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

23 August 2020

Abstract

We study the problem of online learning (OL) from revealed preferences: a learner wishes to learn an 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. Due to the inverse optimization component, attaining or proving convexity is difficult for all of the usual loss functions in the literature. We address this challenge by designing a new loss function that is convex under relatively mild assumptions. Moreover, we establish that the regret with respect to our new loss function also bounds the regret with respect to all other usual loss functions. This then 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 eliminating technical assumptions as well as regret performance and solution time over other OL algorithms from the literature.

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 to determine product prices, or how a company should personalize recommendations to an individual customer. Therefore, the optimum actions of a company/central decision maker should align with 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 company adjusting the prices of their products multiple times and observing the customers' demand at these price levels, or a streaming platform suggesting a number of videos to a user and tracking whether the user watches or likes them. As evident in these examples, 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 their action to maximize their utility function 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.

Varian (2006) presents one of the earliest and most celebrated work from a learning point of view of revealed preferences in the economics literature. This work focuses on 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. Later on Balcan et al. (2014), Saharoy and Tulabandhula (2018), Zadimoghaddam and Roth (2012) propose efficient statistical

learning algorithms for specific classes of utility functions. 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. More recently, Amin et al. (2015), Dong et al. (2018b), Ji et al. (2018), Roth et al. (2016) generalize the query-based model to a new context in which the learner seeks to optimize her/his specific objective function instead of minimizing a loss function measuring how well the agent's utility function is learned.

Learning from revealed preferences can also be abstracted as an inverse optimization problem: given a batch or stream of optimum solutions to the agent's utility maximization problems, the learner aims to recover an unknown parameter vector θ_{true} defining the agent's utility function. In inverse optimization, typically the learning performance is evaluated with respect to a loss function. While the early literature on inverse optimization such as Ahuja and Orlin (2001), Heuberger (2004), Iyengar and Kang (2005) consider a simple non-data-driven setup, where the learner receives a single observation of the agent's action and seeks to identify an optimal estimate for θ_{true}, recent literature Aswani et al. (2018), Bärmann et al. (2017), Dong et al. (2018a), Keshavarz et al. (2011), Mohajerin Esfahani et al. (2018) examine a more realistic data-driven perspective where the learner observes the agent's actions under different data signals, and aims to estimate θ_{true} using multiple observations. Research on data-driven inverse optimization is further distinguished by whether the data is given upfront as a batch or dynamically in an online fashion.

In this paper, we study the problem of learning utility functions from dynamically revealed preferences, and thus focus on the more realistic online data-driven inverse optimization setup.

1.1 Related Literature

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.

The query-based models initiated by Balcan et al. (2014) 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) generalize the price setting problem to a Stackelberg game where a leader player optimizes a utility function in an online fashion without full access to the follower's preferences. Dong et al. (2018b) consider preference learning in strategic classification, which is an example of Stackelberg game with the leader player releasing classifiers to strategic follower players. When the leader's classifier choice problem is a convex program, Dong et al. (2018b) provide an online zeroth-order optimization algorithm for minimizing the leader's Stackelberg regret.

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 valid in practice. We will next review the data-driven inverse optimization literature that accommodates a more realistic online setup 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.

Early studies on inverse optimization examine the setting where the goal is to recover the unknown objec-

tive function of an agent's optimization problem from the given information on the agent's true optimal solution/action. A classical result in this setup due to Ahuja and Orlin (2001) establishes that the inverse optimization of linear programs (LPs) can be reformulated as an LP using duality arguments; Iyengar and Kang (2005) provide an extention of this to the conic problems. See also Heuberger (2004) for an extensive survey of inverse combinatorial optimization and Schaefer (2009) for inverse optimization of integer programs.

The classical inverse optimization view is limited in its practical applicability as its setup ignores uncertainty in the environment as it assumes that the agent's optimization problem is fixed. Motivated by recent progress on utilizing data in decision-making, a new thread of research on data-driven inverse optimization studies a flexible setup, where the learner has access to a batch or stream of data in the form of multiple observations of optimal or sub-optimal solutions 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. When noises are present, although there are a few special polynomial-time solvable classes (see Aswani et al. (2018), Mohajerin Esfahani et al. (2018)), in general such problems are shown to be NP-hard by Aswani et al. (2018).

The data-driven inverse optimization is further categorized based on whether observations are given as a batch upfront or in an online manner. Keshavarz et al. (2011) adopt a batch view and study the inverse optimization of identifying a convex objective function, which has the format of an affine combination of pre-selected basis convex functions with unknown affine weights. The learner aims to find the corresponding weights of the agent's objective function, with which the observed agent decisions satisfy the KKT conditions of the agent's optimization problem. Recently, Aswani et al. (2018) and Mohajerin Esfahani et al. (2018) also consider a batch feedback setup and study inverse optimization of general convex programs without the basis function structure. Aswani et al. (2018) choose minimizing *prediction loss* ℓ^{pre}, which measures the difference between the observed agent action and the predicted agent action through squared norm distance, as their 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 agent action, as their loss function and give a distributionally robust formulation of the inverse problem. The batch setup assumes 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 in inverse optimization study the dynamic information acquisition setup where the learner observes a stream of the agent's actions one by one in an online fashion. Bärmann et al. (2017) and Dong et al. (2018a) study and propose algorithms for online data-driven inverse optimization, where the performance of OL algorithms is measured via the *regret*, i.e., the difference between the losses incurred from online estimates 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 actions in a dynamic environment. They assume that the learner has access to an efficient linear optimization oracle over the agent's domain, which can possibly be nonconvex, and propose two specialized OL algorithms with first-order oracles that both achieve a regret bound of $O(\sqrt{T})$ with respect to the suboptimality loss ℓ^{sub} after T periods. Nevertheless, Bärmann et al. (2017) fails to provide regret guarantees in terms of other loss functions commonly studied in the data-driven inverse optimization context. Dong et al. (2018a) consider the setup where the learner's observations may be corrupted with noise, and the learner wishes to learn an unknown linear component of an agent's quadratic objective function. They utilize the implicit OL framework of Kulis and Bartlett (2010) equipped with a Mixed Integer Second Order Cone Program (MISOCP)-based solution oracle, and establish that when the prediction loss function ℓ^{pre} is convex, their OL algorithm achieves a regret bound of $O(\sqrt{T})$ with respect to the prediction loss ℓ^{pre} after T periods. However, as pointed out in Dong et al. (2018a), ensuring the convexity of ℓ^{pre} is challenging in general. Although Dong et al. (2018a) present a number of rather technical assumptions that guarantee the convexity of ℓ^{pre}, these assumptions are not only restrictive but also difficult to verify; in fact, in Dong et al. (2018a), these were shown to hold only for a very specific case of a specific convex quadratic problem class.

1.2 Contributions and Outline

In this paper, we study an online data-driven inverse optimization perspective for learning from dynamically revealed preferences. Compared to the literature on learning utility functions from revealed preferences in a statistical setup or data-driven inverse optimization setting with batch observations which both assume that all of the data is given upfront, we work with dynamically revealed preferences. In our setup, the learner monitors a sequence of data signals and observes the agent's respective rational decisions without noise over a finite time horizon of T time steps. The agent's decisions reveal his/her preferences dynamically. 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. We give a detailed description of our setup in Section 2.

We measure the accuracy of our estimates θ_t in terms of their regret with respect to a given *loss function* $\ell(\cdot)$. In Section 2.2, we study a number of usual loss functions including ℓ^{pre} and ℓ^{sub} from the inverse optimization literature. As noted in this literature, proving the convexity of these loss functions is challenging due to the inverse optimization component. Therefore, we design another loss function ℓ^{sim} that, under an assumption on the form of the agent's utility function, can be shown to be convex. Indeed, we demonstrate that this assumption holds for all of the utility functions studied in the data-driven inverse optimization literature as well as the CES and Cobb-Douglas utility functions; see e.g., Remarks 2 and 3. In Section 3, we carefully establish a number of relations on the regret bounds associated with our loss function ℓ^{sim} and the existing ones from the literature including ℓ^{sub} and ℓ^{pre}. These relations in particular demonstrate that in the noiseless case any online algorithm which results in a bounded regret with respect to our loss function ℓ^{sim} also guarantees a bounded regret with respect to the other loss functions; see Proposition 2.

We proceed to propose an OL framework based on regret minimization with respect to ℓ^{sim}. An important feature of our framework is that it not only provides a unified treatment of a variety of loss functions and problem classes from the literature, but also offers the flexibility to use different OL algorithms utilizing a variety of oracles. In particular, we establish that one can use the well-known online Mirror Descent (MD) in this setup as an OL algorithm utilizing a first-order oracle; see Section 4.1. In addition, we discuss two other OL algorithms based on a zeroth-order oracle and a solution oracle in Sections 4.2 and 4.3 respectively. 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 bound with respect to *all* of the loss functions studied in this literature in a *unified* manner. In particular, our framework equipped with online MD immediately generalizes the customized algorithms from Bärmann et al. (2017) and completely bypasses the requirement to verify the rather technical assumptions of Dong et al. (2018a) and the need to use their expensive MISOCP-based solution oracle.

We carry out numerical experiments on the practical application of a firm that wishes to learn the preferences of its customers in a changing market. We compare the performances of our ℓ^{sim}-based OL algorithms using first-order, zeroth-order, and solution oracles as well as the ℓ^{pre}-based implicit OL with a MISOCP solution oracle from Dong et al. (2018a) in terms of their regrets with respect to all of the loss functions in the literature. In all different settings we examine, we identify that our ℓ^{sim}-based OL algorithms, particularly online MD, have significant advantages in terms of both the learning performance measured with regret bounds and the computation time. This is directly in line with our theoretical results. We observe that these results are fairly robust with respect to changes in the structure of the agent's domain. Finally, we test the performance of OL algorithms in the imperfect information setup. We observe that they still perform reasonably well when the observations are noisy despite the lack of general theoretical guarantees, yet their performance gets worse as the magnitude of the noises intensifies.

All of the proofs are given in Appendix A, the details on the solution oracles are presented in Appendix B, and supplemental numerical results are provided in Appendix C and Appendix D.

1.3 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 $\mathbb{S}^n$ be the space of $n \times n$ real symmetric matrices, $\mathbb{S}^n_+$ be the set of positive semidefinite matrices in $\mathbb{S}^n$, and $\mathbb{S}^n_{++}$ be the set of symmetric positive definite $n \times n$ matrices. Given $A \in \mathbb{S}^n$, $\lambda_{\min}(A)$ and $\lambda_{\max}(A)$ denote respectively the smallest and the largest eigenvalue of A. We let $[n] \coloneqq \{1, \ldots, n\}$, and

we use $\{a_i\}_{i\in[n]}$ to represent a collection of entries, such as vectors, functions, etc., indexed with $i \in [n]$. $\text{Proj}_S(x)$ denotes the usual projection operation of $x \in \mathbb{R}^n$ onto a set $S \subseteq \mathbb{R}^n$. 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

We study an online setting for learning from dynamically revealed preferences. In this setup, we consider a finite time horizon of T time steps, where the learner monitors a sequence of 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 rational decisions. For a fixed exogenous signal u, the agent's optimal decision $x(\theta_{true}; u)$ is given by the *forward problem*:

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

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. The learner's goal is to estimate θ_{true} with desirable accuracy given $\{(u_t, y_t)\}_{t\in[T]}$ and the knowledge of a convex set Θ containing θ_{true}. A *loss function* ℓ is used to measure the performance of learner's estimates: given signal u, the learner incurs $\ell(\theta, x(\theta; u); y, u)$ as the loss for the estimate θ, where y is what the learner observes of $x(\theta_{true}; u)$ and $x(\theta; u)$ is the prediction of the agent's optimal decision if her/his true parameter were θ. The predicted action $x(\theta; u)$ is obtained from the *forward problem* by replacing θ_{true} with θ, i.e., $x(\theta; u)$ is an optimal solution to

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

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. Throughout this paper, we focus on the *perfect information* case, where the learner observes $y_t = x(\theta_{true}, u_t)$ for all t.

Given a pair of observations u, y, the learner's goal is to find θ that minimizes the loss function $\ell(\theta, x(\theta; u); y, u)$, which measures the inaccuracy of using θ instead of θ_{true}. 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

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

Note that in the above formulation, the signal u is left unspecified. In the general online setup, the learner observes a sequence of signals and the agent's responses to these signals.

2.1 Online Inverse Optimization

Our framework is based on online learning, and thus here we briefly review OL in the context of inverse optimization. Over a finite time horizon T, at each time step $t \in [T]$, the learner generates an estimate $\theta_{t+1} \in \Theta$ for the true parameter θ_{true} using information collected on the loss functions $\ell_t(\theta) \coloneqq \ell(\theta, x(\theta; u_t); y_t, u_t)$ from the previous t time steps. Then the learner receives feedback on the current loss objective $\ell_t(\theta)$. Common types of feedback include the first-order information, i.e., the gradient $\nabla\ell_t(\theta_t)$ or a subgradient from $\partial\ell_t(\theta_t)$, and the zeroth-order information $\ell_t(\theta_t)$. The goal of the online decision maker is to minimize the cumulative loss $\sum_{t\in[T]} \ell_t(\theta_t)$. The performance of an OL algorithm is often 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:

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

2.2 Loss Functions

The following are common loss functions $\ell(\theta)$ used in the context of inverse optimization; see Mohajerin Esfahani et al. (2018) for the motivation of these loss functions. Recall that f is the agent's forward objective in (1).

- 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: $\ell^{est}(\theta, x(\theta; u_t); y_t, u_t) \coloneqq f(x(\theta; u_t); \theta_{true}, u_t) - f(y_t; \theta_{true}, u_t)$.

Observe that ℓ^{pre} is a nonnegative function of θ for every u_t. In addition, we have the following structural properties of these loss functions in the noiseless case.

Observation 1. *In the noiseless case, i.e., $y_t = x(\theta_{true}; u_t)$ for all t, we have for every u_t,*

(a) ℓ^{sub} and ℓ^{est} are nonnegative functions of θ;

(b) $\ell^{pre}(\theta_{true}, x(\theta_{true}; u_t); y_t, u_t) = \ell^{sub}(\theta_{true}, x(\theta_{true}; u_t); y_t, u_t) = \ell^{est}(\theta_{true}, x(\theta_{true}; u_t); y_t, u_t) = 0$.

All of these loss functions share the common disadvantage that it is difficult to conclude whether they are convex in θ. This is due to the fact that $x(\theta; u_t)$ is the optimal solution to (2) with u set to be u_t, and a closed form expression for $x(\theta; u_t)$ as a function of θ is usually challenging to obtain. We are interested in computationally tractable loss functions that can also be used to bound the regrets associated with the classical ones $\ell^{pre}, \ell^{sub}, \ell^{est}$. To this end, we define a new loss function and examine an instrumental assumption on the agent's objective f in (1).

Definition 1. We define the *aggregate loss* as

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

□

Remark 1. By Observation 1, in the noiseless case, $\ell^{ag}(\cdot)$ is a nonnegative function of θ for every u_t. Moreover, from Observation 1(b), we deduce that $\ell^{ag}(\theta_{true}, x(\theta_{true}; u_t); y_t, u_t) = 0$. □

We will rely on a structural assumption on the agent's objective function to define a new tractable loss function.

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 $c(x) = (c_1(x), \ldots, c_p(x))$. □

Observation 2. *Suppose Assumption 1 holds, then the loss function ℓ^{ag} is simplified to*

$$\ell^{ag}(\theta, x(\theta; u_t); y_t, u_t) = \langle \theta, c(y_t) - c(x(\theta; u_t)) \rangle + \langle \theta_{true}, c(x(\theta; u_t)) - c(y_t) \rangle. \tag{5}$$

We note that convexity of ℓ^{ag} with respect to θ is still difficult to decide because of the presence of $x(\theta; u_t)$. Under Assumption 1, we next define a much simpler loss function, which replaces $x(\theta; u_t)$ with $x(\theta_t; u_t)$ in ℓ^{ag}. The term $x(\theta_t; u_t)$ can be viewed as a prediction of the agent's action at the current estimate θ_t of the true parameter θ_{true}.

Definition 2. We define the *simple loss* as

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

□

Note that the function ℓ^{sim} no longer involves the component $x(\theta; u_t)$ that is determined by the forward problem (2). Furthermore, ℓ^{sim} is linear in θ, hence is convex in θ.

Let us introduce some shorthand notations.

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

Proposition 1. *Suppose the agent's objective function f satisfies Assumption 1, then $\ell_t^{sim}(\theta)$ is convex in θ for every $t \in [T]$.*

Convexity of ℓ_t^{sim} heavily depends on Assumption 1 that requires a special structure on the function f, where the only term consisting of both θ and x is linear in θ. Although this structural assumption on f appears to be strong and restrictive, this specific form of f generalizes all problem classes studied in this literature from the OL perspective. In addition, our framework captures two important classes of utility functions that have not been addressed in the OL setup: CES (constant elasticity of substitute) function, and Cobb-Douglas function, which is a limit case of the CES function. We next demonstrate that both of these utility functions can be transformed to satisfy Assumption 1, hence leading to a convex class of loss functions $\{\ell_t^{sim}\}_{t\in[T]}$. Unfortunately, the other well-studied limit case known as the Leontief function does not fit into the same framework.

Remark 2. 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. 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 $U(x)$:

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

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

Remark 3. 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. □

We close this section by noting an important property of ℓ_t^{sim} and its connection with ℓ_t^{ag}.

Observation 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) $\ell_t^{ag}(\theta) - \ell_t^{sim}(\theta) = \langle \theta - \theta_{true}, x(\theta_t; u_t) - x(\theta; u_t) \rangle$ *for all* θ;

(c) $\ell_t^{ag}(\theta_t) - \ell_t^{sim}(\theta_t) = 0$ *for all* θ_t.

3 Regret Performance Measures for Preference Learning

We develop an OL framework, specifically an *online convex optimization* framework, for dynamic preference learning. In order to achieve this goal, we need to identify a loss function that can be shown to be convex under rather broad yet simple-to-verify assumptions. To this end, we have already introduced a loss function, i.e., ℓ^{sim}, that is convex under Assumption 1. Nevertheless, we still need to establish that the regret bounds with respect to this loss function can immediately be translated into the regret bounds associated with different loss functions of interest. In this section, we show that under perfect information ℓ_t^{sim} indeed provides regret bounds with respect to the other loss functions of interest as well. We close this section with a discussion of the main challenges in generalizing our framework to handle imperfect information.

3.1 Perfect Information

We begin with an observation instrumental in simplifying the regret terms in the noiseless case.

Lemma 1. *Consider the noiseless case, i.e., $y_t = x(\theta_{true}, u_t)$ for all t. Then,*

$$\theta_{true} = \arg\min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t^{pre}(\theta) = \arg\min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t^{sub}(\theta) = \arg\min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t^{est}(\theta) = \arg\min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t^{ag}(\theta)$$

$$\text{and} \quad 0 = \min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t^{pre}(\theta) = \min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t^{sub}(\theta) = \min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t^{est}(\theta) = \min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t^{ag}(\theta).$$

In the perfect information case, Lemma 1 implies that the regret $R_T(\{\ell_t\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) = \sum_{t\in[T]} \ell_t(\theta_t)$ for these four loss functions $\ell_t^{pre}, \ell_t^{sub}, \ell_t^{est}$ or ℓ_t^{ag}. We next establish a fundamental guarantee among the regret bounds with respect to the loss functions $\ell_t^{sim}, \ell_t^{sub}, \ell_t^{est}$ or ℓ_t^{ag}.

Proposition 2. *Suppose Assumption 1 holds and there is no noise. Then, for any sequence $\{\theta_t\}_{t\in[T]}$, we have*

(a) $R_T(\{\ell_t^{sub}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}), R_T(\{\ell_t^{est}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$, *and* $R_T(\{\ell_t^{pre}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$ *are all nonnegative;*

(b) $R_T(\{\ell_t^{ag}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) = R_T(\{\ell_t^{sub}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) + R_T(\{\ell_t^{est}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$;

(c) $R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) \geq R_T(\{\ell_t^{ag}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$, *and consequently* $R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$ *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 $R_T(\{\ell_t^{sub}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) \geq \frac{\gamma}{2} R_T(\{\ell_t^{pre}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$.*

As we have previously indicated, guaranteeing or proving the convexity of any of the loss functions $\ell_t^{pre}, \ell_t^{sub}, \ell_t^{est}$ or ℓ_t^{ag} as a function of θ is not easy. In contrast to this, Assumption 1 ensures that ℓ_t^{sim} is a convex function of θ. Therefore, under Assumption 1, any online convex optimization algorithm will be applicable for regret

minimization with respect to $\{\ell_t^{sim}\}_{t\in[T]}$, and as a consequence of Proposition 2 (and Corollary 1), those algorithms will also be minimizing regret with respect to the loss functions $\ell_t^{sub}, \ell_t^{est}$ and ℓ_t^{ag} (and ℓ_t^{pre}), as well.

3.2 Imperfect Information

A natural extension to consider for our framework is the case when the learner has access to only imperfect information about the agent's actions. As stated in Mohajerin Esfahani et al. (2018), there can be two types of noisy information. The first type is *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. The other type is *bounded rationality*, which means for all $t \in [T]$, the agent may choose a sub-optimal action instead of $x(\theta_{true}, u_t)$. Under such imperfect information, the convexity of ℓ^{sim} as shown in Proposition 1 and the relationship between ℓ^{sim}, ℓ^{ag} established in Observation 3 are still valid. However, since y_t is no longer guaranteed to be a minimizer of (1), Lemma 1 and Proposition 2 do 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.

We now examine how the regret relation in Proposition 2(b) may no longer be valid in an imperfect information regime where the learner observes $y_t = x(\theta_{true}; u_t) + \epsilon_t$. From Definition 1, for all θ, we have

$$\sum_{t\in[T]} \ell_t^{ag}(\theta) = \sum_{t\in[T]} \ell_t^{sub}(\theta) + \sum_{t\in[T]} \ell_t^{est}(\theta),$$

$$\text{and thus} \quad \min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t^{ag}(\theta) \geq \min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t^{sub}(\theta) + \min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t^{est}(\theta).$$

In the perfect information setup, we were able to strengthen the relation above into an equality because $\sum_{t\in[T]} \ell_t^{sub}(\theta)$, $\sum_{t\in[T]} \ell_t^{est}(\theta)$ and $\sum_{t\in[T]} \ell_t^{ag}(\theta)$ are all guaranteed to have a zero minimum in the absence of the noise ϵ_t; see Lemma 1. In the imperfect information case, this is no longer guaranteed to hold. Consequently, in the noisy case, the regret definition implies the relation

$$R_T(\{\ell_t^{ag}\}) \leq R_T(\{\ell_t^{sub}\}) + R_T(\{\ell_t^{est}\}).$$

Then, in the imperfect information regime, we conclude that regret convergence with respect to ℓ^{ag} is not sufficient to guarantee regret convergence with respect to the other loss functions.

4 Online Learning Algorithms

We next discuss the use of well-known OL regret minimization algorithms utilizing different oracles in our context. We start by introducing a classical online convex optimization (OCO) algorithm using a first-order oracle. Then, we review two OL algorithms which do not rely on a first-order oracle. These are respectively an *OL with a zeroth-order oracle*, and the *implicit OL utilizing a solution oracle*.

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 $R_T(\{\ell_t\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$. All of these algorithms are used in our numerical comparisons in Section 6.

4.1 Online Convex Optimization with First-Order Oracle

We review the well-known first-order online convex optimization algorithm, namely the online *Mirror Descent* (MD) algorithm in the proximal setup. Following the presentation and notation of Juditsky et al. (2011), let us introduce 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 $\|\zeta\|_* := \max_{\|\theta\|\leq 1} \langle \zeta, \theta \rangle$.

- *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 $\|\cdot\|$:

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

- *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 $V_\theta(\theta') \geq \frac{1}{2}\|\theta - \theta'\|^2 \geq 0$ for all $\theta \in \Theta^\circ$ and $\theta' \in \Theta$ follows from the strong convexity of ω.

- *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*: $\Omega := \max_{\theta \in \Theta} V_{\theta_\omega}(\theta) \leq \max_{\theta \in \Theta} \omega(\theta) - \min_{\theta \in \Theta} \omega(\theta)$.

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 3.

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., $\max_{s_t \in \partial\ell_t(\theta)} \|s_t\|_* \leq G$ 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*

$$\theta_{t+1} \coloneqq \mathrm{Prox}_{\theta_t}(\eta_t s_t) = \arg\min_{\theta \in \Theta} \{\langle \eta_t s_t, \theta \rangle + V_{\theta_t}(\theta)\}, \tag{6}$$

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

4.2 Online Learning with Zeroth-Order Oracle

We review the OL algorithm with a *zeroth-order oracle* from Flaxman et al. (2005) and the associated regret guarantees. At time step t, instead of computing $\nabla\ell_t(\theta_t)$ or $\partial\ell_t(\theta_t)$, it uses zeroth-order information $\ell_t(\theta_t)$ to approximate the (sub)gradient, and then performs an update with a projected (sub)gradient descent step. More formally, Flaxman et al. (2005) computes:

$$\begin{aligned} \hat{\theta}_t &\coloneqq \theta_t + \delta v_t, \\ \theta_{t+1} &\coloneqq \mathrm{Proj}_{(1-\alpha)\Theta}(\theta_t - \eta \ell_t(\hat{\theta}_t) v_t), \end{aligned} \tag{7}$$

where δ, α and η are pre-fixed constants affecting the regret bound, v_t is a random unit vector, Θ is the domain for the unknown parameter and $(1-\alpha)\Theta \coloneqq \{(1-\alpha)\theta : \theta \in \Theta\}$. Since this θ_t update is based on a random vector, we use the *expected regret*, defined below, as the performance measure.

$$R_T^{\mathbb{E}}(\{\ell_t\}_{t \in [T]}, \{\theta_t\}_{t \in [T]}) \coloneqq \mathbb{E}\left[\sum_{t \in [T]} \ell_t(\theta_t)\right] - \min_{\theta \in \Theta} \sum_{t \in [T]} \ell_t(\theta). \tag{8}$$

We next state two expected regret bounds from Flaxman et al. (2005). These bounds depend on a number of parameters. Suppose $\Theta \subset \mathbb{R}^n$, there exist $b_l, b_u \in \mathbb{R}_+$ such that

$$\{\theta \in \mathbb{R}^n : \|\theta\| \leq b_l\} \subseteq \Theta \subseteq \{\theta \in \mathbb{R}^n : \|\theta\| \leq b_u\},$$

and there exists $C \in \mathbb{R}_+$ such that $|\ell_t(\theta)| \leq C$ for all $\theta \in \Theta$ and for all $t \in [T]$.

Theorem 2 ((Flaxman et al. 2005, Theorem 1.)). *Suppose Θ is convex, and $\ell_t : \Theta \mapsto \mathbb{R}$ is a convex and differentiable function for $t \in [T]$. For any $T \geq \left(\frac{3b_u n}{2b_l}\right)^2$, suppose $\{\theta_t\}_{t\in[T]}$ is generated via (7) using $\eta = \frac{b_u}{C\sqrt{T}}$, $\delta = \sqrt[3]{\frac{b_l b_u^2 n^2}{12T}}$, $\alpha = \sqrt[3]{\frac{3b_u n}{2b_l\sqrt{T}}}$. Then, we have*

$$R_T^{\mathbb{E}}(\{\ell_t\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) \leq 3C\sqrt[3]{nb_u/b_l}T^{5/6}. \tag{9}$$

Theorem 3 ((Flaxman et al. 2005, Theorem 2.)). *Suppose Θ is convex, and $\ell_t : \Theta \mapsto \mathbb{R}$ is a convex and differentiable function for $t \in [T]$, and $f(\theta, y_t)$ is L-Lipschitz for each u_t, $t \in [T]$. Then, for T sufficiently large, $\{\theta_t\}_{t\in[T]}$ generated by (7) using $\eta = \frac{b_u}{C\sqrt{T}}$, $\delta = \sqrt{\frac{Cb_u b_l n}{3(Lb_l+C)}}T^{1/4}$, $\alpha = \frac{\delta}{b_l}$ leads to*

$$R_T^{\mathbb{E}}(\{\ell_t\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) \leq 2\sqrt{3b_u Cn(L + C/b_l)}T^{3/4}. \tag{10}$$

As noted in Flaxman et al. (2005), both regret bounds are worse than the common $O(\sqrt{T})$ regret bound that the usual online algorithms based on first-order feedback seek. This is not surprising as this algorithm relies on a much weaker form of feedback given by a zeroth-order oracle.

4.3 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

$$\theta_{t+1} := \text{argmin}_{\theta\in\Theta} L_t(\theta), \tag{11}$$

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 $\min_{\theta\in\Theta} L_t(\theta)$. Kulis and Bartlett (2010) establishes the following regret bound on the OL using implicit update (11).

Theorem 4 ((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 (11) attains*

$$R_T(\{\ell_t\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) \leq \sum_{t\in[T]} \frac{1}{\eta_t}\left[(1-\alpha_t)\eta_t \ell_t(\theta_t) + V_{\theta_t}(\theta^*) - V_{\theta_{t+1}}(\theta^*)\right]. \tag{12}$$

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 (12) further results in a $O(\sqrt{T})$ bound on $R_T(\{\ell_t\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$.

Theorem 5. *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 update rule (11) attains*

$$R_T(\{\ell_t\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) \leq 2\sqrt{\widehat{\Omega}G^2 T}. \tag{13}$$

5 Comparison with the Other Approaches

Bärmann et al. (2017) study a setting where f is a bilinear function of θ and x, i.e., $f(x;\theta) = \langle\theta, x\rangle$, and the learner has access to noiseless observations. This setting is indeed covered by our Proposition 1. They provide two OL algorithms generating sequence $\{\theta_t\}_{t\in[T]}$ based on online gradient descent and Multiplicative Weights Update (MWU) which are customized based on the domain Θ, and show via separate analysis (specialized

based on the bilinear structure of f and the domain Θ) that the resulting estimates $\{\theta_t\}_{t\in[T]}$ have vanishing average losses with respect to the loss functions ℓ^{est} and ℓ^{sub}. Using our Lemma 1, we can deduce that their algorithms also guarantee vanishing average regrets with respect to ℓ^{est} and ℓ^{sub}. In this setting, note that our Proposition 1 immediately that ℓ_t^{sim} is a convex function of θ for every $t \in [T]$, and thus we can utilize our OL framework with a first-order oracle, i.e., use the online Mirror Descent algorithm. In fact, then online MD immediately generalizes the online gradient descent based algorithm in Bärmann et al. (2017). In addition, their MWU algorithm, which can be viewed as a special case of Mirror Descent algorithm where entropy based distance generating function (d.g.f.) is used. Note that Bärmann et al. (2017) do not provide any regret bounds on ℓ^{pre}. In contrast to Bärmann et al. (2017) that focus on this simple form of bilinear functions f, using our flexible framework based on analyzing the relation between different loss functions and their regret bounds, we can establish regret bounds for these loss functions when f is more general.

The objective function f in the forward problem can be easily extended to more general functions by using nontrivial $f_1(x;u)$ and $f_2(\theta;u)$. In this respect, the functions $f_1(x;u)$ that are strongly convex are of special interest. Suppose $f_1(x;u)$ is strongly convex in x with parameter γ, then the regret bound derived for ℓ^{pre} in Corollary 1 also applies.

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

$$\min_x \left\{ \frac{1}{2} x^\top P x - \langle \theta_{true}, x \rangle : \ x \in \mathcal{X}(u) \right\}. \tag{14}$$

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) proposes a regret minimization algorithm utilizing an implicit OL framework introduced in 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. They identify a restrictive technical condition (Dong et al. 2018a, Assumption 3.3) that can guarantee the convexity of $\ell_t^{pre}(\theta)$. Nevertheless, they remark that this condition is 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 (14) 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 (14), the algorithm from Dong et al. (2018a) updates θ_{t+1} as the optimal solution of the following bilevel program:

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

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 the sequence $\{\theta_t\}_{t\in[T]}$. The main convergence result (Dong et al. 2018a, Theorem 3.2) proves that under their assumptions by choosing the step size $\eta_t \propto \frac{1}{\sqrt{t}}$, the sequence of estimates$\{\theta_t\}_{t\in[T]}$ generated with the above update yields a $O(\sqrt{T})$ bound on the regret $R_T(\{\ell_t^{pre}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$.

As discussed in Sections 4.1 and 4.3, in the noiseless case, our OL framework with a first-order oracle, or a solution oracle, can attain the same regret bound with respect to any convex loss function ℓ as their implicit OL algorithm using an MISOCP-based solution oracle. In addition, the format of f in (14) immediately satisfies our Assumption 1. Then, Proposition 1 implies the convexity of $\{\ell_t^{sim}\}_{t\in[T]}$ for any $\mathcal{X}(u)$, and thus our framework is applicable to (14) and through Proposition 2 and Corollary 1 we can attain regret bounds with respect to all of ℓ^{sim}, ℓ^{ag}, ℓ^{est}, ℓ^{sub} and ℓ^{pre}, without further structural assumptions on the agent's domain. In contrast, in order to guarantee a regret bound the algorithm in Dong et al. (2018a) requires additional conditions on the agent's domain, see (Dong et al. 2018a, Assumptions 3.1,3.2,3.3), and is specifically focused on ℓ^{pre} and provides no insight on other performance measures of interest captured by ℓ^{est} and ℓ^{sub}.

One aspect that Dong et al. (2018a) emphasizes but we do not address is the noise in observations. The regret bound from Dong et al. (2018a) holds even with y_t being a noisy observation of $x(\theta_{true};u_t)$. However,

as previously discussed, it is unclear whether our framework can be conveniently extended to the imperfect information setup.

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.

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; see Section 3.2 for a discussion of the main issues. Hence, these experiments essentially shed light to their empirical performance in the noisy setup.

All algorithms are coded in Python 3.6, 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 customer's feasible actions. For each $t \in [T]$, we let u_t denote the market parameters relevant to the customer's decisions at period t. When constraint parameters are fixed as u_t, a customer'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 customer's preferences over the products. We model the customer's optimization problem as a maximization of her/his utility function subject to feasibility constraints. The learner knows the customer's decision problem up to the parameter vector θ_{true}, and the learner's goal is to estimate θ_{true} using observations of the customer'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 customer's utility function.

(a) For direct comparison with Dong et al. (2018a), we examine the case where the customer's utility function has the quadratic form (14), i.e., the customer's action $x(\theta_{true}; u_t)$ is given by

$$x(\theta_{true}; u_t) \coloneqq \arg\max_x \left\{ -\frac{1}{2} x^\top P x + \langle \theta_{true}, x \rangle : \ x \in \mathcal{X}(u_t) \right\}, \tag{15}$$

where $P \in \mathbb{S}^n_{++}$ is a fixed matrix known by both the learner and the customer, 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 customer has a CES utility function with $\rho = 2$. Hence, in period t, the customer's action $x(\theta_{true}; u_t)$ is given by

$$x(\theta_{true}; u_t) \coloneqq \arg\max_x \left\{ \sum_{i \in [n]} -(\theta_{true})_i x_i^2 : \ x \in \mathcal{X}(u_t) \right\}. \tag{16}$$

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

These particular forms of utility functions in (15) and (16) imply that the dimensions of θ and x are the same, i.e., $p = n$. Moreover, observe that both of the objective functions in (15) and (16) 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 as well as 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 customer: $\mathcal{X}(u_t) = \mathcal{X}^{ck}(p_t, b_t) \coloneqq \{x \in \mathbb{R}^n_+ : \langle p_t, x\rangle \leq b_t\}$, 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 (15) 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 (16), 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 implemented four OL algorithms and compare their performances. Recall that the loss function ℓ^{sim} is convex under Assumption 1 (see Proposition 1) and in the perfect information case its regret upper bounds the regrets with respect to all other loss functions as well; see Proposition 2 and Corollary 1. Based on this and our theoretical guarantees for convex loss functions in Section 4, we design three OL algorithms minimizing regret with respect to ℓ^{sim} equipped with different oracles: one with a first-order oracle, one with a zeroth-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}, 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 (6) for the OL with first-order oracle is given explicitly by the formula

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

where $s_t(\theta_t)$ is a subgradient of $\ell_t^{sim}(\theta)$ at θ_t.

For the OL with the zeroth-order oracle, at each time step t, we choose θ_{t+1} as an average of k estimates generated from the approximate projected (sub)gradient descent update (7). The parameters η, δ, α are chosen as specified in Theorem 2. In order to simulate the expected regret performance, we use an average of $k = 20$ iterations instead of doing a one-shot update.

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 2.2 except $\ell_t^{sim}(\theta)$, (11) 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, (11) 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 the loss function 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 B 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) using the KKT conditions of the inner (agent's) problem and then introducing binary variables to rewrite the complementarity constraints as linear inequalities. 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 (15), 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 B. 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 (16) 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

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 that we listed at the beginning of Section 6. We provide additional plots with more detailed information in Appendix C.

6.3.1 Learning a Quadratic Utility Function

In this case, we assume that the agent's utility function is of form (15). We first compare the performance of the four 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. These means are computed based on all 50 random instances generated in the experiment. 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, the OL with the zeroth-order oracle has a significantly worse performance yet still shows promise for a convergence to zero but at a very slow rate, and the performance of the other three OL algorithms are quite similar. This empirical observation is in line with the theoretical regret guarantees given in Section 4; recall that this particular domain type was the focus of Dong et al. (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 disadvantages of the OL with the zeroth-order oracle 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 4.3, 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 (15) 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 (15) 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 next report the time spent by all four 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. In each iteration of the OL with the zeroth-order oracle, we include the time to compute $\ell_t(\hat{\theta}_t)$ and θ_{t+1} from all $k = 20$ repetitions. 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, OL with the zeroth-order oracle finishes in 545 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. For an arbitrary instance with the polytope domain polytope, OL with the first-order oracle, OL with the zeroth-order oracle, ℓ^{sim}-based implicit OL with the solution oracle and ℓ^{pre}-based implicit OL with the solution oracle take 0.08 seconds, 792 seconds, 1.97 seconds, and 153 seconds, respectively. 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

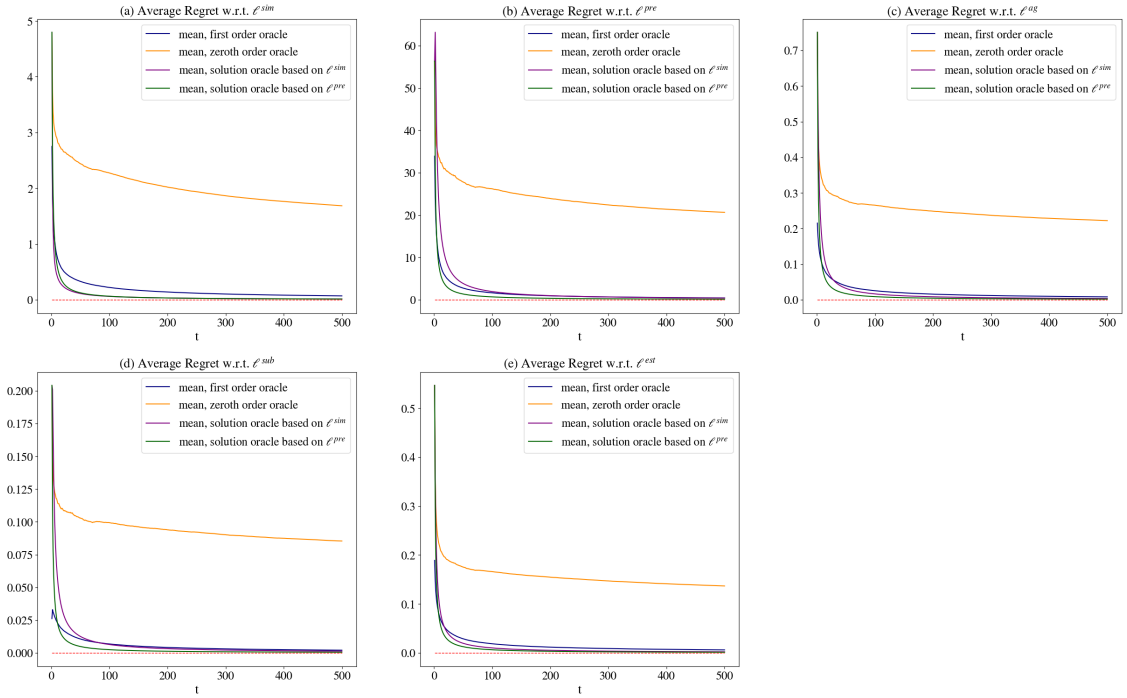

Figure 1: Means of average regret with respect to different loss functions over $T = 500$ iterations for continuous knapsack instances with $n = 50$.

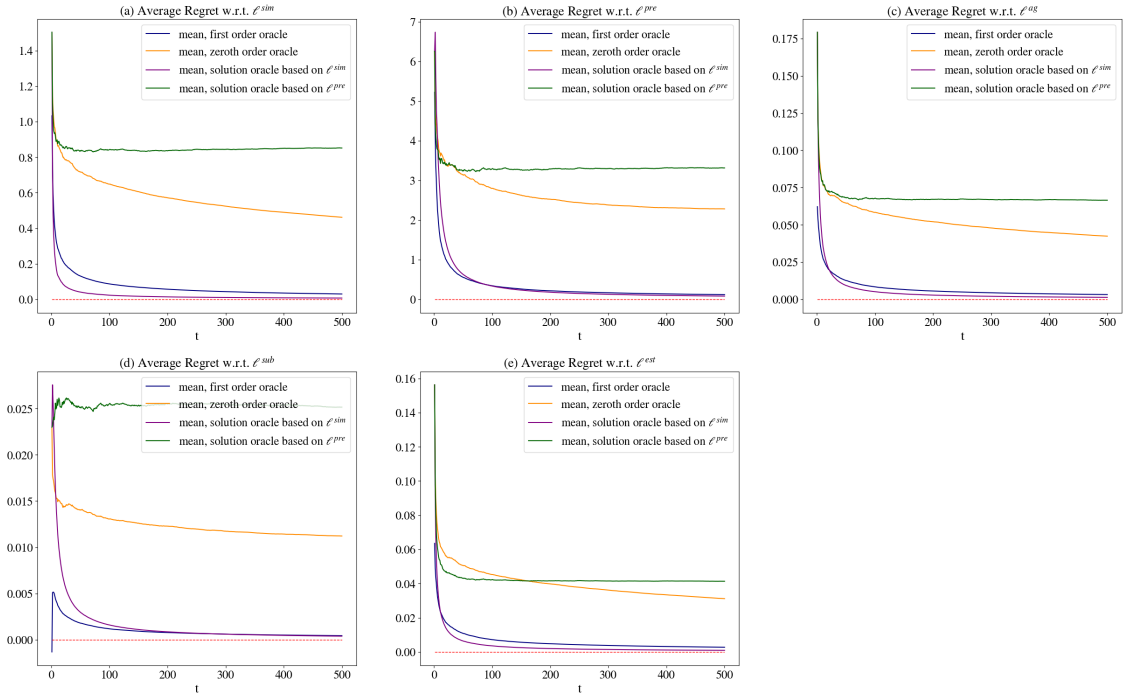

Figure 2: Means of average regret with respect to different loss functions over $T = 500$ iterations for continuous polytope instances with $n = 50$ and $m = 10$.

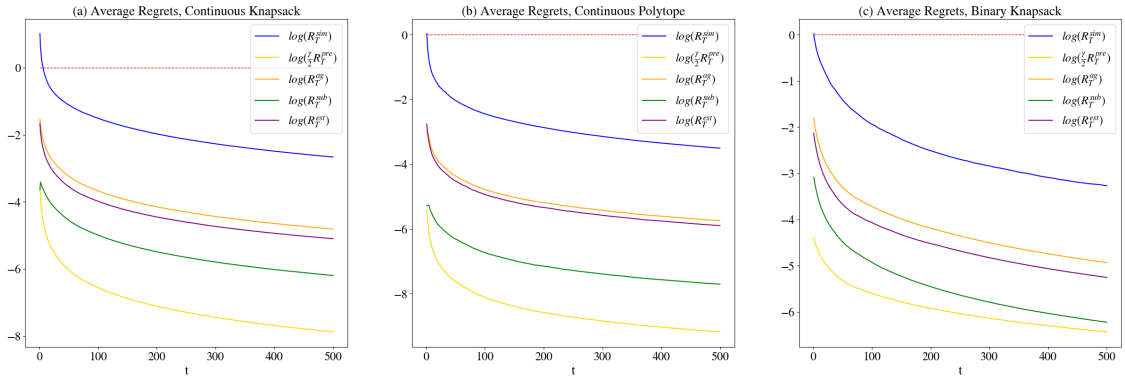

Figure 3: Means of average regret (on a logarithmic scale) with respect to different loss functions over $T = 500$ iterations for (a) continuous knapsack instances, (b) continuous polytope instances, (c) binary knapsack instances, when OL with first-order oracle is used.

ℓ^{pre}-based implicit OL with the MISOCP solution oracle one from Dong et al. (2018a) or the one utilizing a zeroth-order oracle.

We next examine the regret performance of OL with the first-order oracle with respect to different loss functions $\ell(\theta)$ from Section 2.2. Figures 1 and 2 indicate that the average regret with respect to any of the five 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 3 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: $R_T^{sim} \geq R_T^{ag} = R_T^{sub} + R_T^{est} \geq \frac{\gamma}{2} R_T^{pre} \geq 0$, where γ is the strong convexity parameter of the quadratic objective function in (15). Recall that our instance generation guarantees $P \in \mathbb{S}^n_{++}$, i.e., $\lambda_{min}(P) > 0$, and then by the definition of strong convexity, we deduce $\gamma = \lambda_{min}(P)$. Figure 3 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 3.

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 becomes slightly more obvious in the polytope setting than in the continuous knapsack setting. In Figure 4, 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^{ag}, 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 visibly more noticeable in the case of loss functions ℓ^{pre} and ℓ^{sim}.

6.3.2 Learning a CES Utility Function

Here, we examine the case when the agent's utility function is of form (16) and summarize our findings on the average regrets in Figure 5. We again note that the OL with the the first-order oracle has a noticeable advantage over the other two OL algorithms in terms of the regret convergence. Contrary to the previous experiments, the OL algorithm based on the zeroth-order oracle gives better regret performance than the one with the solution oracle. In this case, on a typical instance, OL with the first-order oracle takes 0.12 seconds to complete, OL with the zeroth-order oracle takes 103 seconds, and ℓ^{sim}-based implicit OL with the solution oracle takes 2.02 seconds. We also note that in contrast to the setting in Section 6.3.1 the regret convergence for CES utility function learning is not as good: after $T = 500$ iterations, the average regrets from OL with the first-order oracle are not as close to 0 yet, but all OL algorithms demonstrate decreasing trends as expected from our theoretical results.

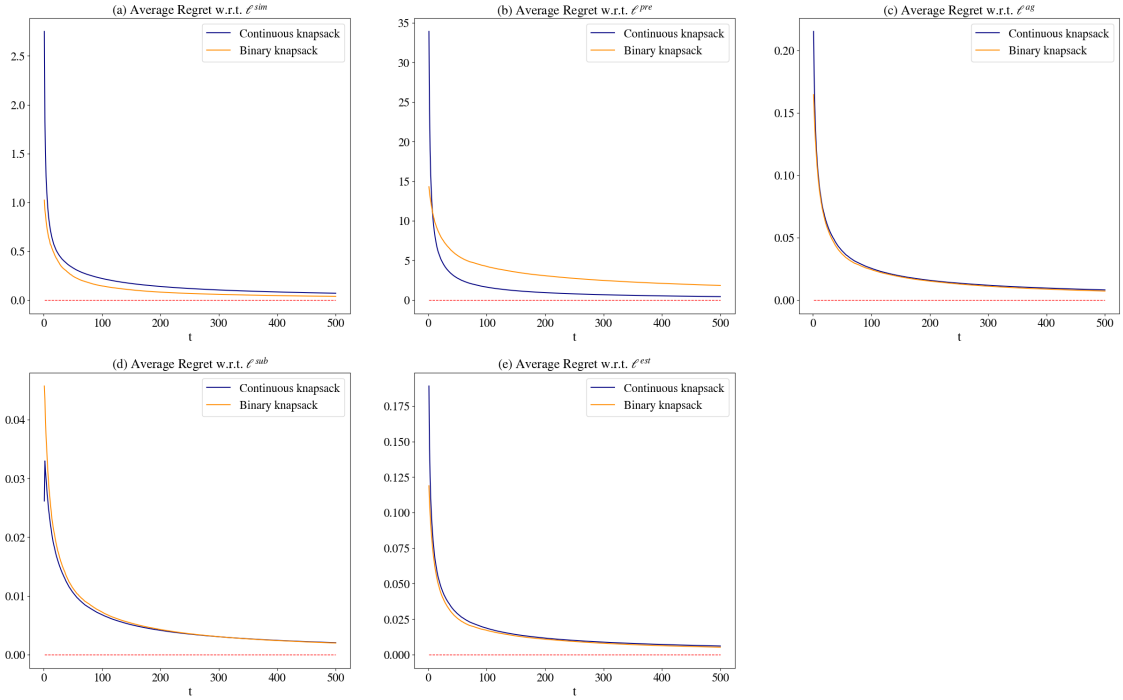

Figure 4: Means of average regret with respect to different loss functions contrasting continuous knapsack instances with binary knapsack instances, when OL with the first-order oracle is used.

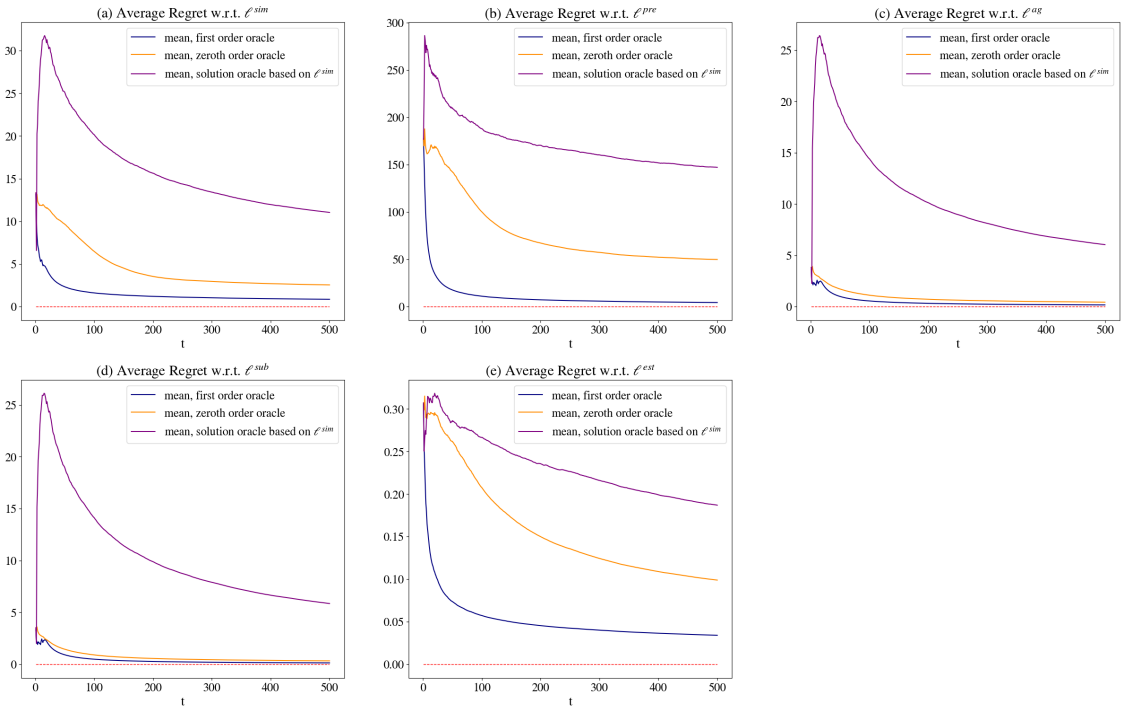

Figure 5: Learning a CES utility function: means of average regrets with respect to different loss functions over $T = 500$ iterations for equality constrained knapsack instances.

6.4 Imperfect Information

We next study the robustness of these 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.

Before discussing the performance and robustness of OL algorithms, we note some key differences with the perfect information case. Because the noisy observations $\{y_t\}$ may be suboptimal or even infeasible to the agent's problems, we no longer have the convenient guarantee of Lemma 1 that θ_{true} gives the offline optimal losses $\min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t(\theta) = 0$ with respect to $\ell^{sub}, \ell^{est}, \ell^{ag}$ and ℓ^{pre}. Specifically, $\ell_t^{est}(\theta) = f(x(\theta; u_t); \theta_{true}, u_t) - f(y_t; \theta_{true}, u_t) < 0$ is possible when y_t is feasible and $x(\theta; u_t)$ is a better solution than y_t for the agent; $\ell_t^{sub}(\theta) = f(y_t; \theta, u_t) - f(x(\theta; u_t); \theta, u_t) < 0$ can happen when y_t is outside the agent's feasible domain $\mathcal{X}(u_t)$. Furthermore, negative values in either $\ell_t^{est}(\theta)$ or $\ell_t^{sub}(\theta)$ may lead to a negative $\ell_t^{ag}(\theta)$. The prediction loss $\ell_t^{pre}(\theta)$ is still nonnegative, but the lowest value is not necessarily $\ell_t^{pre}(\theta_{true})$. Therefore, when the observations y_t are noisy, computing regret R_T requires solving the optimization problem $\min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t(\theta)$, which can be computationally difficult as the term $x(\theta; u_t)$ makes it a bilevel program. To avoid such difficulty, we plot the outcomes differently: instead of showing average regrets with respect to all of our loss functions, we show only the average regret with respect to ℓ^{sim}, but we also report the average losses computed with the regular loss definitions and with y_t replaced by $x(\theta_{true}; u_t)$.

We report the results for when the agent's problem has the form (15) with the domain $\mathcal{X}(u_t) = \mathcal{X}^{ck}(u_t)$ in Figures 6, 7, 8, and 9. In the imperfect information setup, while only the ℓ^{pre} minimization-based implicit OL with the solution oracle has a theoretical guarantee for convergence under further assumptions on $\mathcal{X}(u_t)$, we observe a convergence behavior for all three of the ℓ^{sim}-based OL algorithms as well. For example, in the case of small noises, Figure 6 shows that the average regret with respect to ℓ^{sim} has a quite similar convergence trend as in Figure 1. Figures 8, 9 show that the effects of the noises become more noticeable when their magnitudes are larger as the average regret with respect to ℓ^{sim} appears to converge much more slowly for the ℓ^{sim}-based OL algorithms and the ℓ^{pre}-based implicit OL with the solution oracle seems to fail to converge, and there are cases of negative average losses with respect to a number of loss functions. It is notable that in the case of ℓ^{sim}-based OL algorithms, the average losses in terms of ℓ^{pre} computed with respect to $x(\theta_{true}; u_t)$ decrease with T even under large noises, which indicates that these OL algorithms' predictions of the agent's actions are becoming more accurate as T increases. We further note that such trends are most explicit in the OL with the first-order oracle. These results demonstrate that ℓ^{sim}-based OL algorithms have some degree of robustness for certain types of imperfect information, and the performance of the ℓ^{sim}-based OL algorithm with first-order feedback seems to be slightly superior in the noisy setup. See Appendix D for the corresponding numerical study in the CES setup.

Acknowledgments

This research is supported in part by NSF grant CMMI 1454548.

References

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

Amin, K., Cummings, R., Dworkin, L., Kearns, M., and Roth, A. (2015). Online learning and profit maximization from revealed preferences. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Aswani, A., Shen, Z.-J. M., and Siddiq, A. (2018). Inverse optimization with noisy data. *Operations Research*, 66(3):870–892.

Balcan, M.-F., Daniely, A., Mehta, R., Urner, R., and Vazirani, V. V. (2014). Learning economic parameters from revealed preferences. In *International Conference on Web and Internet Economics*, pages 338–353. Springer.

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

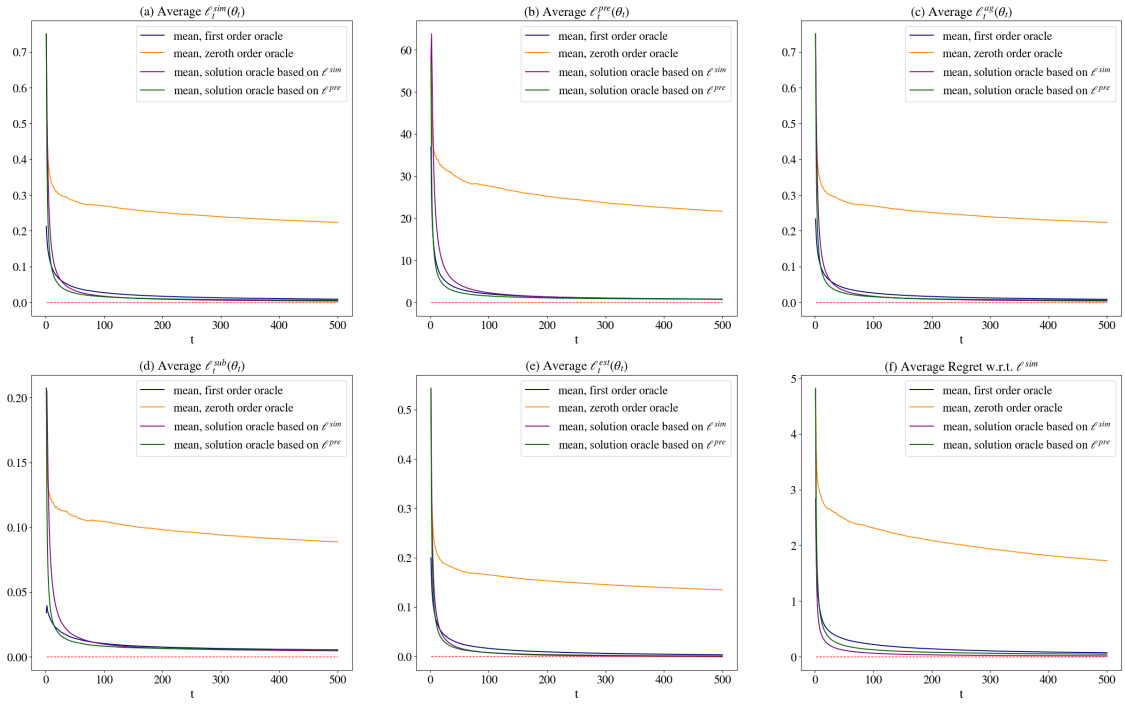

Figure 6: Learning a quadratic utility function under small noises: means of average losses with respect to different loss functions and means of average regret with respect to ℓ^{sim} over $T = 500$ iterations for continuous knapsack instances.

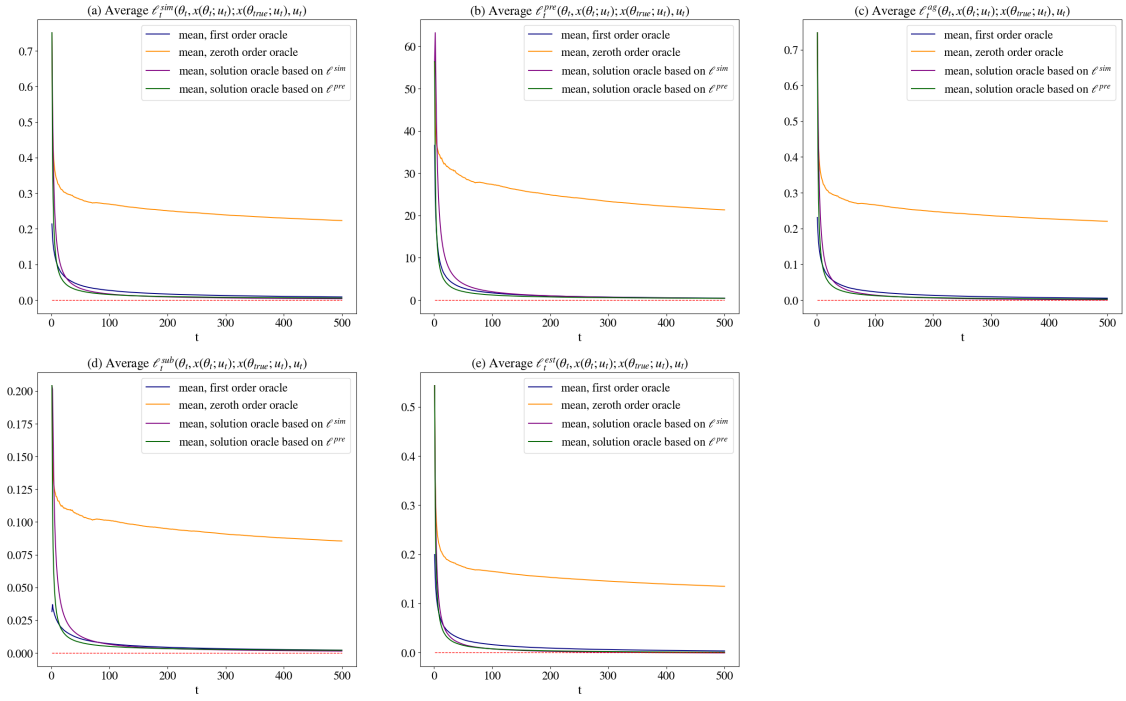

Figure 7: Learning a quadratic utility function under small noises: means of average losses with respect to different loss functions measured at $x(\theta_{true}; u_t)$ over $T = 500$ iterations for continuous knapsack instances.

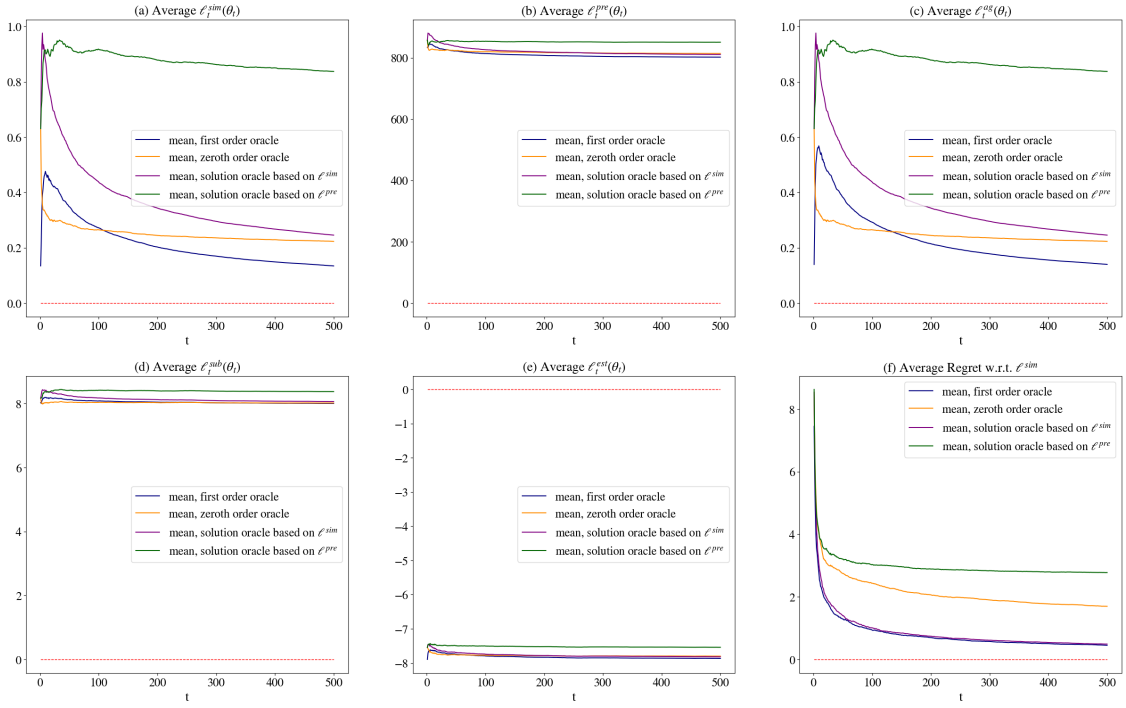

Figure 8: Learning a quadratic utility function under large noises: means of average losses with respect to different loss functions and means of average regret with respect to ℓ^{sim} over $T = 500$ iterations for continuous knapsack instances.

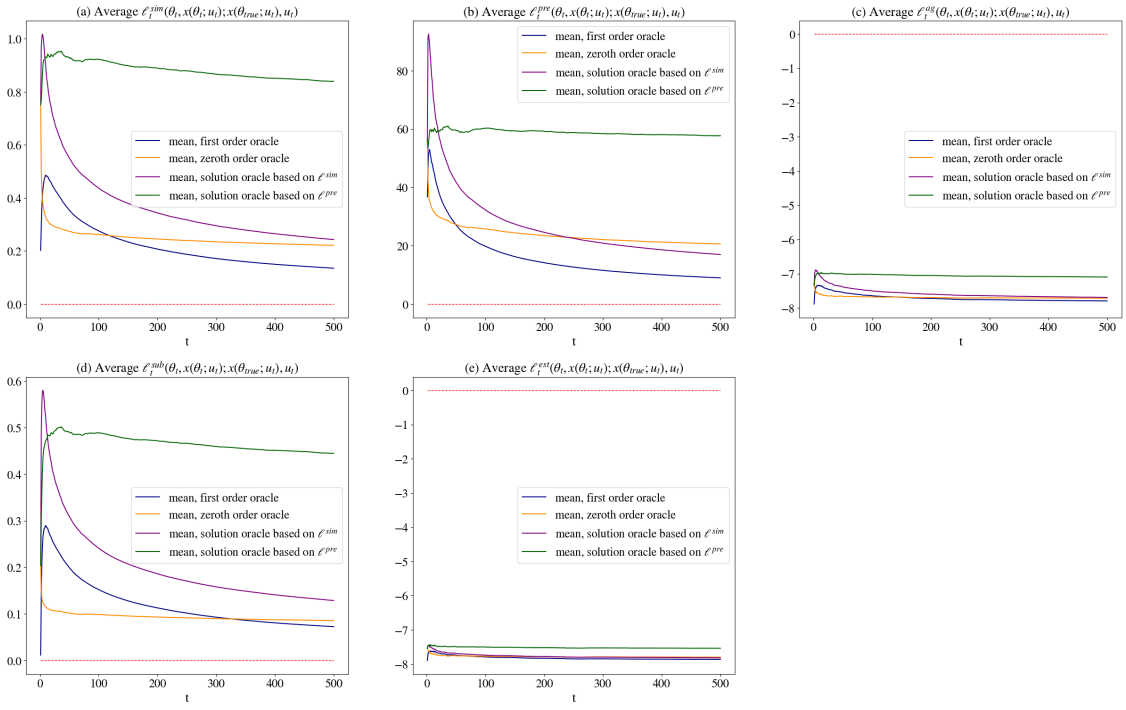

Figure 9: Learning a quadratic utility function under large noises: means of average losses with respect to different loss functions measured at $x(\theta_{true}; u_t)$ over $T = 500$ iterations for continuous knapsack instances.

Beigman, E. and Vohra, R. (2006). Learning from revealed preference. In *Proceedings of the 7th ACM Conference on Electronic Commerce*, EC '06, pages 36–42, New York, NY, USA. ACM.

Dong, C., Chen, Y., and Zeng, B. (2018a). Generalized inverse optimization through online learning. In *Proceedings of the 31st Conference on Neural Information Processing Systems*, NeurIPS '18, pages 86–95.

Dong, J., Roth, A., Schutzman, Z., Waggoner, B., and Wu, Z. S. (2018b). Strategic classification from revealed preferences. In *Proceedings of the 2018 ACM Conference on Economics and Computation (EC '18)*, pages 55–70. ACM.

Flaxman, A. D., Kalai, A. T., Kalai, A. T., and McMahan, H. B. (2005). Online convex optimization in the bandit setting: gradient descent without a gradient. In *Proceedings of the sixteenth annual ACM-SIAM Symposium on Discrete Algorithms (SODA '05)*, pages 385–394. Society for Industrial and Applied Mathematics.

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

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

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

Ji, Z., Mehta, R., and Telgarsky, M. (2018). Social welfare and profit maximization from revealed preferences. In *International Conference on Web and Internet Economics*, pages 264–281. Springer.

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

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

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

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

Roth, A., Ullman, J., and Wu, Z. S. (2016). Watch and learn: Optimizing from revealed preferences feedback. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 949–962. ACM.

Saharoy, D. and Tulabandhula, T. (2018). An online algorithm for learning buyer behavior under realistic pricing restrictions. *arXiv preprint arXiv:1803.01968*.

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

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

Zadimoghaddam, M. and Roth, A. (2012). Efficiently learning from revealed preference. In *International Workshop on Internet and Network Economics*, pages 114–127. Springer.

A Proofs

A.1 Proof of Observation 1

Part (a) holds because $y_t = x(\theta_{true}; u_t)$ is feasible to (2) and is an optimal solution to (1), whereas $x(\theta; u_t)$ is an optimal solution to (2) and is feasible to (1). Part (b) follows from evaluating these loss functions at $\theta = \theta_{true}$, and noting that we are considering the noiseless case. ■

A.2 Proof of Observation 2

$$\begin{aligned}
\ell^{ag}(\theta, x(\theta; u_t); y_t, u_t) &= f(y_t; \theta, u_t) - f(x(\theta; u_t); \theta, u_t) + f(x(\theta; u_t); \theta_{true}, u_t) - f(y_t; \theta_{true}, u_t) \\
&= f_1(y_t; u_t) + f_2(\theta; u_t) + h(y_t, \theta; u_t) - f_1(x(\theta; u_t); u_t) - f_2(\theta; u_t) - h(x(\theta; u_t), \theta; u_t) \\
&+ f_1(x(\theta; u_t); u_t) + f_2(\theta_{true}; u_t) + h(x(\theta; u_t), \theta_{true}; u_t) \\
&- f_1(y_t; u_t) - f_2(\theta_{true}; u_t) - h(y_t, \theta_{true}; u_t) \\
&= h(y_t, \theta; u_t) - h(x(\theta; u_t), \theta; u_t) + h(x(\theta; u_t), \theta_{true}; u_t) - h(y_t, \theta_{true}; u_t).
\end{aligned}$$

■

A.3 Proof of Proposition 1

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. ■

A.4 Proof of Lemma 1

For convenience, ℓ_t refers to any one of $\ell_t^{pre}, \ell_t^{sub}, \ell_t^{est}, \ell_t^{ag}$. From the definition of these loss functions and Observation 1, we have $\sum_{t\in[T]} \ell_t(\theta) \geq 0$ for all θ. Since $y_t = x(\theta_{true}, u_t)$ for all t and $\ell_t(\theta_{true}) = 0$ for all t, we have $\sum_{t\in[T]} \ell_t(\theta_{true}) = 0$ and thus $\min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t(\theta) = 0$. ■

A.5 Proof of Proposition 2

(a) Let ℓ_t represent any of the loss functions ℓ_t^{sub}, ℓ_t^{est}, and ℓ_t^{pre}. In the noiseless case, by Observation 1, ℓ_t is a nonnegative function of θ. Then, using the definition of regret and Lemma 1, we deduce 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 and Lemma 1, we have

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

where the second equation follows from the definition of ℓ_t^{ag}, and the last equation follows from Lemma 1.

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

$$\begin{aligned} R_T(\{\ell_t^{sim}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) &= \sum_{t\in[T]} \ell_t^{sim}(\theta_t) - \min_{\theta\in\Theta} \sum_{t\in[T]} \ell_t^{sim}(\theta) \\ &\geq \sum_{t\in[T]} \ell_t^{sim}(\theta_t) - \sum_{t\in[T]} \ell_t^{sim}(\theta_{true}) \\ &= \sum_{t\in[T]} \ell_t^{ag}(\theta_t) - 0 \\ &= R_T(\{\ell_t^{ag}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}), \end{aligned}$$

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 Observation 3, and the last equation follows from Lemma 1. The final part of Part (c) follows from Part (a) and Part (b). ■

A.6 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 1, we deduce $R_T(\{\ell_t^{sub}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]}) \geq \frac{\gamma}{2} R_T(\{\ell_t^{pre}\}_{t\in[T]}, \{\theta_t\}_{t\in[T]})$. ■

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

B.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 that under Assumption 1, the agent's objective is to minimize $f(x;\theta,u) = f_1(x;u) + f_2(\theta;u) + \langle\theta, c(x)\rangle$, where $c(x) = (c_1(x), \ldots, c_p(x))$, and consequently the form of ℓ^{sim} from Definition 2 is as follows:

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

Then, $\ell_t^{sim}(\theta)$ is the sum of a linear function of θ given by $\langle\theta, c(y_t) - c(x(\theta_t;u_t))\rangle$ and a constant term $\langle\theta_{true}, c(x(\theta_t;u_t)) - c(y_t)\rangle$. 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, and thus 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 (15) 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 (16), 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

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

B.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:

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

where

$$x(\theta;u_t) \in \arg\min_{x} \left\{ f(x;\theta,u_t) :\ g(x;u_t)\le 0,\ x\in\mathcal{X}\right\}.$$

Recall that when the agent's problem has the form (15) with a continuous polytope domain, i.e., $\mathcal{X}(u_t) = \mathcal{X}^{cp}(A_t,c_t)$, we have

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

where $P\in\mathbb{S}^n_{++}$ is a fixed 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:

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

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 (16) 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.

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

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.

C Supplemental Figures for the Perfect Information Experiments

In this appendix, we provide figures that summarize not only the means, but also standard deviations of the corresponding regret bounds for our perfect information experiments.

D Supplemental Material for the Imperfect Information Experiments

In this appendix, we provide details on imperfect information experiments in the CES setup, i.e., when the agent's problem has the form (16) with the equality constrained knapsack domain, i.e., $\mathcal{X}(u_t) = \mathcal{X}^{eck}(u_t)$, under small noises in Figures 13, 14, and under large noises in Figures 15, 16. Our findings are similar to Section 6.4. Both OL algorithms utilizing the first-order oracle and the zeroth-order oracle are again robust to the small noises, and generate average losses converging in roughly the same patterns as their counterparts in the perfect information case. Not surprisingly, the performance of the algorithms degrade as the noises get larger. The OL algorithm with the first-order oracle is still more robust than the one with the zeroth-order oracle, as demonstrated by its much better loss performance.

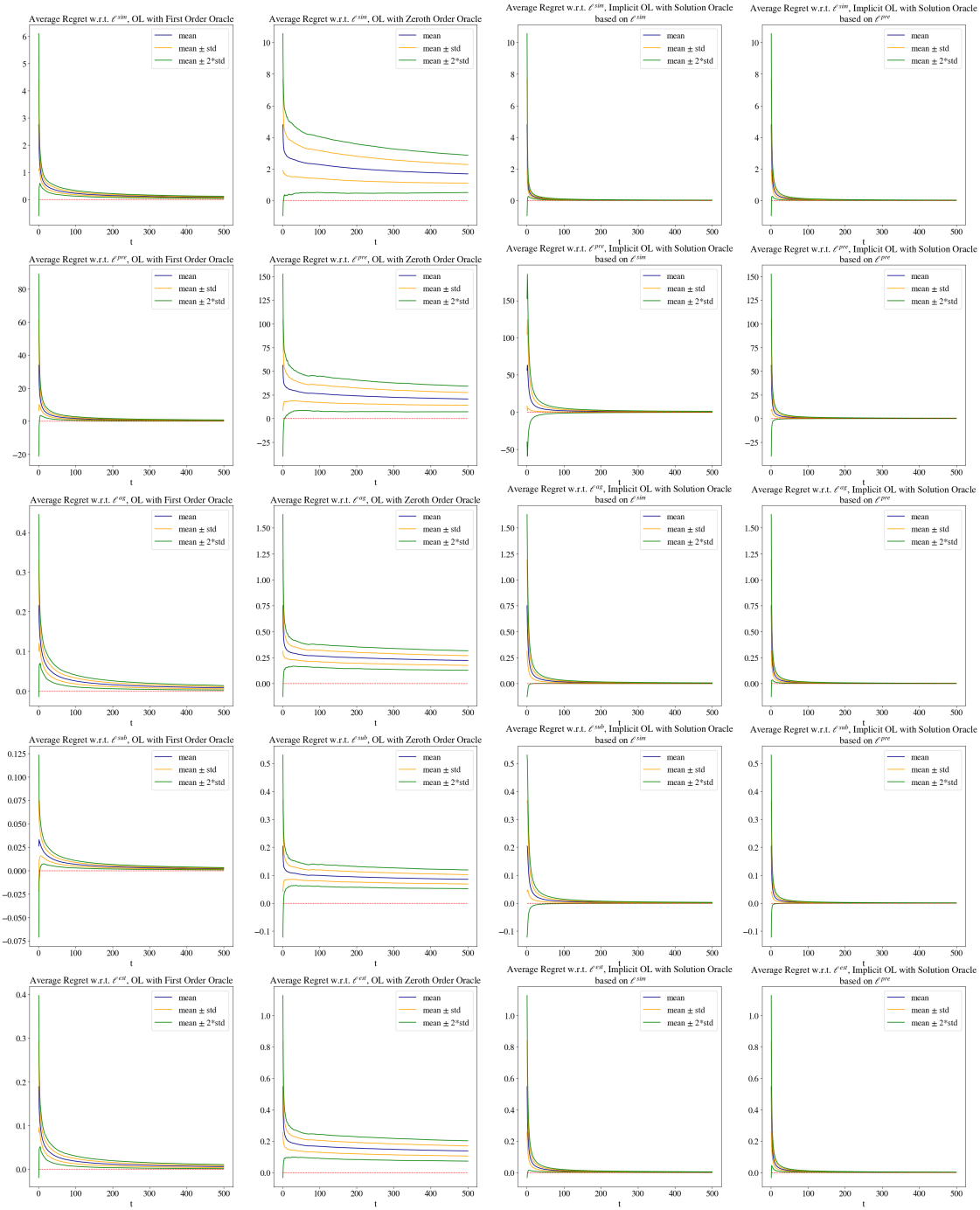

Figure 10: Means and standard deviations of average regret with respect to $\ell^{sim}, \ell^{pre}, \ell^{ag}, \ell^{sub}, \ell^{est}$ loss functions over $T = 500$ iterations for continuous knapsack instances with $n = 50$.

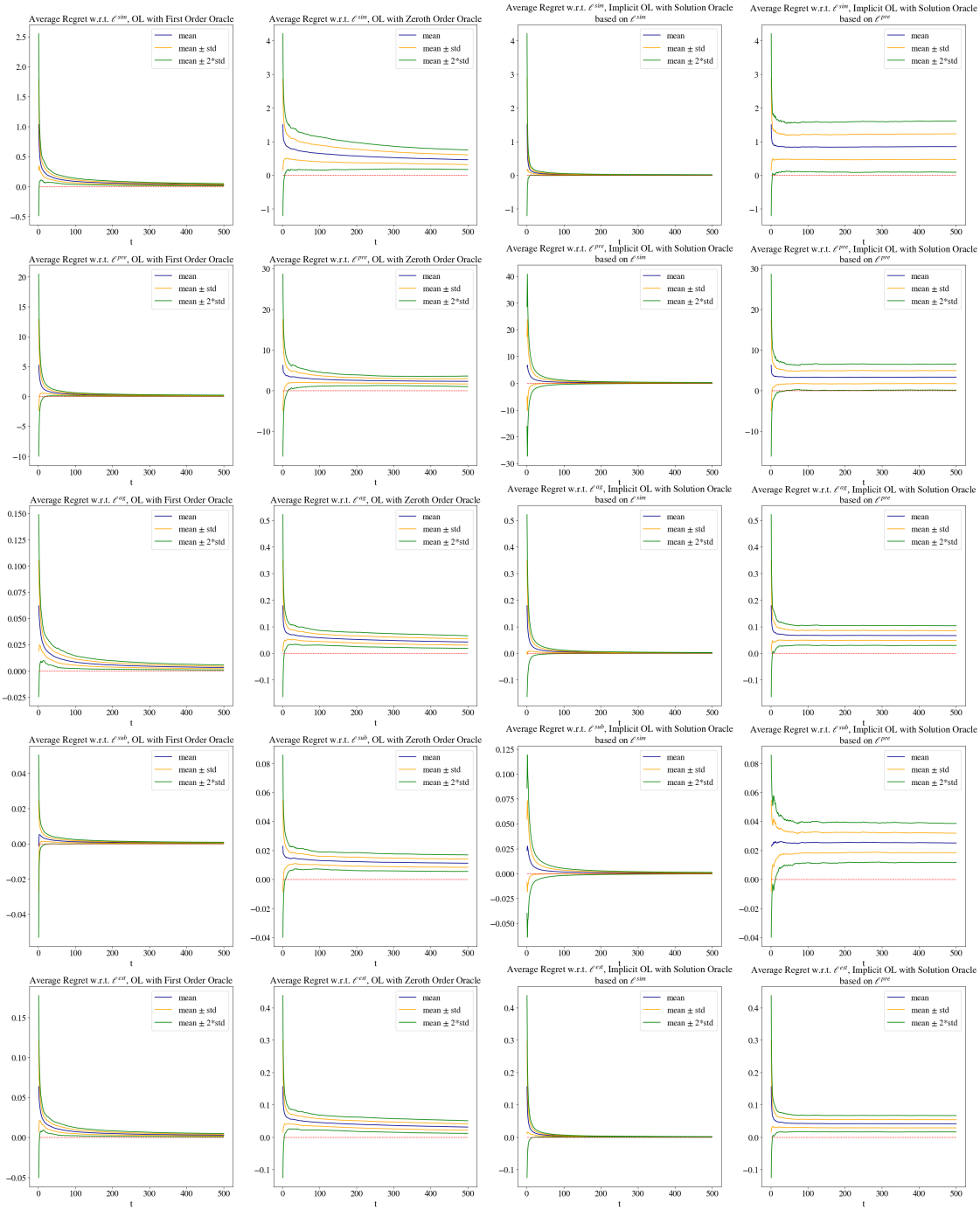

Figure 11: Means and standard deviations of average regret with respect to $\ell^{sim}, \ell^{pre}, \ell^{ag}, \ell^{sub}, \ell^{ag}$ loss functions over $T = 500$ iterations for continuous polytope instances with $n = 50, m = 10$.

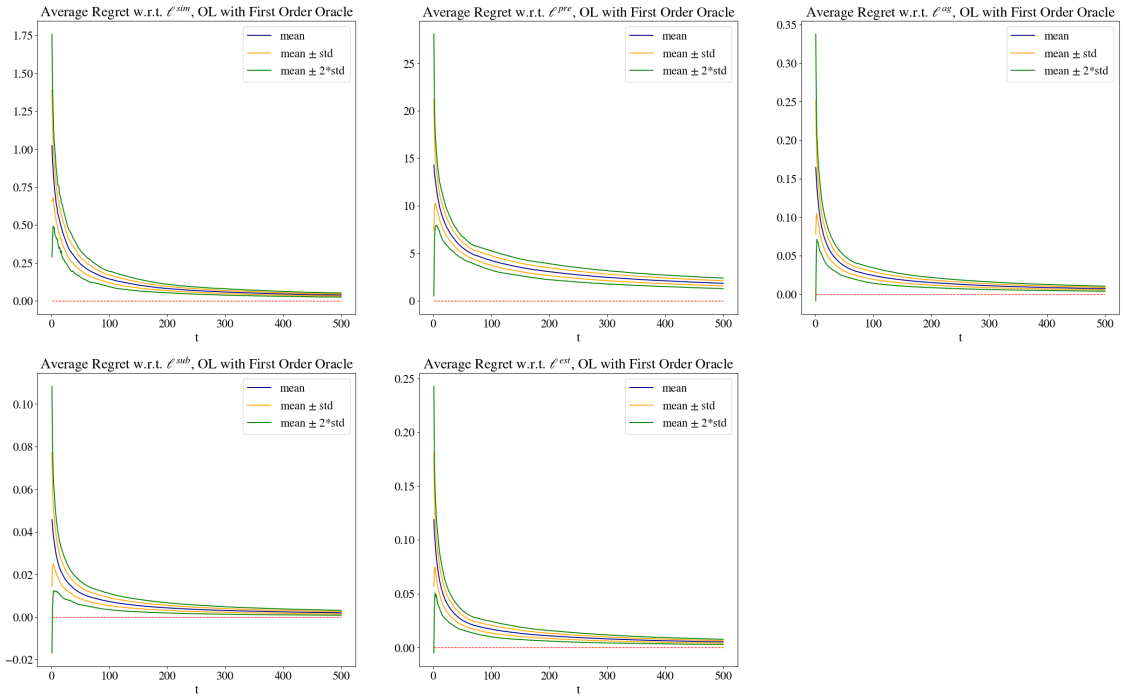

Figure 12: Means and standard deviations of average regret with respect to $\ell^{sim}, \ell^{pre}, \ell^{ag}, \ell^{sub}, \ell^{est}$ loss functions over $T = 500$ iterations for binary knapsack instances with $n = 50$.

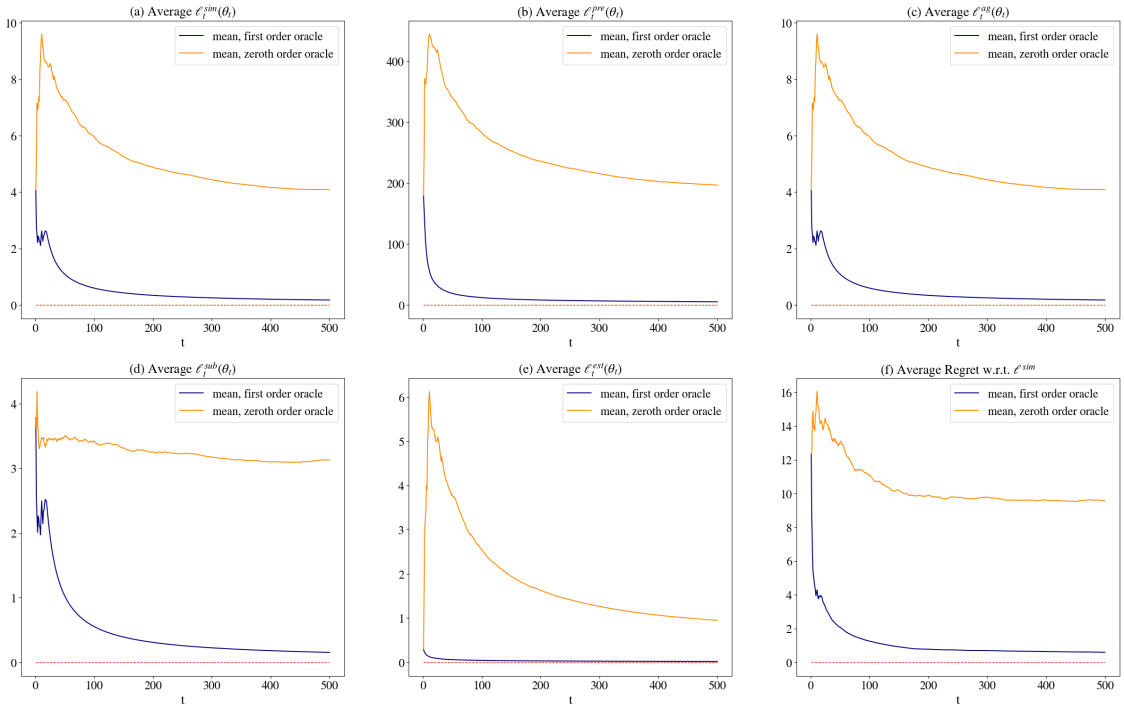

Figure 13: Learning a CES utility function under small noises: means of average losses with respect to different loss functions and means of average regret with respect to ℓ^{sim} over $T = 500$ iterations for equality constrained knapsack instances with $n = 50$.

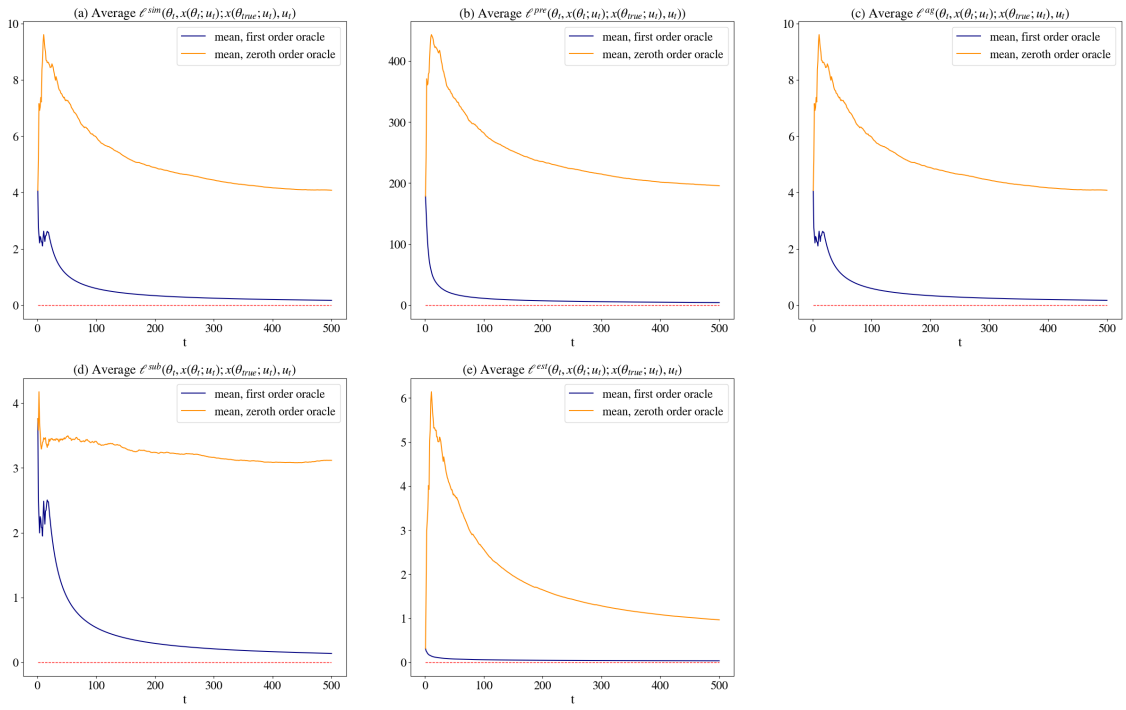

Figure 14: Learning a CES utility function under small noises: means of average losses with respect to different loss functions measured at $x(\theta_{true}; u_t)$ over $T = 500$ iterations for equality constrained knapsack instances with $n = 50$.

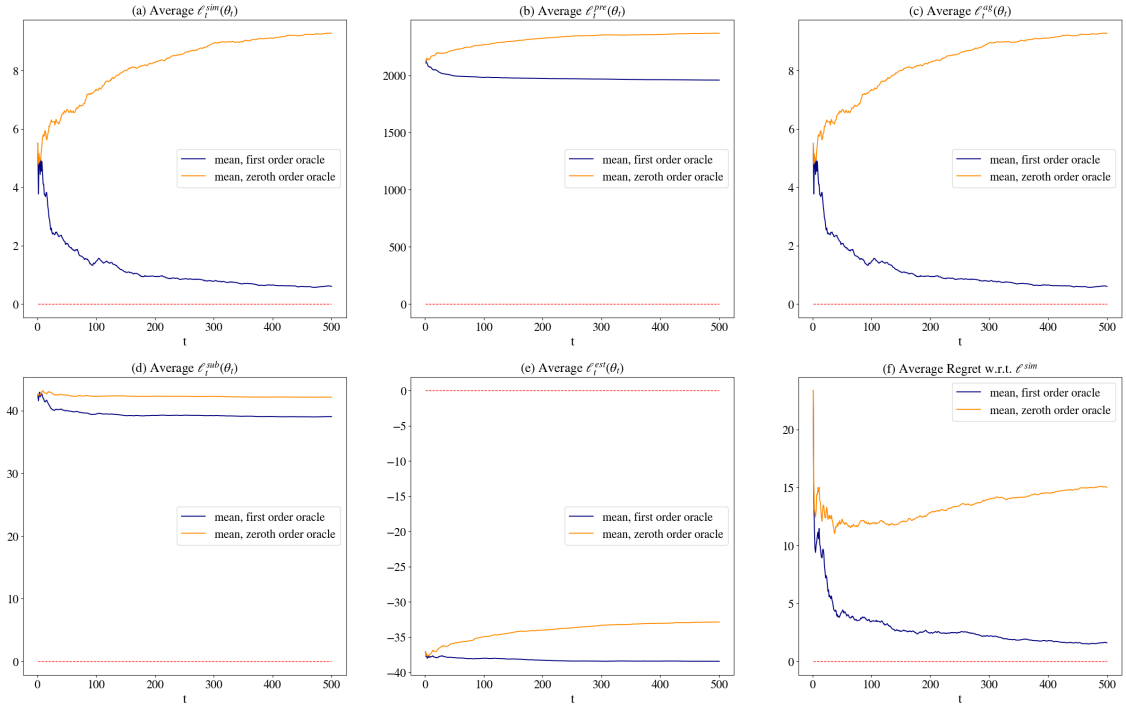

Figure 15: Learning a CES utility function under large noises: means of average losses with respect to different loss functions and means of average regret with respect to ℓ^{sim} over $T = 500$ iterations for equality constrained knapsack instances with $n = 50$.

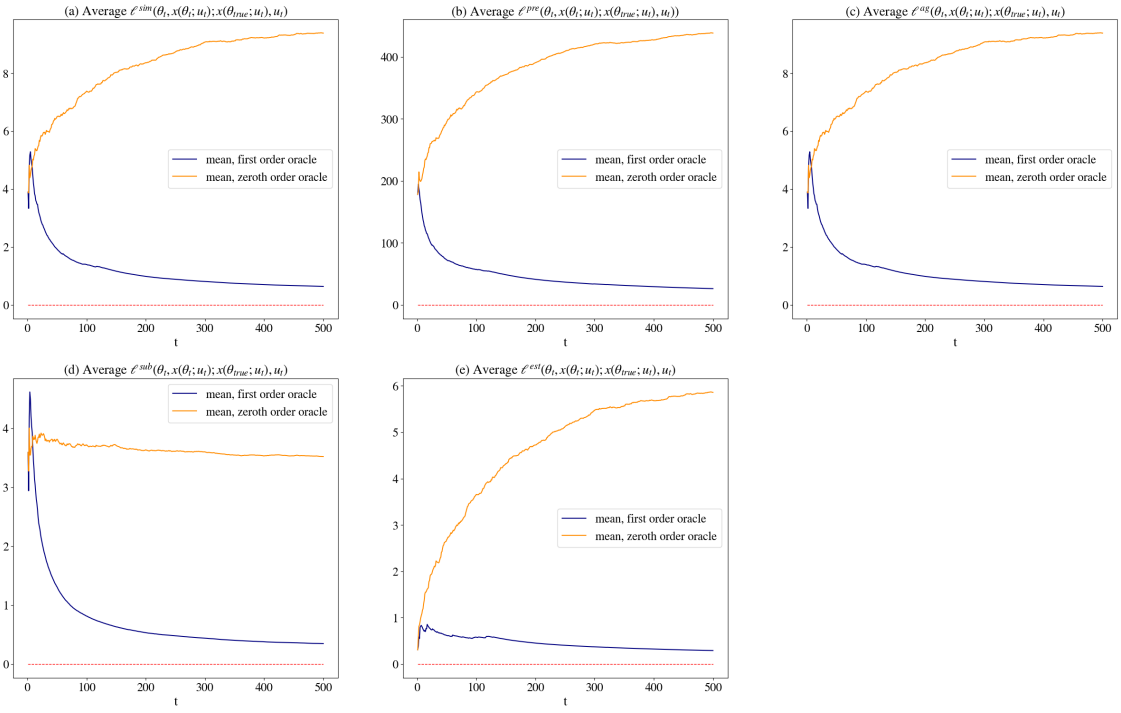

Figure 16: Learning a CES utility function under large noises: means of average losses with respect to different loss functions measured at $x(\theta_{true}; u_t)$ over $T = 500$ iterations for equality constrained knapsack instances with $n = 50$.