domain: in this case, we again impose a single budget constraint, but also require that the agent's action is a binary vector: $\mathcal{X}(u_t) = \mathcal{X}^{bk}(p_t, b_t) \coloneqq \{x \in \{0,1\}^n : \langle p_t, x\rangle \leq b_t\}$.

(iv) *Equality constrained knapsack* domain: that is, $\mathcal{X}(u_t) = \mathcal{X}^{eck}(p_t, b_t) \coloneqq \{x \in \mathbb{R}^n_+ : \langle p_t, x\rangle = b_t\}$.

We ran experiments with the utility function (10) where we choose the matrix P to be a positive definite diagonal matrix and generate each of its diagonal entries P_{ii} by first drawing a number from $[1, 21]$ uniformly and then normalizing the drawn vector $(P_{11}, \ldots, P_{nn})$ to have a unit ℓ_1-norm, and we also set the domain to be $\mathcal{X}^{ck}(p_t, b_t)$, $\mathcal{X}^{cp}(A_t, c_t)$, or $\mathcal{X}^{bk}(p_t, b_t)$. In the case of CES utility function (11), for implementation simplicity, we use instances with the domain $\mathcal{X}^{eck}(u_t)$.

In all of our experiments, we consider a market with $n = 50$ goods. We compare OL algorithms by running $T = 500$ iterations on a batch of 50 randomly generated instances for each setting. The domain Θ is set be a unit simplex, i.e., $\Theta = \left\{\theta \in \mathbb{R}^n_+ : \sum_{i\in[n]} \theta_i = 1\right\}$. We follow the same instance generation methodology used in (Bärmann et al. 2017, Section 4.1) for generating the true parameter θ_{true} and the agent's domain $\mathcal{X}(u_t)$. In each instance, θ_{true} is obtained by drawing a random sample from a uniform distribution over $[1, 1000]^n$ and then normalizing the sampled vector to have a unit ℓ_1-norm. In the case of $\mathcal{X}^{ck}(p_t, b_t)$, $\mathcal{X}^{bk}(p_t, b_t)$, and $\mathcal{X}^{eck}(p_t, b_t)$, for all $t \in [T]$, the constraint parameters p_t, b_t are generated randomly as follows: p_t is set as $\theta_{true} + 100 \cdot \mathbf{1}_n + r$, where r is an integer vector sampled from a *discrete uniform* distribution over the collection of integer vectors in $[-10, 10]^n$ (*numpy.random.randint* function is used). The budget b_t is selected uniformly random from the range $[1, \sum_{i=1}^n (p_t)_i]$. In the case of continuous polytope domain $\mathcal{X}^{cp}(A_t, c_t)$, we choose A_t as an $m \times n$ matrix with $m = 10$, where each row of A_t is generated in the same way as p_t, and each coordinate i in the vector c_t is drawn uniformly random from $[1, \sum_{j=1}^m (A_t)_{ji}]$.

In the OL setup, at time step t, the learner observes the signal u_t and the agent's action, and uses the information revealed so far to construct the estimate θ_{t+1}. Under perfect information, we have $y_t = x(\theta_{true}; u_t)$ for all t; under imperfect information, we assume $y_t = x(\theta_{true}; u_t) + \epsilon_t$, where ϵ_t denotes the random noise. In the noisy setup, each coordinate in ϵ_t is randomly drawn from a uniform distribution over a given range. We consider two ranges to simulate small and large noises, and we choose the range bounds based on the average $\delta := \frac{1}{T}\sum_{t\in[T]} \|x(\theta_{true}; u_t)\|$: the small noises are generated with the range $[-\delta/n, \delta/n]$, and the large noises are generated with $[-\delta, \delta]$.

6.2 Implementation Details

In order to compute the estimates $\{\theta_t\}_{t\in[T]}$, we implement three OL algorithms and compare their performances. By taking advantage of the convexity of ℓ^{sim}, we design two OL algorithms minimizing regret with respect to ℓ^{sim}: one equipped with a first-order oracle and one with a solution oracle. In addition, for comparison with the literature, we implemented another implicit OL algorithm with a solution oracle aimed to minimize the regret associated with ℓ^{pre}, i.e., the one from Dong et al. (2018a) that utilizes an MISOCP solution oracle. We provided precisely the same dynamic observations, i.e., the realizations of signals u_t along with the agent's optimum action $x(\theta_{true}; u_t)$ in each iteration $t \in [T]$ to all of these OL algorithms.

In the case of OL with the first-order oracle, because Θ is a unit simplex, we use the negative entropy function $\omega(\theta) = \sum_{i=1}^n \theta_i \ln(\theta_i)$ as the distance generating function in the definition of Bregman distance $V_{\theta_t}(\theta)$. Then, the update rule (5) for the OL with first-order oracle is given explicitly by the following formula, where $s_t(\theta_t)$ is a subgradient of $\ell_t^{sim}(\theta)$ at θ_t.

$$(\theta_{t+1})_i = \frac{(\theta_t)_i \exp(-\eta_t(s_t(\theta_t))_i)}{\sum_{j=1}^n (\theta_t)_j \exp(-\eta_t(s_t(\theta_t))_j)}, \text{ for all } i \in [n].$$

The main challenge in the implementation of implicit OL algorithm with a solution oracle is whether one can design a computationally tractable solution oracle. When the loss function $\ell_t(\theta)$ used in the implicit OL involves $x(\theta; u_t)$, as is the case in all loss functions from Section 3 except $\ell_t^{sim}(\theta)$, (6) is a bilevel program. Bilevel programs are difficult to solve in general, but can be reformulated into a single level problem using KKT conditions of the inner level problem whenever the inner level is a convex problem. In contrast to this, when ℓ_t^{sim} is used as the loss function in an implicit OL algorithm, (6) becomes a single level optimization

problem in θ and thus the solution oracle becomes much simpler. Consequently, we study two variants of the implicit OL algorithm based on ℓ^{sim} and ℓ^{pre} that are necessarily equipped with different solution oracles.

In the first variant, we design an implicit OL algorithm to minimize the regret with respect to ℓ^{sim}. Using the squared Euclidean norm as the d.g.f., we arrive at the implicit OL algorithm with a solution oracle that updates θ_{t+1} as the optimal solution to

$$\theta_{t+1} \coloneqq \arg\min_{\theta \in \Theta} \frac{1}{2} \|\theta - \theta_t\|^2 + \eta_t \ell_t^{sim}(\theta).$$

Under Assumption 1, $\ell_t^{sim}(\theta)$ is convex in θ, and when the domain Θ is convex, the above problem is a convex program. Therefore, the implementation requires only a convex solution oracle; see Appendix E for the explicit formulations of these oracles.

For comparison purposes, we implement a second variant of the implicit OL algorithm minimizing regret with respect to the loss function ℓ^{pre}. By following the same approach taken in Dong et al. (2018a), we use the squared Euclidean norm as the d.g.f., and the resulting solution oracle updates θ_{t+1} with the following bilevel program, where the inner level computes $x(\theta, u_t)$ used in $\ell_t^{pre}(\theta)$:

$$\theta_{t+1} \coloneqq \arg\min_{\theta \in \Theta} \frac{1}{2} \|\theta - \theta_t\|^2 + \eta_t \ell_t^{pre}(\theta).$$

When the agent is maximizing a concave objective function over a polyhedral domain $\mathcal{X}(u_t)$, we can reformulate the above bilevel program into a mixed integer program (MIP).

Consequently, at time t, this ℓ^{pre}-based implicit OL algorithm requires a nonconvex oracle given by the MIP formulation to obtain θ_{t+1}. In the case of (10), it was demonstrated in Dong et al. (2018a) that when the domain $\mathcal{X}(u_t)$ is polyhedral, the MIP reformulation admits a nice MISOCP structure due to the quadratic objective. For completeness, we provide the MISOCP reformulation of this solution oracle in Appendix E. Note that due to the advanced capabilities of modern MIP solvers, the resulting MISOCP still remains computationally tractable whenever the scale of the agent's problem is relatively small.

On the other hand, when the domain of the inner problem $\mathcal{X}(u_t)$ is nonconvex, e.g., when we consider $\mathcal{X}^{bk}(p_t, b_t)$ that involves binary variables, or when the agent maximizes a nonconcave function over a convex domain $\mathcal{X}(u_t)$ as in the case of (11) for $\theta \notin \mathbb{R}^n_+$, we no longer have access to KKT based optimality certificates for the inner problem. Consequently, in such cases, we do not know how to design a computationally tractable solution oracle, and this is an open question. Therefore, we did not experiment with the ℓ^{pre}-based implicit OL algorithm in these cases.

6.3 Perfect Information Experiments

In this section, we discuss our numerical results along with plots that highlight our key observations pertinent to the questions of interest to the perfect information case listed at the beginning of Section 6.

6.3.1 Learning a Quadratic Utility Function

In this case, we assume that the agent's utility function is of form (10). We first compare the performance of the three OL approaches in terms of both average regret performance and the solution time. Figures 1 and 2 display the means of average (expected) regret performance of the iterates $\{\theta_t\}_{t\in[T]}$ returned by the OL algorithms with respect to all five loss functions of interest for the instances where the agent's domain is of continuous knapsack and polytope type, respectively; the shaded areas indicate 95% confidence interval for the means. These means are computed based on all 50 random instances generated in the experiment. In Figures 1 and 2 (and all the later ones as well), the scale of loss functions naturally differ because the associated regret and loss values are evaluated with respect to different terms present in their corresponding loss definitions. In terms of the rate at which the average regret converges, in the case of the continuous knapsack instances, Figure 1 shows that regardless of the loss function used to evaluate the performance, all three OL algorithms have quite similar performances. This empirical observation is in line with the theoretical regret guarantees given in Section 5; recall that this particular domain type was the focus of Dong et al.

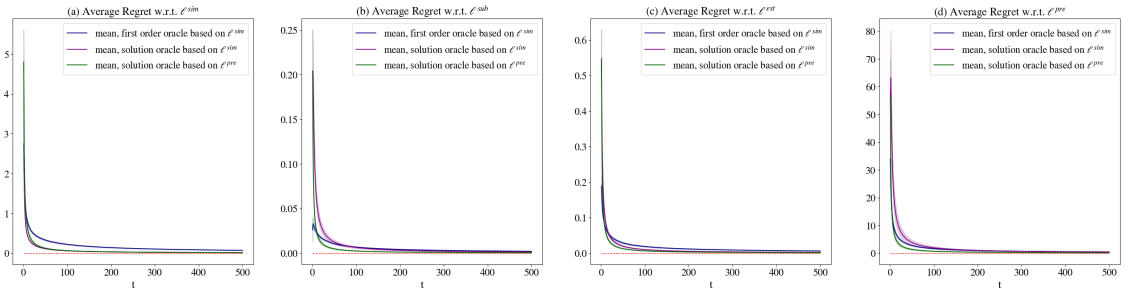


Figure 1: Means of average regret with respect to different loss functions over $T = 500$ iterations for continuous knapsack instances; the shaded region is 95% confidence interval for the means.

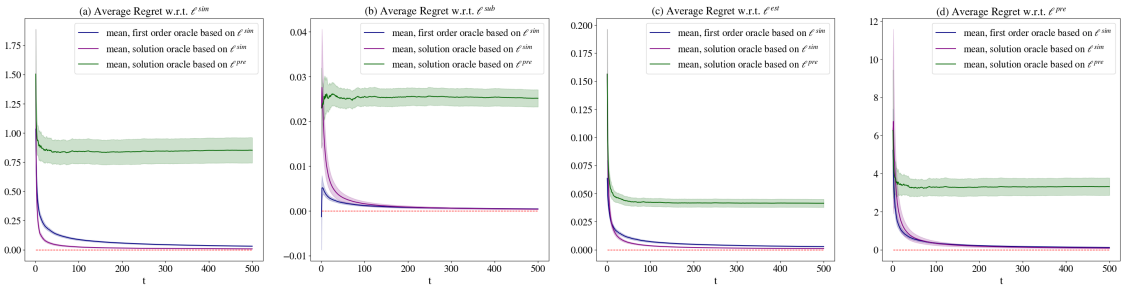


Figure 2: Means of average regrets with respect to different loss functions over $T = 500$ iterations for continuous polytope instances; the shaded region is 95% confidence interval for the means.

(2018a), and their analysis presents some restrictive assumptions guaranteeing convergence of their approach on this type of instances. For the continuous polytope instances, Figure 2 demonstrates similar performances from the two OL algorithms based on ℓ^{sim}, but highlights the drastically different performance of the implicit OL with the ℓ^{pre}-minimizing solution oracle, which now leads to average regrets converging to non-zero values. Recall from Section 5.2, the regret convergence of an implicit OL algorithm with a solution oracle requires the convexity of the selected loss function. In fact, Dong et al. (2018a) adopt further strong assumptions on $\mathcal{X}(u_t)$ to guarantee that ℓ^{pre} is a convex function of θ when the agent's problem is of form (10) with $\mathcal{X}(u_t) = \mathcal{X}^{ck}(p_t, b_t)$. Our empirical results indicate that these assumptions are indeed hard to satisfy in general and our randomly generated continuous polytope instances do not necessarily satisfy their required assumption. In contrast, since ℓ^{sim} is guaranteed to be a convex function of θ when the agent's problem is of form (10) regardless of the structure of the agent's domain $\mathcal{X}(u_t)$, the average regrets of the ℓ^{sim}-based implicit OL algorithm with the solution oracle converge to zero for instances with polytope domain as well. Furthermore, we note that the regret convergence of the ℓ^{sim}-based implicit OL algorithm with the solution oracle is slightly better than the OL with the first-order oracle in both types of instances.

In our numerical study, we observe almost no variation in terms of the solution time of the OL algorithms across different random instances generated from the same setting. We thereby report the time spent by all three OL algorithms on a randomly selected instance from our problem set. When computing the solution time at iteration t, we always ignore the time taken to find $x(\theta_{true}; u_t)$. In iteration t of the OL with the first-order oracle, we account for the time to compute $x(\theta_t; u_t)$ and generate θ_{t+1} using the first-order oracle. Lastly, in each iteration of both of the ℓ^{sim}- and ℓ^{pre}-based implicit OL algorithms with a solution oracle, we account for the time used by the corresponding solution oracles in updating θ_{t+1}. For an arbitrary instance with the continuous knapsack domain, OL with the first-order oracle finishes in about 0.08 seconds, ℓ^{sim}-based implicit OL with the solution oracle takes 2.03 seconds, and ℓ^{pre}-based implicit OL with the solution oracle takes 146 seconds. These highlight that, by a significant margin, our OL algorithms minimizing regret with respect to the loss function ℓ^{sim} and utilizing the first-order oracle and the solution oracle are much more computationally efficient than the ℓ^{pre}-based implicit OL with the MISOCP solution oracle one from Dong et al. (2018a).

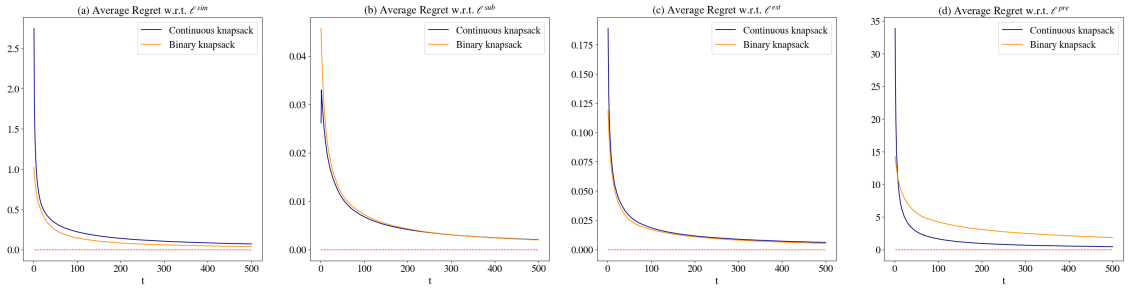


Figure 3: Means of average regret with respect to different loss functions contrasting continuous knapsack instances with binary knapsack instances, when OL with the first-order oracle is used.

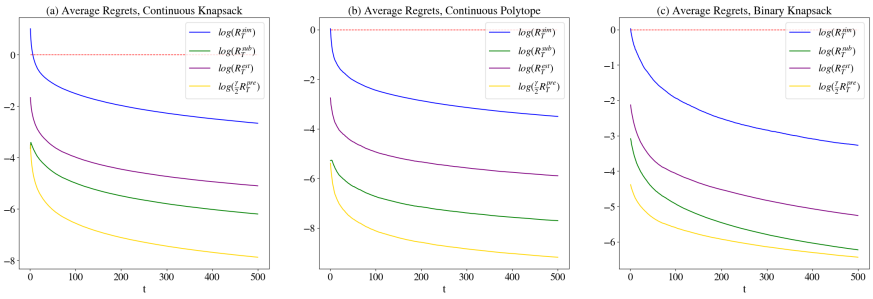


Figure 4: Means of average regret (on a logarithmic scale) with respect to different loss functions over $T = 500$ iterations for (a) continuous knapsack instances, (b) continuous polytope instances, (c) binary knapsack instances, when OL with first-order oracle is used.

We next analyze whether the agent's domain structure has any visible effect on the overall regret performance of the OL with the first-order oracle. From Figures 1 and 2, we observe that the superiority of the OL with first-order oracle in terms of the average regret is slightly more obvious in the continuous knapsack setting than in the polytope setting. In Figure 3, we compare the means of average regrets for the continuous knapsack instances versus the binary knapsack instances. The regret performances with respect to the loss functions ℓ^{sub} and ℓ^{est} seem to vary only slightly when the agent's domain type changes from continuous knapsack to binary knapsack; yet these differences are slightly more noticeable in the case of loss functions ℓ^{pre} and ℓ^{sim}.

Lastly, we examine the regret performance of OL with the first-order oracle with respect to different loss functions $\ell(\theta)$. From the experiment results from continuous knapsack and continuous polytope instances (respectively Figures 1 and 2), we observe that the average regret with respect to any of the four loss functions convergences roughly at the same rate, but the corresponding regret bounds differ in their scales. This is not surprising, as the corresponding regrets are based on different terms, e.g., norms of solutions or objective function values, etc. Moreover, recall from Section 4 that in the perfect information case the following relationship among the regret bounds with respect to different loss functions (here for simplicity in notation, we denote $R_T^{sim} := R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$, etc.) holds: $R_T^{sim} \geq R_T^{sub} + R_T^{est} \geq \frac{\gamma}{2} R_T^{pre} \geq 0$, where γ is the strong convexity parameter of the quadratic objective function in (10). Recall that our instance generation guarantees $P \in \mathbb{S}_{++}^n$, i.e., its smallest eigenvalue $\lambda_{min}(P) > 0$, and then by the definition of strong convexity, we deduce $\gamma = \lambda_{min}(P)$. Figure 4 displays (on a logarithm scale) the means of the average regrets for different loss functions for θ_t estimates generated from the OL with the first-order oracle on instances in which the agent's domain is either a continuous knapsack, polytope, or a binary knapsack type. These results also confirm the theoretical relationship among the regrets for different loss functions we have established in Section 4.

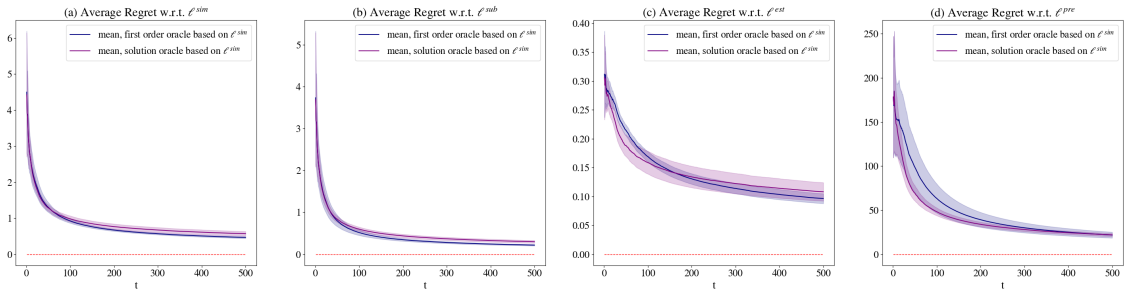


Figure 5: Means of average regrets with respect to different loss functions over $T = 500$ iterations for equality constrained knapsack instances; the shaded region is 95% confidence interval for the means.

6.3.2 Learning a CES Utility Function

Here, we examine the case when the agent's utility function is of form (11) and summarize our findings on the average regrets in Figure 5. We note that the OL with the first-order oracle has a quite noticeable advantage over the implicit OL with a solution oracle in terms of the regret convergence. In this case, on a typical instance, OL with the first-order oracle takes 0.12 seconds to complete and ℓ^{sim}-based implicit OL with the solution oracle takes 2.02 seconds.

6.4 Imperfect Information Experiments

We next study the performance of the two ℓ^{sim}-based OL algorithms when the observations are corrupted with random noise. We test this imperfect information setup on two types of instances where (1) the agent is maximizing a concave quadratic utility function on a continuous knapsack domain, and (2) the agent is maximizing a CES utility function over an equality constrained knapsack domain. We observed that the impact of the noises on the solution time of the OL algorithms was negligible in both of these instance types.

We plot the outcomes differently from the perfect information case. We still show the average regret with respect to ℓ^{sim} to illustrate that the ℓ^{sim}-based OL algorithms remain valid under noises. For ℓ^{sub} and ℓ^{est}, we report the difference between the average loss incurred from $\{\theta_t, x(\theta_t; u_t)\}$ and the average loss from $\{\theta_{true}, x(\theta_{true}; u_t)\}$, where the latter evaluates the loss due to imperfect information. Lastly, we plot the average squared norm distance between $\{x(\theta_t; u_t)\}$ and $\{x(\theta_{true}; u_t)\}$ as a measure of prediction loss.

We report the results for when the agent's problem has the form (10) with the domain $\mathcal{X}(u_t) = \mathcal{X}^{ck}(u_t)$ in Figures 6 and 7. First, we observe that under small noises, the average regret with respect to ℓ^{sim} has a similar convergence trend as in the perfect information experiment results plotted in Figure 1, and the convergence appears to be slower under large noises. For the three loss based measures, both OL algorithms lead to decreasing trends; in particular, the decreasing average squared norm distance between the predicted actions and the true actions indicates that these OL algorithms' predictions of the agent's actions are becoming more accurate as T increases. Not surprisingly, the effects of large noises on performances are more noticeable, and it is worth noting that the OL algorithm with first order oracle outperforms the one with solution oracle under large noises. See Appendix F.1 for the corresponding numerical study in the CES setup.

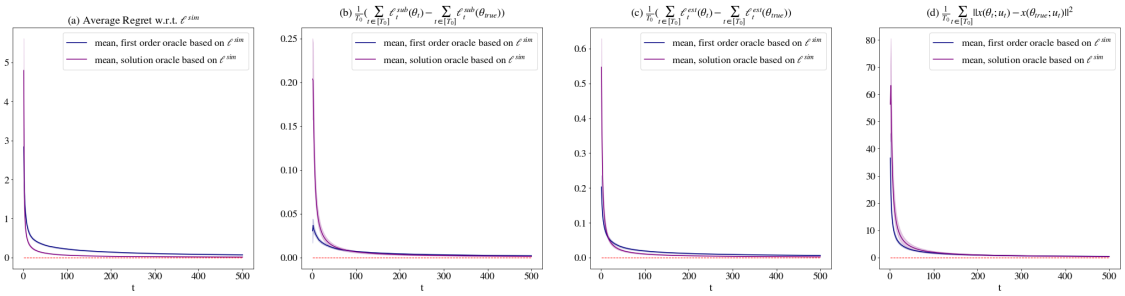


Figure 6: Learning a quadratic utility function under small noises: means of selected performance measures over $T = 500$ iterations for continuous knapsack instances; the shaded region is 95% confidence interval for the means.

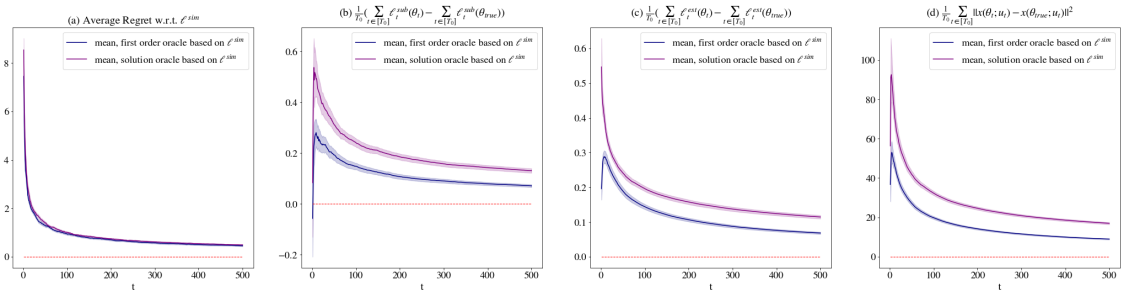


Figure 7: Learning a quadratic utility function under large noises: means of selected performance measures over $T = 500$ iterations for continuous knapsack instances; the shaded region is 95% confidence interval for the means.

References

Farzin Ahmadi, Fardin Ganjkhanloo, and Kimia Ghobadi. 2020. Inverse Learning: A Data-driven Framework to Infer Optimizations Models. *arXiv preprint arXiv:2011.03038* (2020).

Ravindra K Ahuja and James B Orlin. 2001. Inverse optimization. *Operations Research* 49, 5 (2001), 771–783.

Kareem Amin, Rachel Cummings, Lili Dworkin, Michael Kearns, and Aaron Roth. 2015. Online learning and profit maximization from revealed preferences. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Anil Aswani, Zuo-Jun (Max) Shen, and Auyon Siddiq. 2018. Inverse Optimization with Noisy Data. *Operations Research* 66, 3 (2018), 870–892.

Maria-Florina Balcan, Amit Daniely, Ruta Mehta, Ruth Urner, and Vijay V Vazirani. 2014. Learning economic parameters from revealed preferences. In *International Conference on Web and Internet Economics*. Springer, 338–353.

Andreas Bärmann, Sebastian Pokutta, and Oskar Schneider. 2017. Emulating the expert: inverse optimization through online learning. In *Proceedings of the 34th International Conference on Machine Learning (ICML '17)*. 400–410.

Eyal Beigman and Rakesh Vohra. 2006. Learning from Revealed Preference. In *Proceedings of the 7th ACM Conference on Electronic Commerce* (Ann Arbor, Michigan, USA) *(EC '06)*. ACM, New York, NY, USA, 36–42.

Aharon Ben-Tal, Elad Hazan, Tomer Koren, and Shie Mannor. 2015. Oracle-Based Robust Optimization via Online Learning. *Operations Research* 63, 3 (2015), 628–638. arXiv:http://dx.doi.org/10.1287/opre.2015.1374 http://dx.doi.org/10.1287/opre.2015.1374

Timothy CY Chan, Rafid Mahmood, and Ian Yihang Zhu. 2021. Inverse Optimization: Theory and Applications. *arXiv preprint arXiv:2109.03920* (2021).

Chaosheng Dong, Yiran Chen, and Bo Zeng. 2018a. Generalized Inverse Optimization through Online Learning. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NeurIPS '18)*. 86–95.

Chaosheng Dong and Bo Zeng. 2020. Expert Learning through Generalized Inverse Multiobjective Optimization: Models, Insights, and Algorithms. In *Proceedings of the 37th International Conference on Machine Learning*

(Proceedings of Machine Learning Research, Vol. 119), Hal Daumé III and Aarti Singh (Eds.). PMLR, 2648–2657. https://proceedings.mlr.press/v119/dong20f.html

Jinshuo Dong, Aaron Roth, Zachary Schutzman, Bo Waggoner, and Zhiwei Steven Wu. 2018b. Strategic classification from revealed preferences. In *Proceedings of the 2018 ACM Conference on Economics and Computation (EC '18)*. ACM, 55–70.

Elad Hazan. 2019. Introduction to online convex optimization. *arXiv preprint arXiv:1909.05207* (2019).

Clemens Heuberger. 2004. Inverse combinatorial optimization: A survey on problems, methods, and results. *Journal of combinatorial optimization* 8, 3 (2004), 329–361.

Nam Ho-Nguyen and Fatma Kılınç-Karzan. 2018. Online First-Order Framework for Robust Convex Optimization. *Operations Research* 66, 6 (December 2018), 1670–1692. https://doi.org/10.1287/opre.2018.1764

Nam Ho-Nguyen and Fatma Kılınç-Karzan. 2019. Exploiting problem structure in optimization under uncertainty via online convex optimization. *Mathematical Programming* 177, 1-2 (2019), 113–147.

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

Ziwei Ji, Ruta Mehta, and Matus Telgarsky. 2018. Social welfare and profit maximization from revealed preferences. In *International Conference on Web and Internet Economics*. Springer, 264–281.

Anatoli Juditsky, Arkadi Nemirovski, et al. 2011. First order methods for nonsmooth convex large-scale optimization, i: general purpose methods. *Optimization for Machine Learning* (2011), 121–148.

A. Keshavarz, Y. Wang, and S. Boyd. 2011. Imputing a convex objective function. In *2011 IEEE International Symposium on Intelligent Control*. 613–619.

Brian Kulis and Peter L Bartlett. 2010. Implicit online learning. In *Proceedings of the 27th International Conference on Machine Learning (ICML '10)*. 575–582.

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

Aaron Roth, Jonathan Ullman, and Zhiwei Steven Wu. 2016. Watch and learn: Optimizing from revealed preferences feedback. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*. ACM, 949–962.

Andrew J Schaefer. 2009. Inverse integer programming. *Optimization Letters* 3, 4 (2009), 483–489.

Hal R Varian. 2006. Revealed preference. *Samuelsonian economics and the twenty-first century* (2006), 99–115.

Morteza Zadimoghaddam and Aaron Roth. 2012. Efficiently learning from revealed preference. In *International Workshop on Internet and Network Economics*. Springer, 114–127.

A CES and Cobb-Douglas Utility Functions

We illustrate how a CES function and a Cobb-Douglas function can be transformed to satisfy our Assumption 1. This then confirms that our online inverse optimization framework is able to handle both types of utility functions.

A.1 CES Function

For $x \in \mathbb{R}^n_+$, the function $U(x) \coloneqq (\sum_{i=1}^n a_i x_i^\rho)^{1/\rho}$ for some $-\infty < \rho < 0$ or $0 < \rho < \infty$ and $a \in \mathbb{R}^n_+$ such that $\sum_{i=1}^n a_i = 1$ is referred to as a CES function. An economic interpretation of CES functions is provided in Balcan et al. (2014): x represents an outcome where the agent receives amount x_i of good i, and the utility $U(x)$ captures both substituteness and complementarity of the n goods. Here, for consistency of notation, we replace a with θ_{true}.

$U(x)$ is a concave function of $x \in \mathcal{X} \subseteq \mathbb{R}^n_+$ whenever $\rho \in (-\infty, 0) \cup (0, 1]$, and the agent's forward problem maximizes $U(x)$:

$$\max_x \left\{ \left(\sum_{i=1}^n (\theta_{true})_i x_i^\rho \right)^{1/\rho} : \; g(x;u) \le 0, \; x \in \mathcal{X} \right\}.$$

Equivalently, the agent's optimal solution $x(\theta_{true}; u)$ can be obtained with the following systems.

$$x(\theta_{true}; u) = \begin{cases} \arg\min_x \left\{ \sum_{i=1}^n (\theta_{true})_i x_i^\rho : \; g(x;u) \le 0, \; x \in \mathcal{X} \right\}, & -\infty < \rho < 0, \\ \arg\min_x \left\{ -\sum_{i=1}^n (\theta_{true})_i x_i^\rho : \; g(x;u) \le 0, \; x \in \mathcal{X} \right\}, & 0 < \rho \le 1. \end{cases}$$

$U(x)$ is a convex function of $x \in \mathcal{X} \subseteq \mathbb{R}^n_+$ whenever $\rho \in (1, \infty)$, and the agent's forward problem minimizes $U(x)$:

$$\min_x \left\{ \left(\sum_{i=1}^n (\theta_{true})_i x_i^\rho \right)^{1/\rho} : \ g(x;u) \le 0, \ x \in \mathcal{X} \right\},$$

$$\text{and thus } \ x(\theta_{true};u) = \ \arg\min_x \left\{ \textstyle\sum_{i=1}^n (\theta_{true})_i x_i^\rho : \ g(x;u) \le 0, \ x \in \mathcal{X} \right\}, \quad 1 < \rho < \infty.$$

Note that these alternative representations of agent's objective function satisfy Assumption 1.

A.2 Cobb-Douglas Function

For $x \in \mathbb{R}^n_+$, the function $U(x) = \Pi_{i=1}^n x_i^{a_i}$, where $a_i > 0$ and $\sum_{i=1}^n a_i \le 1$, is referred to as a Cobb-Douglas function; see Roth et al. (2016). This utility function can be derived from the CES function by taking $\rho \to 0$. We replace a with θ_{true} for consistency, then an agent with the given Cobb-Douglas utility function chooses her/his optimal action $x(\theta_{true};u)$ as:

$$x(\theta_{true};u) = \arg\max_x \left\{ \sum_{i=1}^n (\theta_{true})_i \log x_i : \ g(x;u) \le 0, \ x \in \mathcal{X} \right\}.$$

We obtain this reformulation by taking logarithm of the product form objective. We immediately observe that Assumption 1 holds for this transformed representation.

B Information Obscuring Agent Objective Example

We give a simple 1-dimensional example of an agent's utility function that obscures information due to its particular choice of $c(x)$. Suppose $\theta_{true} = 0$ and $\Theta = [-3, 3]$, an agent's forward problem is $\min_x\{x + \theta c(x) : x \in [-1, 1]\}$ and let $\mathcal{X}(\theta)$ denote the set of optimal solutions to the agent's problem. With a given θ, the predicted agent action is denoted by $x(\theta) \in \mathcal{X}(\theta)$.

We define

$$c(x) := \begin{cases} 0 & \text{if } x \neq 0 \\ -1 & \text{if } x = 0 \end{cases}.$$

This particular function obscures information on x. The agent's objective function simplifies to x when $x \neq 0$ and $x - 2\theta$ when $x = 0$. Since $\theta_{true} = 0$, it is clear that $x(\theta_{true}) = -1$ is the minimizer. For $\theta \neq 0$, the agent's problem is given by $\min\{\min_x\{x : \ x \in [-1, 1], x \neq 0\}, \ 0 + \theta * (-1)\} = \min\{-1, -\theta\}$. Then, we deduce that the agent's optimal solution will satisfy the following:

- When $\theta < 0$, agent's optimal solution is $x(\theta) = -1$;
- When $\theta > 0$, $x(\theta) = -1$ if $\theta < 1$, and $x(\theta) = 0$ if $\theta \ge 1$ (note that in the case of alternative optima, we assume that the solver will break ties by selecting the solution with smaller norm);
- When $\theta = 0$, $x(\theta) = -1$.

To summarize, in the given domain Θ, when $\theta \in [-3, 1)$, $x(\theta) = -1$ and $c(x(\theta)) = 0$; when $\theta \in [1, 3]$, $x(\theta) = 0$ and $c(x(\theta)) = -1$.

We next show that implicit OL based on ℓ^{pre} with a solution oracle may lead to an unbounded regret. Suppose we choose $\eta_t = \frac{1}{t}$ for all t. Then, at time step t, based on the implicit OL based on ℓ^{pre}, we update θ_{t+1} by solving the following optimization problem: in this example, $\ell_t^{pre}(\theta) = \|x(\theta_{true}) - x(\theta)\|^2 = (-1 - x(\theta))^2$, hence

$$\theta_{t+1} = \arg\min_{\theta \in [-3,3]} \frac{1}{2}(\theta - \theta_t)^2 + \frac{1}{t}(-1 - x(\theta))^2.$$

If we initialize $\theta_1 = 3$, then the above update will generate $\theta_2 = \arg\min_{\theta \in [-3,3]} \frac{1}{2}(\theta - 3)^2 + \frac{1}{t}(-1 - x(\theta))^2$. To decide the optimal solution, we need to compare three scenarios: when $\theta = \theta_1$, the objective value is

$0 + \frac{1}{t}(-1 - x(\theta_1))^2 = \frac{1}{t}(-1-0)^2 = \frac{1}{t}$; when $\theta < 1$, the objective value is $\frac{1}{2}(\theta - \theta_1)^2 + \frac{1}{t}(-1 - x(\theta))^2 = \frac{1}{2}(\theta - 3)^2 + 0 \geq \frac{1}{2}(1-3)^2 = 2 > \frac{1}{t}$ (where we used $x(\theta) = -1$ for $\theta \leq 1$); when $1 \leq \theta < \theta_1$, the objective value is $\frac{1}{2}(\theta - \theta_1)^2 + \frac{1}{t}(-1 - x(\theta))^2 = \frac{1}{2}(\theta - 3)^2 + \frac{1}{t}(-1-0)^2 > \frac{1}{t}$. Therefore, we have $\theta_2 = \theta_1$, and by the same derivation, later iterations will always stay at the same estimate $\theta_t = \theta_1$. This means the implicit OL algorithm will generate $\theta_t = 3$ for all t, and each iteration the learner incurs prediction loss as $\ell_t^{pre}(\theta_t) = \|x(\theta_{true}) - x(\theta_t)\|^2 = \|x(0) - x(3)\|^2 = 1$. Therefore, the associated regret with respect to ℓ^{pre} is unbounded as $T \to \infty$:

$$R_T(\{\ell_t^{pre}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) = \sum_{t\in[T]} \ell_t^{pre}(\theta_t) - \sum_{t\in[T]} \ell_t^{pre}(\theta_{true}) = T.$$

We note that this example does not invalidate the regret convergence in Theorem 3. With the contrived definition of $c(x)$, the loss function ℓ^{pre} does not satisfy the Lipschitz continuity assumption needed for regret convergence guarantees given in Theorem 3. To be more specific, $\ell_t^{pre}(\theta) = (x(\theta_{true}) - x(\theta))^2$: consider $\epsilon > 0$, $\theta_1 = 1$, $\theta_2 = 1 + \epsilon > 1$, we conclude $\ell_t^{pre}(\theta_1) = (-1-(-1))^2 = 0$ and $\ell_t^{pre}(\theta_2) = (-1-0)^2 = 1$. As $\epsilon \to 0$, there is no finite G as a valid Lipschitz constant for ℓ_t^{pre}.

We also examine the use of online Mirror Descent (MD) based on ℓ^{sim} in the same setup. Let Euclidean distance be the distance generating function in Bregman distance, then online MD simplies to projected gradient descent. We again choose $\eta_t = \frac{1}{t}$ for all t, then at time step t we update θ_{t+1} via

$$\theta_{t+1} = \text{proj}_{[-3,3]}\left[\theta_t - \frac{1}{t}\big(c(x(\theta_{true})) - c(x(\theta_t))\big)\right].$$

Suppose we initialize $\theta_1 = 3$, then $\theta_2 = \text{proj}_{[-3,3]}[\theta_1 - \frac{1}{t}(0-(-1))] = \theta_1 - \frac{1}{t} = 2$. Following similar derivations, we will update θ_t in the later iterations as:

$$\begin{aligned}
\theta_3 &= \text{proj}_{[-3,3]}[\theta_2 - \frac{1}{t}(0-(-1))] = 2 - \frac{1}{2} = \frac{3}{2} \\
\theta_4 &= \text{proj}_{[-3,3]}[\theta_3 - \frac{1}{t}(0-(-1))] = \frac{3}{2} - \frac{1}{3} = \frac{7}{6} \\
\theta_5 &= \text{proj}_{[-3,3]}[\theta_4 - \frac{1}{t}(0-(-1))] = \frac{7}{6} - \frac{1}{4} = \frac{11}{12} \text{ (Note: } c(x(\theta_5)) = 0) \\
\theta_6 &= \text{proj}_{[-3,3]}[\theta_5 - \frac{1}{t}(0-0)] = \frac{11}{12}.
\end{aligned}$$

It is clear that all later iterations will stay at the same estimate $\theta_t = \frac{11}{12}$ for $t \geq 5$. By definition, $\ell_t^{sim}(\theta_t) = (\theta_t - \theta_{true})\left[c(x(\theta_{true})) - c(x(\theta_t))\right] = \theta_t(0-(-1)) = \theta_t$ for $t = 1, \ldots, 4$, and $\ell_t^{sim}(\theta_t) = \theta_t(0-0) = 0$ for $t \geq 5$. Therefore, the regret with respect to ℓ^{sim} becomes

$$R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) = 3 + 2 + 3/2 + 7/6.$$

As T increases, then the average regret converges to 0. Contrary to the ℓ^{pre} based implicit OL algorithm, online MD based on ℓ^{sim} leads to converging regret with respect to the loss function of choice. In addition, we point out that this does not violate the bounding relationship between regrets based on ℓ^{sim} and ℓ^{pre} in Corollary 1, because the agent's objective function in this simple example is not strongly convex.

C Convexity Status of ℓ^{pre} and ℓ^{est}

In this appendix, we examine the convexity status ℓ^{pre} and ℓ^{est} under our Assumption 1. Recall that we already establish in Lemma 1 that under Assumption 1 $\ell^{sub}(\theta)$ is convex in θ. On the other hand, ℓ^{pre} and ℓ^{est} are not guaranteed to be convex in θ even under Assumption 1 and even when agent's problem is a one dimensional optimization problem.

Example 1. Suppose $\Theta = [-1, 1]$ and $\theta_{true} = 1$, and an agent's forward problem is $\min_x\{\theta x : x \in \mathcal{X}\}$. We consider a convex domain $\mathcal{X} = [-1, 1]$. Let $x(\theta) \coloneqq \arg\min_x\{\theta x : x \in \mathcal{X}\}$, i.e., the set of optimizers of the

agent's problem for given θ. Then, we easily deduce that the agent's optimal action(s) at a given θ are: $x(\theta) = -1$ if $\theta > 0$, $x(\theta) \in \mathcal{X}$ if $\theta = 0$, and $x(\theta) = 1$ if $\theta < 0$. Specifically, this implies $x(\theta_{true}) = -1$.

Consider $\theta_1 = 1$, $\theta_2 = -1$ and $\lambda = \frac{1}{4}$, then $\tilde{\theta} = \lambda\theta_1 + (1-\lambda)\theta_2 = -\frac{1}{2}$. By the format of $x(\theta)$ in this problem and the definition of ℓ^{est}, we observe that

$$\begin{aligned}
\ell^{est}(\theta_1) &= \theta_{true}(x(1) - x(\theta_{true})) = 0,\\
\ell^{est}(\theta_2) &= \theta_{true}(x(-1) - x(\theta_{true})) = 2,\\
\ell^{est}(\tilde{\theta}) &= \theta_{true}(x(-1/2) - x(\theta_{true})) = 2.
\end{aligned}$$

Therefore, we deduce $\ell^{est}(\tilde{\theta}) > \lambda\ell^{est}(\theta_1) + (1-\lambda)\ell^{est}(\theta_2)$ that shows that ℓ^{est} is not a convex function of θ. Similarly, in the case of ℓ^{pre}, we arrive at

$$\begin{aligned}
\ell^{pre}(\theta_1) &= (x(1) - x(\theta_{true}))^2 = 0,\\
\ell^{pre}(\theta_2) &= (x(-1) - x(\theta_{true}))^2 = 4,\\
\ell^{pre}(\tilde{\theta}) &= (x(-1/2) - x(\theta_{true}))^2 = 4.
\end{aligned}$$

Similarly, we arrive at $\ell^{pre}(\tilde{\theta}) > \lambda\ell^{pre}(\theta_1) + (1-\lambda)\ell^{pre}(\theta_2)$ and hence conclude ℓ^{pre} is not convex. □

Note that the nonconvexity of ℓ^{est} and ℓ^{pre} established in this example remains the same even if we switch to an integer domain of $\mathcal{X} = \{-1, 1\}$.

D Proofs

D.1 Proof of Lemma 1

By definition of $\ell^{sub}(\theta)$, we have

$$\begin{aligned}
&\ell^{sub}(\theta, x(\theta; u_t); y_t, u_t) = f(y_t; \theta, u_t) - f(x(\theta; u_t); \theta, u_t)\\
&= f_1(y_t; u_t) + f_2(\theta; u_t) + \langle \theta, c(y_t)\rangle - f_2(\theta; u_t) - \min_x\{f_1(x; u_t) + \langle \theta, c(x)\rangle :\ g(x; u_t) \le 0\}\\
&= f_1(y_t; u_t) + \langle \theta, c(y_t)\rangle - \min_x\{f_1(x; u_t) + \langle \theta, c(x)\rangle :\ g(x; u_t) \le 0\}\\
&= f_1(y_t; u_t) + \max_x\{\langle \theta, c(y_t) - c(x)\rangle - f_1(x; u_t) :\ g(x; u_t) \le 0\}.
\end{aligned}$$

Here, the second equation follows from Assumption 1 and the remaining equations are simply cancellation and re-arrangements of the terms. Thus, under Assumption 1, $\ell^{sub}(\theta)$ is a point-wise maximum of affine functions of θ and hence it is a convex function of θ. ■

D.2 Proof of Lemma 2

When Assumption 1 holds, based on the given form of f, $\ell_t^{sim}(\theta)$ simplifies to a function linear in θ, hence is convex with respect to θ. ■

D.3 Proof of Proposition 1

We break down the proof of Proposition 1 into several intermediate results.

We first observe some important properties of ℓ_t^{sim} and its connection with $\ell_t^{sub}, \ell_t^{est}$. These properties play a key role in the development of our regret guarantees.

Lemma 3. *Suppose Assumption 1 holds. Then, for $t \in [T]$ and any signal u_t, we have*

(a) $\ell_t^{sim}(\theta_{true}) = 0$;

(b) $\ell_t^{sim}(\theta) + \langle \theta - \theta_{true}, c(x(\theta_t; u_t)) - c(x(\theta; u_t))\rangle = \ell_t^{sub}(\theta) + \ell_t^{est}(\theta)$ *for all* θ;

(c) $\ell_t^{sim}(\theta_t) = \ell_t^{sub}(\theta_t) + \ell_t^{est}(\theta_t)$ *for all* t.

Proof. Part (a) is evident from the definition of ℓ^{sim}. Part (b) follows from evaluating these loss functions at a given θ under Assumption 1:

$$\begin{aligned}
\ell_t^{sub}(\theta) + \ell_t^{est}(\theta) &= (f_1(y_t) + f_2(\theta) + \langle \theta, c(y_t)\rangle - f_1(x(\theta;u_t)) - f_2(\theta) - \langle \theta, c(x(\theta;u_t))\rangle) \\
&\quad + (f_1(x(\theta;u_t)) + f_2(\theta_{true}) + \langle \theta_{true}, c(x(\theta;u_t))\rangle - f_1(y_t) - f_2(\theta_{true}) - \langle \theta_{true}, c(y_t)\rangle) \\
&= \langle \theta, c(y_t) - c(x(\theta;u_t))\rangle + \langle \theta_{true}, c(x(\theta;u_t)) - c(y_t)\rangle \\
&= \langle \theta - \theta_{true}, c(y_t) - c(x(\theta;u_t))\rangle \\
&= \ell_t^{sim}(\theta) + \langle \theta - \theta_{true}, c(x(\theta_t;u_t)) - c(x(\theta;u_t))\rangle
\end{aligned}$$

Finally, part (c) follows from the fact that when we replace θ with θ_t in (b), the term representing the difference between $\ell_t^{sim}(\theta)$ and $\ell_t^{sub}(\theta) + \ell_t^{est}(\theta)$ is equal to 0. ■ ■

We next state a simple observation on the properties of the loss functions.

Observation 1. *For every* t*, we have*

(a) $\ell_t^{pre}(\theta)$*, is a nonnegative function of* θ*,*

(b) $\ell_t^{sub}(\theta)$ *and* $\ell_t^{est}(\theta)$ *are nonnegative functions of* θ *whenever there is no noise, i.e.,* $y_t = x(\theta_{true}; u_t)$*.*

Proof. In part (a), the non-negativity of ℓ_t^{pre} is obvious from its squared-norm definition. In part (b), $\ell_t^{sub}(\theta)$ is nonnegative because the objective function value of (2) at a feasible solution $y_t = x(\theta_{true}; u_t)$ is no smaller than the optimal objective value at an optimal solution $x(\theta; u_t)$. By a similar argument, $\ell_t^{est}(\theta)$ is nonnegative because $y_t = x(\theta_{true}; u_t)$ is an optimal solution and $x(\theta; u_t)$ is a feasible solution to the forward problem (1). ■

Observation 1 leads to the following result that is instrumental in simplifying the regret terms in the noiseless case.

Lemma 4. *Consider the noiseless case, i.e.,* $y_t = x(\theta_{true}, u_t)$ *for all* t*. Let* ℓ_t *denote any one of the loss functions* $\ell_t^{pre}, \ell_t^{sub}$*, or* ℓ_t^{est}*. Then,*

(a) $\arg\min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t(\theta) = \theta_{true}$*, and* $0 = \min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t(\theta)$*,*

(b) $R_T(\{\ell_t\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) = \sum_{t\in[T]} \ell_t(\theta_t)$*.*

Proof. By Observation 1, we have $\sum_{t\in[T]} \ell_t(\theta) \geq 0$ for all θ. In addition, from evaluating these loss functions at $\theta = \theta_{true}$ in the noiseless case, we have $\ell_t(\theta_{true}) = 0$ for all t, therefore $\sum_{t\in[T]} \ell_t(\theta_{true}) = 0$. This then implies $\min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t(\theta) = 0$ and the minimum is achieved at θ_{true}, proving Part (a). Part (b) follows from the definition of the regret and Part (a). ■ ■

We are now ready to prove Proposition 1.

Proof of Proposition 1. (a) Let ℓ_t represent any of the loss functions ℓ_t^{sub}, ℓ_t^{est}, and ℓ_t^{pre}. In the noiseless case, from Observation 1 we deduce that ℓ_t is a nonnegative function of θ. Then, from Lemma 4(b), we conclude that all of the corresponding regret terms, i.e., $R_T(\{\ell_t^{sub}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$, $R_T(\{\ell_t^{est}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$ and $R_T(\{\ell_t^{pre}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$ are nonnegative.

(b) From the definition of regret, Lemma 3 and Lemma 4, we have

$$\begin{aligned}
&R_T(\{\ell_t^{sub}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) + R_T(\{\ell_t^{est}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) \\
&= \left(\sum_{t\in[T]} \ell_t^{sub}(\theta_t) - 0\right) + \left(\sum_{t\in[T]} \ell_t^{est}(\theta_t) - 0\right) = \sum_{t\in[T]} \ell_t^{sim}(\theta_t)
\end{aligned}$$

where the first equation follows from Lemma 4(a), and the second equation follows from Lemma 3(c).

(c) By definition of regret term $R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$, we have

$$\begin{aligned} R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) &= \sum_{t\in[T]} \ell_t^{sim}(\theta_t) - \min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t^{sim}(\theta) \\ &\geq \sum_{t\in[T]} \ell_t^{sim}(\theta_t) - \sum_{t\in[T]} \ell_t^{sim}(\theta_{true}) \\ &= \sum_{t\in[T]} \ell_t^{sim}(\theta_t) \\ &= R_T(\{\ell_t^{sub}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) + R_T(\{\ell_t^{est}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}), \end{aligned}$$

where the inequality follows from $\sum_{t\in[T]} \ell_t^{sim}(\theta_{true}) \geq \min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t^{sim}(\theta)$, the second equation follows from Lemma 3(a), and the last equation follows from Part (b). ■

■

D.4 Proof of Corollary 1

It was shown in (Mohajerin Esfahani et al. 2018, Proposition 2.5) that when f is strongly convex in x with parameter γ, we have $\ell_t^{sub}(\theta) \geq \frac{\gamma}{2}\ell_t^{pre}(\theta)$ for all t and for all $\theta \in \Theta$. Then, using Lemma 4, we deduce $R_T(\{\ell_t^{sub}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) \geq \frac{\gamma}{2} R_T(\{\ell_t^{pre}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$. ■

E Formulations for the Solution Oracles Used in the Implicit OL Algorithms

In this section, we give the solution oracles used in implicit OL algorithms based on ℓ^{sim} and ℓ^{pre} for two forms of agent's utility functions corresponding to the ones used in our numerical experiments. We include ℓ^{pre}-based implicit OL in our discussion for the sake of comparison between ℓ^{sim}-based OL framework and the previous work Dong et al. (2018a). In our computational study, we implemented the following solution oracles when they can be readily solved by standard optimization software.

E.1 Solution Oracle for ℓ^{sim}-based Implicit OL Algorithm

Suppose that the squared Euclidean norm is used as the distance generating function in the implicit OL algorithm with the solution oracle. Recall from Definition 1 that ℓ^{sim} has the following form under Assumption 1:

$$\ell^{sim}(\theta; x(\theta_t; u_t), y_t, u_t) := \langle \theta, c(y_t) - c(x(\theta_t; u_t))\rangle + \langle \theta_{true}, c(x(\theta_t; u_t)) - c(y_t)\rangle.$$

Since the constant term in $\ell^{sim}(\theta)$ has no impact when $\ell_t^{sim}(\theta)$ is used in the objective function of an optimization problem, it can be ignored in the solution oracle formulation. Then, we deduce that the solution oracle for the ℓ^{sim}-based implicit OL algorithm updates θ_{t+1} as

$$\theta_{t+1} = \arg\min_{\theta\in\Theta} \frac{1}{2}\|\theta - \theta_t\|^2 + \eta_t\langle \theta, c(y_t) - c(x(\theta_t; u_t))\rangle.$$

In particular, when the agent's problem has the form (10) we have $f(x; \theta, u) = \frac{1}{2}x^\top Px - \langle \theta, x\rangle$, i.e., $c(x) = -x$. Thus, in this case, the solution oracle for the ℓ^{sim}-based implicit OL algorithm updates θ_{t+1} as

$$\theta_{t+1} = \arg\min_{\theta\in\Theta} \frac{1}{2}\|\theta - \theta_t\|^2 + \eta_t\langle \theta, -y_t + x(\theta_t; u_t)\rangle.$$

In the case of CES utility function, i.e., when the agent's problem has the form (11), we have $f(x;\theta,u) = \sum_{i\in[n]}(\theta)_i x_i^2$, and in this case the solution oracle for the ℓ^{sim}-based implicit OL algorithm updates θ_{t+1} as

$$\theta_{t+1} = \operatorname*{arg\,min}_{\theta\in\Theta} \frac{1}{2}\|\theta-\theta_t\|^2 + \eta_t \sum_{i\in[n]} \theta_i \left((y_t)_i^2 - x(\theta_t;u_t)_i^2\right).$$

E.2 Solution Oracle for ℓ^{pre}-based Implicit OL Algorithm

Suppose that the squared Euclidean norm is used as the distance generating function in the implicit OL algorithm with the solution oracle. Then, the solution oracle for the ℓ^{pre}-based implicit OL algorithm updates θ_{t+1} by solving the following bilevel program:

$$\theta_{t+1} = \operatorname*{arg\,min}_{\theta\in\Theta} \frac{1}{2}\|\theta-\theta_t\|^2 + \eta_t \|y_t - x(\theta;u_t)\|^2,$$

where

$$x(\theta;u_t) \in \operatorname*{arg\,min}_x \{f(x;\theta,u_t):\ g(x;u_t)\le 0,\ x\in\mathcal{X}\}.$$

Recall that when the agent's problem has the form (10) with a continuous polytope domain, i.e., $\mathcal{X}(u_t) = \mathcal{X}^{cp}(A_t,c_t)$, we have

$$x(\theta;u_t) \coloneqq \operatorname*{arg\,max}_x \left\{-\frac{1}{2}x^\top P x + \langle\theta,x\rangle:\ A_t x\le c_t,\ x\in\mathbb{R}^n_+\right\},$$

where $P\in\mathbb{S}^n_{++}$ is a fixed positive definite matrix known by both the learner and the agent. Using the KKT optimality conditions for the inner problem, and then introducing binary variables to linearize the resulting nonlinear relations, it is possible to reformulate this bilevel problem into a single level optimization problem with binary variables. In particular, in this case, following these outlined steps, Dong et al. (2018a) proposed the following reformulation of this bilevel problem into a single level MISOCP:

$$\begin{aligned}
\theta_{t+1} = \operatorname*{arg\,min}_{\theta,x,w,v,y,z}\quad & \frac{1}{2}\|\theta-\theta_t\|^2 + \eta_t\|y_t - x\|^2 \\
\text{s.t.}\quad & A_t x\le c_t,\ x\in\mathbb{R}^n_+ \\
& w_i \le M y_i,\ i\in[n] \\
& -x_i \ge -M(1-y_i),\ i\in[n] \\
& v_j \le M z_j,\ j\in[m] \\
& (A_t)_j^\top x - (c_t)_j \ge -M(1-z_j),\ j\in[m] \\
& Px - \theta + A_t^\top v - w = 0 \\
& v\in\mathbb{R}^m_+, w\in\mathbb{R}^n_+,\ y\in\{0,1\}^n,\ z\in\{0,1\}^m \\
& \theta\in\Theta.
\end{aligned}$$

Here, M is the so-called big-M constant. The variables $v\in\mathbb{R}^m_+, w\in\mathbb{R}^n_+$ are the variables corresponding to the Lagrangian multipliers, the binary variables $y_i\in\{0,1\}$ for all $i\in[n]$ are used to linearize the KKT condition $w_i x_i = 0$, and $z_j\in\{0,1\}$ for all $j\in[m]$ are introduced to linearize the KKT relation $v_j((A_t)_j^\top x - (c_t)_j) = 0$. Therefore, the big-$M$ constants must be selected so that they upper bound the components in the bilinear expressions, e.g., x_i and w_i for the complementarity constraint $w_i x_i = 0$ as well as $(A_t)_j^\top x - (c_t)_j$ and v_j for the constraint $v_j((A_t)_j^\top x - (c_t)_j) = 0$. Because in our instances the agent's domain for x is bounded, we can easily obtain bounds on x_i and $(A_t)_j^\top x - (c_t)_j$ terms. It is also possible to derive an upper bound for the Lagrange multipliers under a Slater condition assumption on the primal problem. Nevertheless, it is well known that using big-M formulations significantly degrade the optimization solver performance, and instead it is encouraged in Gurobi solver that such big-M constraints are encoded as indicator constraints, which is a form of logical constraints supported by Gurobi. In our experiments, we follow this approach and use

the indicator constraint feature of the Gurobi solver. Note that this alternative implementation is possible because the big-M constraints essentially represent a complementarity type logical condition.

Note that the continuous knapsack domain $\mathcal{X}^{ck}(p_t, b_t)$ is a special case of the continuous polytope domain $\mathcal{X}^{cp}(A_t, c_t)$, and thus the same reformulation also holds in that case.

Finally note that when the agent's problem has the form (11) with an equally constrained knapsack domain, i.e., $\mathcal{X}(u_t) = \mathcal{X}^{eck}(p_t, b_t)$, we have

$$x(\theta; u_t) \coloneqq \arg\min_x \left\{ \sum_{i\in[n]} \theta_i x_i^2 : \; p_t^\top x = b_t, \; x \in \mathbb{R}^n_+ \right\}.$$

In this case, the bilevel program corresponding to the solution oracle in the ℓ^{pre}-based implicit OL algorithm has the following single level reformulation.

$$\begin{aligned} \theta_{t+1} = \arg\min_{\theta,x,w,v,y} \quad & \frac{1}{2}\|\theta - \theta_t\|^2 + \eta_t \|y_t - x\|^2 \\ \text{s.t.} \quad & p_t x = b_t, \; x \in \mathbb{R}^n_+ \\ & w_i \le M y_i, \; i \in [n] \\ & -x_i \ge -M(1-y_i), \; i \in [n] \\ & 2\theta_i x_i + v(p_t)_i - w_i = 0, \; i \in [n] \\ & v \in \mathbb{R}, w \in \mathbb{R}^n_+, \; y \in \{0,1\}^n \\ & \theta \in \Theta. \end{aligned}$$

Unfortunately, this nonconvex mixed integer program contains the bilinear terms $\theta_i x_i$, where both x and θ are continuous variables, in a general constraint, not of a complementarity type constraint. Note that the primal domain is equality constrained continuous knapsack, and thus we can find an upper bound on x variables. Moreover, for $\theta \in \Theta$ and when Θ is bounded like the Euclidean ball or the simplex case that we focus on in this paper, we can find a bound on θ as well. However, because this bilinear term of $\theta_i x_i$ is appearing in a general constraint and not in a complementary constraint, there is no technique to reformulate this nonconvexity as linear constraints by introducing new binary variables. Hence, in this case the ℓ^{pre}-based implicit OL algorithm requires a computationally expensive general purpose nonconvex solution oracle.

F Performance for ℓ^{sim}-based Online Inverse Optimization under Imperfect Information

In this appendix, we look into the learning performance in the imperfect information setup under Assumption 1. Recall that for any of the four loss definitions, the loss value $\ell_t(\theta)$ can be split into a component measuring the difference between $x(\theta; u_t)$ and $x(\theta_{true}; u_t)$, and another one reflecting the noise shifting $x(\theta_{true}; u_t)$ to y_t. For instance, we can write $\ell^{pre}(\theta_t) = \|y_t - x_t^* + x_t^* - x_t\|^2 = \|y_t - x_t^*\|^2 + \|x_t^* - x_t\|^2 + 2\langle y_t - x_t^*, x_t^* - x_t\rangle$. Moreover, this means $\ell_t(\theta_{true}) = \ell(\theta_{true}, x(\theta_{true}; u_t); y_t, u_t)$ can be viewed as an imperfect information loss.

In Section 6.4, we consider a different collection of performance measures, consisting of the average regret with respect to ℓ^{sim}, the difference between the average loss incurred from $\{\theta_t, x(\theta_t; u_t)\}$ and the average imperfect information loss from $\{\theta_{true}, x(\theta_{true}; u_t)\}$ with respect to ℓ^{sub} and ℓ^{est}, and the average squared norm distance between $\{x(\theta_t; u_t)\}$ and $\{x(\theta_{true}; u_t)\}$. We next show that the average regret with respect to ℓ^{sim} can be used to bound the other three loss-based measures.

For notation ease, we denote $x_t \coloneqq x(\theta_t; u_t)$ and $x_t^* \coloneqq x(\theta_{true}; u_t)$ for all t.

Proposition 2. *Suppose Assumption 1 holds and the observations contain noise, namely, in each time step*

t, $y_t = x(\theta_{true}; u_t) + \epsilon_t$ *for some nonzero* ϵ_t. *For any sequence* $\{\theta_t\}_{t\in[T]}$, *we have*

$$
\begin{aligned}
(a)\ & \sum_{t\in[T]} \ell_t^{sub}(\theta_t) - \sum_{t\in[T]} \ell_t^{sub}(\theta_{true}) \le R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}),\\
(b)\ & \sum_{t\in[T]} \ell_t^{est}(\theta_t) - \sum_{t\in[T]} \ell_t^{est}(\theta_{true}) = \sum_{t\in[T]} f(x_t; \theta_{true}, u_t) - f(x_t^*; \theta_{true}, u_t)\\
& \le R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) + \sum_{t\in T} \langle \theta_{true} - \theta_t, c(y_t) - c(x_t^*)\rangle,
\end{aligned}
$$

Proof. (a) By definition of ℓ^{sub}, we have

$$
\begin{aligned}
\sum_{t\in[T]} \ell_t^{sub}(\theta_t) - \sum_{t\in[T]} \ell_t^{sub}(\theta_{true}) &= \sum_{t\in[T]} f(y_t; \theta_t, u_t) - f(x_t; \theta_t, u_t) - f(y_t; \theta_{true}, u_t) + f(x_t^*; \theta_{true}, u_t)\\
&= \sum_{t\in[T]} \langle \theta_t, c(y_t) - c(x_t)\rangle - f_1(x_t) - \langle \theta_{true}, c(y_t) - c(x_t^*)\rangle + f_1(x_t^*)\\
&= \sum_{t\in[T]} \langle \theta_t - \theta_{true}, c(y_t) - c(x_t)\rangle + \sum_{t\in[T]} f(x_t^*; \theta_{true}, u_t) - f(x_t; \theta_{true}, u_t)\\
&\le \sum_{t\in[T]} \ell_t^{sim}(\theta_t) - \min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t^{sim}(\theta) + \min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t^{sim}(\theta)\\
&\le R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) + \sum_{t\in[T]} \ell_t^{sim}(\theta_{true}) = R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}),
\end{aligned}
$$

where the initial three equations follow from definition and simple algebra, the first inequality follows from the fact that x_t^* minimizes f given θ_{true}, u_t, and the last equation is due to $\ell_t^{sim}(\theta_{true}) = 0$ for all t.

(b) The equation is a direct consequence of the loss definition.

$$
\sum_{t\in[T]} \ell_t^{est}(\theta_t) - \sum_{t\in[T]} \ell_t^{est}(\theta_{true}) = \sum_{t\in[T]} f(x_t; \theta_{true}, u_t) - f(y_t; \theta_{true}, u_t) - f(x_t^*; \theta_{true}, u_t) + f(y_t; \theta_{true}, u_t) = \sum_{t\in[T]} f(x_t; \theta_{tru}
$$

To derive the inequality, we observe that Lemma 3 still holds under imperfect information, hence we can apply Lemma 3 (b) and (c) to rewrite the loss difference.

$$
\begin{aligned}
\sum_{t\in[T]} \ell_t^{est}(\theta_t) - \sum_{t\in[T]} \ell_t^{est}(\theta_{true}) &= \sum_{t\in[T]} \ell_t^{sim}(\theta_t) - \sum_{t\in[T]} \ell_t^{sub}(\theta_t) + \sum_{t\in[T]} \ell_t^{sub}(\theta_{true})\\
&= \sum_{t\in[T]} \ell_t^{sim}(\theta_t) + \sum_{t\in[T]} \left(\langle \theta_{true} - \theta_t, c(y_t) - c(x_t^*)\rangle + f_1(x_t) - f_1(x_t^*) + \langle \theta_t, c(x_t) - c(x_t^*)\rangle\right)\\
&= \sum_{t\in[T]} \ell_t^{sim}(\theta_t) + \sum_{t\in[T]} \langle \theta_{true} - \theta_t, c(y_t) - c(x_t^*)\rangle + \sum_{t\in[T]} f(x_t; \theta_t, u_t) - f(x_t^*; \theta_t, u_t)\\
&\le R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) + \sum_{t\in[T]} \langle \theta_{true} - \theta_t, c(y_t) - c(x_t^*)\rangle,
\end{aligned}
$$

where the equations follow from definition and algebra, then the inequality uses the fact that x_t minimizes f given θ_t, u_t. ■

■

Under imperfect information, (Mohajerin Esfahani et al. 2018, Proposition 2.5) remains valid for a strongly convex f, namely, $\ell_t^{sub}(\theta) \ge \frac{\gamma}{2}\ell_t^{pre}(\theta)$ for all t and for all $\theta \in \Theta$, with γ being the strong convexity parameter of f. Therefore, we can further derive bounds with respect to $\{\ell_t^{pre}\}_{t\in[T]}$.

Corollary 2. *Suppose Assumption 1 holds and the observations $\{y_t\}_{t\in[T]}$ contain noises. Suppose further that f is a strongly convex function of x for every θ with a constant $\gamma > 0$, i.e., $f(x;\theta,u) - f(y;\theta,u) \geq \langle s_y, x-y\rangle + \frac{\gamma}{2}\|x-y\|^2$, where s_y is a subgradient of $f(y;\theta,u)$ with respect to y. Then, for any sequence $\{\theta_t\}_{t\in[T]} \subseteq \Theta$ we have:*

$$(a)\ \frac{\gamma}{2}\left(\sum_{t\in[T]}\ell_t^{pre}(\theta_t) - \sum_{t\in[T]}\ell_t^{pre}(\theta_{true})\right) \leq R_T(\{\ell_t^{sim}\}_{t\in[T]},\{\theta_t\}_{t\in[T]}) - \gamma\sum_{t\in[T]}\|x_t^* - y_t\|^2 - \sum_{t\in[T]}\langle s_{y_t}, x_t^* - y_t\rangle,$$

$$(b)\ \frac{\gamma}{2}\sum_{t\in[T]}\|x_t - x_t^*\|^2 \leq R_T(\{\ell_t^{sim}\}_{t\in[T]},\{\theta_t\}_{t\in[T]}) + \sum_{t\in T}\langle\theta_{true}-\theta_t, c(y_t)-c(x_t^*)\rangle.$$

Proof. The strong convexity of f has a few implications that we will use to derive the desirable statements.

$$f(y_t;\theta_t,u_t) - f(x_t;\theta_t,u_t) \geq \langle s_{x_t}, y_t - x_t\rangle + \frac{\gamma}{2}\|x_t - y_t\|^2 \geq \frac{\gamma}{2}\|x_t - y_t\|^2 \tag{12}$$

$$f(x_t^*;\theta_{true},u_t) - f(y_t;\theta_{true},u_t) \geq \langle s_{y_t}, x_t^* - y_t\rangle + \frac{\gamma}{2}\|x_t^* - y_t\|^2 \tag{13}$$

$$f(x_t;\theta_{true},u_t) - f(x_t^*;\theta_{true},u_t) \geq \langle s_{x_t^*}, x_t - x_t^*\rangle + \frac{\gamma}{2}\|x_t - x_t^*\|^2 \geq \frac{\gamma}{2}\|x_t - x_t^*\|^2. \tag{14}$$

(a) We apply (12) and (13) to bound the loss gap.

$$\begin{aligned}
&\frac{\gamma}{2}\left(\sum_{t\in[T]}\ell_t^{pre}(\theta_t) - \sum_{t\in[T]}\ell_t^{pre}(\theta_{true})\right) = \frac{\gamma}{2}\sum_{t\in[T]}\|x_t - y_t\|^2 - \frac{\gamma}{2}\sum_{t\in[T]}\|x_t^* - y_t\|^2\\
&\leq \sum_{t\in[T]} f(y_t;\theta_t,u_t) - f(x_t;\theta_t,u_t) - \frac{\gamma}{2}\sum_{t\in[T]}\|x_t^* - y_t\|^2 = \sum_{t\in[T]}\ell_t^{sub}(\theta_t) - \frac{\gamma}{2}\sum_{t\in[T]}\|x_t^* - y_t\|^2\\
&\leq R_T(\{\ell_t^{sim}\}_{t\in[T]},\{\theta_t\}_{t\in[T]}) + \sum_{t\in[T]}\ell_t^{sub}(\theta_{true}) - \frac{\gamma}{2}\sum_{t\in[T]}\|x_t^* - y_t\|^2\\
&= R_T(\{\ell_t^{sim}\}_{t\in[T]},\{\theta_t\}_{t\in[T]}) + \sum_{t\in[T]}\left(f(y_t;\theta_{true},u_t) - f(x_t^*;\theta_{true},u_t)\right) - \frac{\gamma}{2}\sum_{t\in[T]}\|x_t^* - y_t\|^2\\
&\leq R_T(\{\ell_t^{sim}\}_{t\in[T]},\{\theta_t\}_{t\in[T]}) - \gamma\sum_{t\in[T]}\|x_t^* - y_t\|^2 - \langle s_{y_t}, x_t^* - y_t\rangle,
\end{aligned}$$

where the first inequality uses (12), the second inequality follows from Proposition 2(b), and the last inequality uses (13).

(b) We first use (14) to bound the squared distance, then apply Proposition 2 (b).

$$\begin{aligned}
&\frac{\gamma}{2}\sum_{t\in[T]}\|x_t - x_t^*\|^2 \leq \sum_{t\in T} f(x_t;\theta_{true},u_t) - f(x_t^*;\theta_{true},u_t)\\
&\leq R_T(\{\ell_t^{sim}\}_{t\in[T]},\{\theta_t\}_{t\in[T]}) + \sum_{t\in T}\langle\theta_{true}-\theta_t, c(y_t)-c(x_t^*)\rangle.
\end{aligned}$$

■

■

In the imperfect information case, a sublinear bound on $R_T(\{\ell_t^{sim}\}_{t\in[T]},\{\theta_t\}_{t\in[T]})$ has similar implications as in the perfect information, that is, it indicates a vanishing gap between the average ℓ^{sim} incurred by estimates $\{\theta_t\}$ and by the offline optimal estimate. For the other three loss functions, we focus on the difference between the total losses from $\{\theta_t\}$ and from θ_{true}, namely the total imperfect information losses. By Proposition 2(a), with a sublinear ℓ^{sim}-based regret bound, the difference between the average ℓ^{sub} incurred by $\{\theta_t\}$ and the average imperfect information loss vanishes overtime. This means the generated estimates eventually perform at least as well as θ^{true} with respect to ℓ^{sub}.

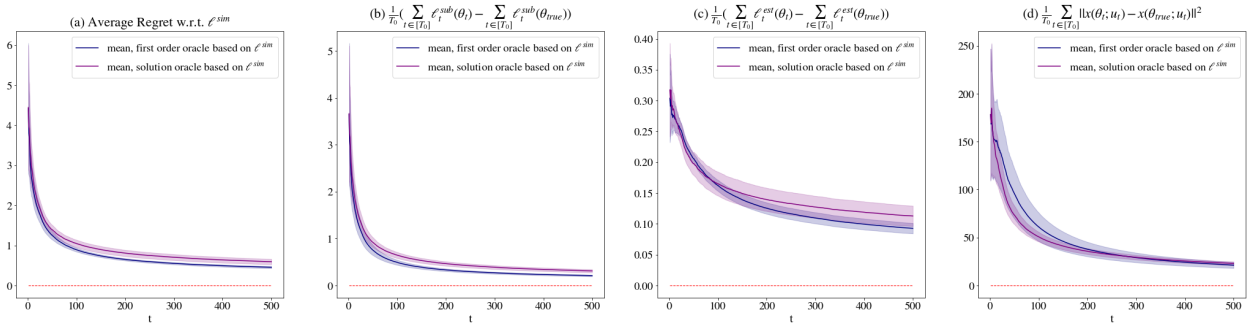


Figure 8: Learning a CES utility function under small noises: means of selected performance measures over $T = 500$ iterations for equality constrained knapsack instances with $n = 50$; the shaded region is 95% confidence interval for the means.

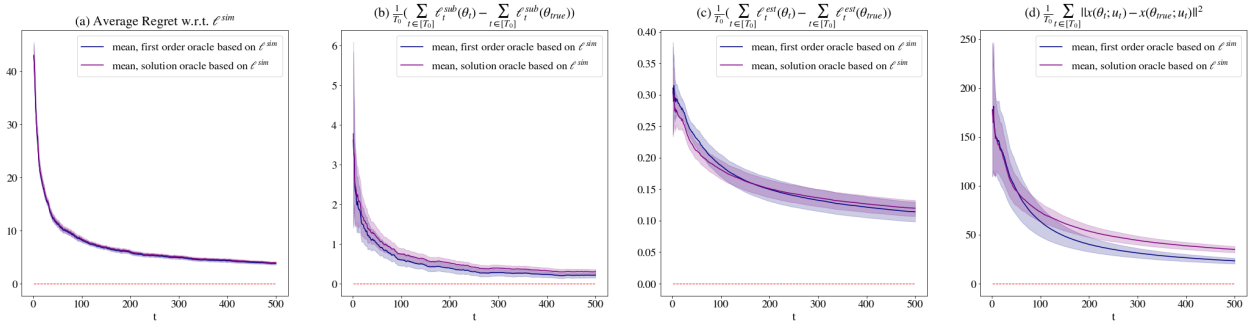


Figure 9: Learning a CES utility function under large noises: means of selected performance measures over $T = 500$ iterations for equality constrained knapsack instances with $n = 50$; the shaded region is 95% confidence interval for the means.

For ℓ^{est} and ℓ^{pre}, we have weaker results. Proposition 2(b) shows that the average loss difference between $\{\theta_t\}$ and θ_{true} decreases overtime to be below a noise-dependent limit $\frac{1}{T}\sum_{t\in T}\langle \theta_{true} - \theta_t, c(y_t) - c(x_t^*)\rangle$. Lastly, in the special case of a strongly convex forward objective f, Corollary 2(a) bounds the prediction loss gap with $R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$ and additional noise-dependent terms. Corollary 2(b) proves that the same bound on the average loss difference based on ℓ^{est} applies to the average squared norm distance between the predicted actions and the true actions.

Note that the experiment results from learning a quadratic utility function given in Section 6.4 illustrate these theoretical bounding relations. Moreover, as we next show, the results from learning a CES utility function demonstrate similar patterns.

F.1 Learning a CES Utility Function under Imperfect Information

We now provide details on imperfect information experiments in the CES setup, i.e., when the agent's problem has the form (11) with the equality constrained knapsack domain, i.e., $\mathcal{X}(u_t) = \mathcal{X}^{eck}(u_t)$, under small noises in Figure 8, and under large noises in Figure 9. Similar to the instance of learning a quadratic utility function, both ℓ^{sim}-based OL algorithms lead to converging average regrets with respect to ℓ^{sim} and decreasing average loss gaps, which are consistent with our above theoretical analysis. We note that the magnitudes of noises appears to have much smaller effects on the learning performance, as both figures show similar output.

G ℓ^{sub}-based Online Learning Algorithms

We prove in Lemma 1 that $\ell^{sub}(\theta)$ is a convex loss function under Assumption 1, so it is a potential loss function to use to develop OCO frameworks. As we next examine, ℓ^{sub} does not appear to lead to OCO methods with desirable regret convergence performances or computational tractability.

G.1 OCO with First-Order Oracle

We first consider online MD based on ℓ^{sub}. The algorithm would require a first-order oracle to obtain a subgradient of $\ell_t^{sub}(\theta)$. Since ℓ^{sub} takes the format of a point-wise maximum to a function, it is not immediately obvious to derive its subgradients. In any case, we note that, under Assumption 1, the gradient to $\ell_t^{sim}(\theta)$, $s_t = c(y_t) - c(x(\theta_t; u_t))$, is also a subgradient of $\ell_t^{sub}(\theta)$. We conclude this by comparing $\ell_t^{sub}(\theta) - \ell_t^{sub}(\theta_t)$ with $\langle s_t, \theta - \theta_t \rangle$.

$$
\begin{aligned}
&\ell_t^{sub}(\theta) - \ell_t^{sub}(\theta_t) \\
&= f_1(y_t) + \langle \theta, c(y_t) \rangle - \min_{x \in \mathcal{X}(u_t)} (f_1(x) + \langle \theta, c(x) \rangle) - \left(f_1(y_t) + \langle \theta_t, c(y_t) \rangle - \min_{x \in \mathcal{X}(u_t)} (f_1(x) + \langle \theta_t, c(x) \rangle) \right) \\
&= \langle \theta, c(y_t) \rangle - \min_{x \in \mathcal{X}(u_t)} (f_1(x) + \langle \theta, c(x) \rangle) - \langle \theta_t, c(y_t) \rangle + f_1(x(\theta_t)) + \langle \theta_t, c(x(\theta_t)) \rangle \\
&\geq \langle \theta, c(y_t) \rangle - f_1(x(\theta_t)) - \langle \theta, c(x(\theta_t)) \rangle - \langle \theta_t, c(y_t) \rangle + f_1(x(\theta_t)) + \langle \theta_t, c(x(\theta_t)) \rangle \\
&= \langle \theta - \theta_t, c(y_t) - c(x(\theta_t)) \rangle = \langle \theta - \theta_t, s_t \rangle
\end{aligned}
$$

This means online MD with respect to $\{\ell_t^{sim}(\theta)\}_{t \in [T]}$ using $s_t = c(y_t) - c(x(\theta_t; u_t))$ can be viewed as an online MD algorithm with respect to $\{\ell_t^{sub}(\theta)\}_{t \in [T]}$.

Another possible first-order OCO method based on ℓ^{sub} utilizes the saddle point structure in the minimization of ℓ^{sub}. As we note in the proof of Lemma 1, $\ell^{sub}(\theta)$ can be reformulated as follows:

$$
\ell^{sub}(\theta, x(\theta; u_t); y_t, u_t) = f_1(y_t; u_t) + \max_x \{ \langle \theta, c(y_t) - c(x) \rangle - f_1(x; u_t) : \ g(x; u_t) \leq 0 \}.
$$

From this reformulation of ℓ^{sub}, we further conclude that the inverse problem (3) based on ℓ^{sub} becomes a Saddle-Point (SP) problem:

$$
\min_{\theta \in \Theta} \max_{x \in \mathcal{X}, g(x; u_t) \leq 0} \langle \theta, c(y_t) - c(x) \rangle - f_1(x; u_t). \tag{15}
$$

Moreover, in the online learning setup as we consider, Ho-Nguyen and Kılınç-Karzan (2019) introduce the online Saddle Point problem, and show that the online Mirror Descent algorithm is applicable to the online SP problem and provides the standard $O(1/\sqrt{T})$ convergence rate bounds for the average SP gap. For the above ℓ^{sub} minimization problem, the SP gap has the following format:

$$
R_T^{SP}(\{\theta_t\}_{t \in [T]}, \{x_t\}_{t \in [T]}) = \max_{x \in \mathcal{X}, g(x; u_t) \leq 0} \sum_{t=1}^{T} (\langle \theta_t, c(y_t) - c(x) \rangle - f_1(x; u_t)) - \min_{\theta \in \Theta} \sum_{t=1}^{T} (\langle \theta, c(y_t) - c(x_t) \rangle - f_1(x_t; u_t)) \tag{16}
$$

We observe that the convergence of the average SP gap unfortunately does not imply the convergence of the average regrets with respect to ℓ^{sub}, because $R_T^{SP}(\{\theta_t\}_{t \in [T]}, \{x_t\}_{t \in [T]})$ is not guaranteed to upper bound $R_T^{sub}(\{\theta_t\}_{t \in [T]})$. As we show below, without additional assumptions, we can only derive a range containing $R_T^{SP}(\{\theta_t\}_{t \in [T]}, \{x_t\}_{t \in [T]}) - R_T^{sub}(\{\theta_t\}_{t \in [T]})$, and the range may contain 0 in its interior, which then implies that R_T^{SP} may be either larger or smaller than R_T^{sub}.

We first observe the following bounding relations on the two summation terms in $R_T^{SP}(\{\theta_t\}_{t \in [T]}, \{x_t\}_{t \in [T]})$

separately.

$$\max_{x\in\mathcal{X},g(x;u_t)\le 0}\sum_{t=1}^{T}\langle\theta_t,c(y_t)-c(x)\rangle-f_1(x;u_t)\le\sum_{t=1}^{T}\max_{x\in\mathcal{X},g(x;u_t)\le 0}\langle\theta_t,c(y_t)-c(x)\rangle-f_1(x;u_t)$$
$$=\sum_{t=1}^{T}\langle\theta_t,c(y_t)\rangle-\min_{x\in\mathcal{X},g(x;u_t)\le 0}(\langle\theta_t,c(x)\rangle+f_1(x;u_t))=\sum_{t=1}^{T}\langle\theta_t,c(y_t)-c(x(\theta_t))\rangle-f_1(x(\theta_t);u_t)\ ;$$
$$\min_{\theta\in\Theta}\sum_{t=1}^{T}\langle\theta,c(y_t)-c(x_t)\rangle-f_1(x_t;u_t)\ge\sum_{t=1}^{T}\min_{\theta\in\Theta}\langle\theta,c(y_t)-c(x_t)\rangle-f_1(x_t;u_t)$$

The inequality steps follow from the fact that the optimal value to a summation of functions as the objective must be no better than the summation of the optimal values to the individual components in the objective function. And the equality steps in bounding the maximization term follow from the definition of the forward problem (2) and substituting the notation $x(\theta_t)$ for the optimizer.

For simplicity, we focus on the special perfect information case where $x(\theta_{true};u_t)$ is the same under all u_t, namely $y_t=x(\theta_{true})$ for all t, here u_t is skipped. We use the previous bounds and the definition of R_T^{sub} to derive a upper bound on $R_T^{SP}-R_T^{sub}$:

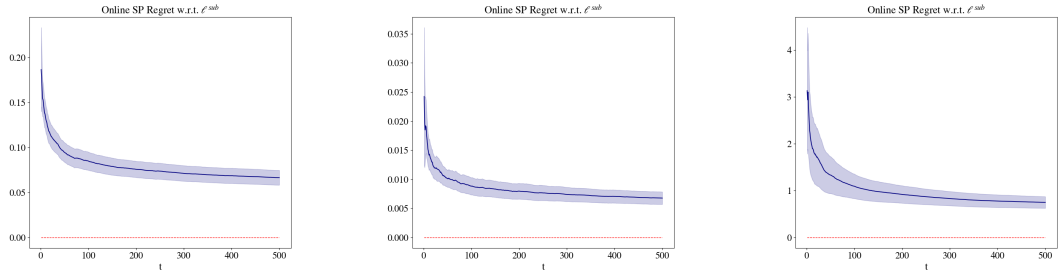
$$\begin{aligned}
&R_T^{SP}(\{\theta_t\}_{t\in[T]},\{x_t\}_{t\in[T]})-R_T^{sub}(\{\theta_t\}_{t\in[T]})\\
\le&\sum_{t=1}^{T}(\langle\theta_t,c(y_t)-c(x(\theta_t))\rangle-f_1(x(\theta_t);u_t)+f_1(y_t;u_t))-\sum_{t=1}^{T}\min_{\theta\in\Theta}(\langle\theta,c(y_t)-c(x_t)\rangle-f_1(x_t;u_t)+f_1(y_t;u_t))\\
&-\sum_{t=1}^{T}(\langle\theta_t,c(y_t)-c(x(\theta_t))\rangle-f_1(x(\theta_t);u_t)+f_1(y_t;u_t))+\min_{\theta\in\Theta}\sum_{t=1}^{T}(\langle\theta,c(y_t)-c(x(\theta))\rangle-f_1(x(\theta);u_t)+f_1(y_t;u_t))\\
=&-\sum_{t=1}^{T}\min_{\theta\in\Theta}(\langle\theta,c(y_t)-c(x_t)\rangle-f_1(x_t;u_t)+f_1(y_t;u_t))+0\\
=&\sum_{t=1}^{T}\max_{\theta\in\Theta}\left[f(\theta,x_t)-f(\theta,x(\theta_{true}))\right].
\end{aligned}$$

This final upper bound formula is non-negative because θ_{true} is a feasible solution to each maximization problem, and the corresponding differences in f are non-negative due to the optimality of $x(\theta_{true})$ in (1).

From the other direction, we pick $x(\theta_{true})$ as a feasible solution to the first maximization term in R_T^{SP} and θ^{true} as a feasible solution to the minimization term, then we have

$$\begin{aligned}
&R_T^{SP}(\{\theta_t\}_{t\in[T]},\{x_t\}_{t\in[T]})-R_T^{sub}(\{\theta_t\}_{t\in[T]})\\
\ge&\sum_{t=1}^{T}(\langle\theta_t,c(y_t)-c(x(\theta_{true}))\rangle-f_1(x(\theta_{true});u_t)+f_1(y_t;u_t))-\sum_{t=1}^{T}(\langle\theta_{true},c(y_t)-c(x_t)\rangle-f_1(x_t;u_t)+f_1(y_t;u_t))\\
&-\sum_{t=1}^{T}(\langle\theta_t,c(y_t)-c(x(\theta_t))\rangle-f_1(x(\theta_t);u_t)+f_1(y_t;u_t))+\min_{\theta\in\Theta}\sum_{t=1}^{T}(\langle\theta,c(y_t)-c(x(\theta))\rangle-f_1(x(\theta);u_t)+f_1(y_t;u_t))\\
=&\sum_{t=1}^{T}(\langle\theta_t,c(x(\theta_t))-c(x(\theta_{true}))\rangle-f_1(x(\theta_{true});u_t)+f_1(x(\theta_t);u_t))-\sum_{t=1}^{T}(\langle\theta_{true},c(y_t)-c(x_t)\rangle-f_1(x_t;u_t)+f_1(y_t;u_t))\\
=&\sum_{t=1}^{T}\left[f(\theta_t,x(\theta_t))-f(\theta_t,x(\theta_{true}))\right]-\sum_{t=1}^{T}\left[f(\theta_{true},x(\theta_{true}))-f(\theta_{true},x_t)\right].
\end{aligned}$$

The optimality of $x(\theta_t)$ and $x(\theta_{true})$ for all t indicates that both summations in the final bound formula are non-positive, but we cannot conclude the sign of their differences without additional assumptions.



(a) Learning a quadratic utility function for continuous knapsack instances.

(b) Learning a quadratic utility function for continuous polytope instances.

(c) Learning a CES utility function for equality constrained knapsack instances.

Figure 10: Means of average SP regrets over $T = 500$ iterations; the shaded region is 95% confidence interval for the means.

Combining both bounds, we conclude that R_T^{SP} is not guaranteed to be an upper bound on R_T^{sub} in general. We next test the online SP algorithm stated in Ho-Nguyen and Kılınç-Karzan (2019) on three instances used in the computational study, and the experiment results demonstrate the insufficiency of online SP with respect to ℓ^{sub} in bounding the other loss function based regrets.

G.1.1 Perfect Information Experiments: Online SP with respect to ℓ^{sub}

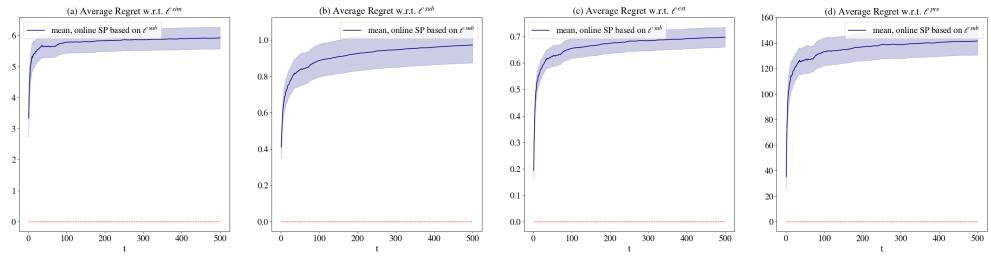
We implement the online SP algorithm for (15) for learning a quadratic utility function with both $\mathcal{X}^{ck}$ and $\mathcal{X}^{cp}$ domains in the forward problem, and for learning a CES utility with a $\mathcal{X}^{eck}$ domain. In all experiments, we run the SP algorithm for $T = 500$ iterations on the 50 random instances (same as those used for the other ℓ^{sim} and ℓ^{pre} based experiments). We report the average SP regret associated with the generated $\{\theta_t, x_t\}_{t\in[T]}$, and the average regrets of the estimates $\{\theta_t\}_{t\in[T]}$ with respect to all four loss functions respectively.

Figure 10 shows that, in all three instances, the associated average SP regret decreases at roughly the same rate. These observed trends confirm the convergence result for the online SP algorithm established in (Ho-Nguyen and Kılınç-Karzan 2019, Theorem 3). On the other hand, Figure 11 demonstrates that the estimates generated from online SP algorithm with respect to ℓ^{sub} fail to provide convergence guarantees on the average regrets with respect to the selected loss functions. In both instances of learning a quadratic utility function, we observe from Figs. 11a and 11b that all four average regrets on the estimates $\{\theta_t\}_{t\in[T]}$ are increasing and appear to converge to a positive value in some cases. In the instance of learning a CES utility function, to the contrary, Fig. 11c shows that the average regret with respect to ℓ^{sub} follows a similar convergence trend as the average SP regret. Even though the online SP algorithm appears to be sufficient to guarantee the convergence of the average regret with respect to ℓ^{sub} in this case, we observe that there is still a lack of convergence guarantees for the other regrets, in particular, R_T^{sim} and R_T^{pre} are increasing throughout all iterations.

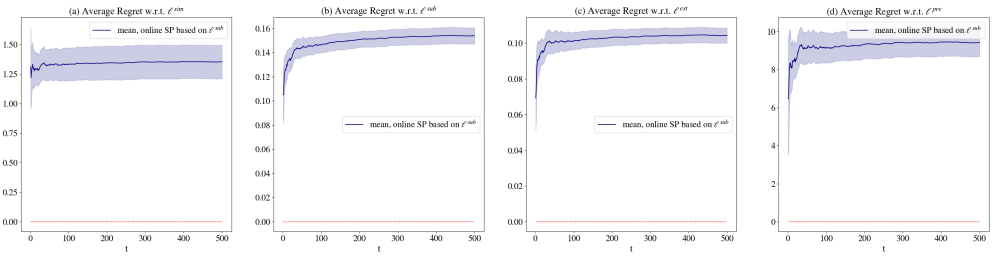
G.2 OCO with Solution Oracle

To apply the implicit OL algorithm with respect to ℓ^{sub}, we need a solution oracle for solving $\min_{\theta\in\Theta} V_\theta(\theta_t) + \eta_t \ell_t^{sub}(\theta)$. Even though $\ell_t^{sub}(\theta)$ is convex in θ, the required solution oracle may be computationally intractable, because the format of $\ell_t^{sub}(\theta)$ makes the underlying optimization model a bilevel program with both θ and x as the decision variables. Recall from Appendix E, the solution oracle for ℓ^{sim}-based implicit OL algorithm is naturally a convex program in θ with the convenient structure of ℓ^{sim}. The ℓ^{pre}-based solution oracles are bilevel programs, which may not be tractable in general, as we have discussed for the case of learning a CES utility function.

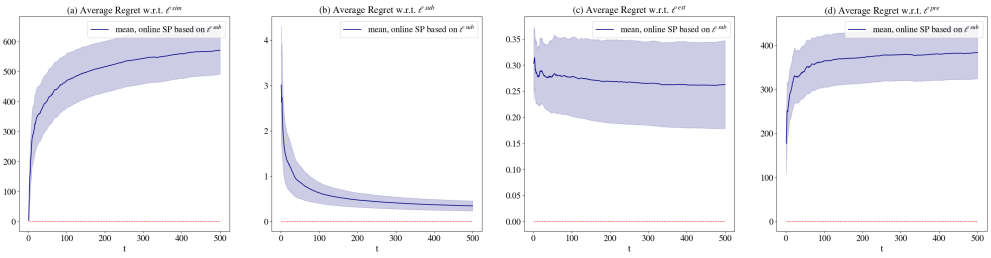
Similar computational difficulties occur for using ℓ^{sub}-based solution oracles. We again suppose that the squared Euclidean norm is used as the distance generating function in the solution oracle, then the ℓ^{sub}-based



(a) Learning a quadratic utility function for continuous knapsack instances with $n = 50$.



(b) Learning a quadratic utility function for continuous polytope instances with $n = 50$.



(c) Learning a CES utility function for equality constrained knapsack instances with $n = 50$.

Figure 11: Means of average regret with respect to different loss functions over $T = 500$ iterations; the shaded region is 95% confidence interval for the means.

implicit OL algorithm updates θ_{t+1} by solving the following bilevel program:

$$\theta_{t+1} = \arg\min_{\theta \in \Theta} \frac{1}{2} \|\theta - \theta_t\|^2 + \eta_t \left(\langle \theta, c(y_t) - c(x(\theta; u_t)) \rangle - f_1(x(\theta; u_t); u_t) \right)$$

where

$$x(\theta; u_t) \in \arg\min_x \{ f(x; \theta, u_t) : \ g(x; u_t) \leq 0, \ x \in \mathcal{X} \}.$$

To obtain a single-level reformulation, we can again apply the KKT conditions to the inner problem to represent $x(\theta; u_t)$ in the constraints. We demonstrate the following reformulations for our experiment instances to further illustrate where the computational challenges rise. First, when the agent's problem has the form (10) with a continuous polytope domain, $\mathcal{X}(u_t) = \mathcal{X}^{cp}(A_t, c_t)$, the ℓ^{sub} based solution oracle simplifies to the following single level model.

$$\begin{aligned}
\theta_{t+1} = \arg\min_{\theta, x, w, v, y, z} \quad & \frac{1}{2} \|\theta - \theta_t\|^2 + \eta_t (\langle \theta, x - y \rangle - \frac{1}{2} x^T P x) \\
\text{s.t.} \quad & A_t x \leq c_t, \ x \in \mathbb{R}^n_+ \\
& w_i \leq M y_i, \ i \in [n] \\
& -x_i \geq -M(1 - y_i), \ i \in [n] \\
& v_j \leq M z_j, \ j \in [m] \\
& (A_t)_j^\top x - (c_t)_j \geq -M(1 - z_j), \ j \in [m] \\
& Px - \theta + A_t^\top v - w = 0 \\
& v \in \mathbb{R}^m_+, w \in \mathbb{R}^n_+, \ y \in \{0,1\}^n, \ z \in \{0,1\}^m \\
& \theta \in \Theta.
\end{aligned}$$

For the other case, when the agent's problem has the form (11) with an equally constrained knapsack domain, $\mathcal{X}(u_t) = \mathcal{X}^{eck}(p_t, b_t)$, the ℓ^{sub} based solution oracle is equivalent to:

$$\begin{aligned}
\theta_{t+1} = \arg\min_{\theta, x, w, v, y} \quad & \frac{1}{2} \|\theta - \theta_t\|^2 + \eta_t (\langle \theta, x - y \rangle - \frac{1}{2} x^T P x) \\
\text{s.t.} \quad & p_t x = b_t, \ x \in \mathbb{R}^n_+ \\
& w_i \leq M y_i, \ i \in [n] \\
& -x_i \geq -M(1 - y_i), \ i \in [n] \\
& 2\theta_i x_i + v(p_t)_i - w_i = 0, \ i \in [n] \\
& v \in \mathbb{R}, w \in \mathbb{R}^n_+, \ y \in \{0,1\}^n \\
& \theta \in \Theta.
\end{aligned}$$

We note that the reformulated objective functions contain the bilinear term $\langle \theta, x \rangle$, which means the objectives are not guaranteed to be convex in general. As a result, the ℓ^{sub}-based implicit OL algorithm would require an expensive general purpose nonconvex solution oracle.