Joint Online Learning and Decision-making via Dual Mirror Descent

Alfonso Lobos¹, Paul Grigas¹, Zheng Wen²

¹University of California, Berkeley  
²Google DeepMind, Mountain view, California

April 21, 2021

Abstract

We consider an online revenue maximization problem over a finite time horizon subject to lower and upper bounds on cost. At each period, an agent receives a context vector sampled i.i.d. from an unknown distribution and needs to make a decision adaptively. The revenue and cost functions depend on the context vector as well as some fixed but possibly unknown parameter vector to be learned. We propose a novel offline benchmark and a new algorithm that mixes an online dual mirror descent scheme with a generic parameter learning process. When the parameter vector is known, we demonstrate an $O(\sqrt{T})$ regret result as well an $O(\sqrt{T})$ bound on the possible constraint violations. When the parameter is not known and must be learned, we demonstrate that the regret and constraint violations are the sums of the previous $O(\sqrt{T})$ terms plus terms that directly depend on the convergence of the learning process.

1 Introduction

We consider an online revenue maximization problem over a finite time horizon, subject to multiple lower and upper bound constraints. At each time period, an agent receives a context vector and needs to make a real-time decision. After making a decision, the agent earns some revenue and also incurs multiple costs, which may alternatively be interpreted as the consumption of multiple resources. Unlike the typical situation in online optimization and learning (see, e.g., (Hazan, 2019)), the agent has estimates of the revenue and cost functions available before making a decision. These estimates are updated sequentially via an exogenous learning process. Thus, there are three major challenges in this online learning and decision-making environment: (i) balancing the trade-off between revenue earned today and ensuring that we do not incur too many costs too early, (ii) ensuring that enough costs are incurred to meet the lower bound constraints over the full time horizon, and (iii) understanding the effects of the parameter learning process.

Examples of this online learning and decision-making setup occur in revenue management, online advertising, and online recommendation. In revenue management, pricing and allocation decisions for goods and services with a limited supply need to be made in real-time as customer arrivals occur (Bertsimas and Popescu, 2003; Boyd and Bilegan, 2003). This setup is also prevalent in online advertising, for example, in the case of a budget-constrained advertiser who bids in real-time.
auctions in order to acquire valuable impressions. Importantly, each arrival typically has associated a feature vector to it, for example, the cookie history of a user to which an ad can be shown. How that feature may relate to useful quantities, e.g., the probability of a user clicking an ad, may need to be learned. Finally, our setting considers lower bounds on cost since in many industries minimum production or marketing goals are desired.

1.1 Contributions

Our contributions may be summarized as follows:

1. We propose a novel family of algorithms to tackle a joint online learning and decision making problem. Our setting considers both lower and upper bound constraints on cost functions and does not require strong assumptions over the revenue and cost functions used, such as convexity. Our work can be understood as an extension of an online optimization problem in which we may also need to learn a generic parameter. Furthermore, our work can be considered as in a 1-lookup ahead setting as the agent can observe the current context vector before taking a decision.

2. We propose a novel benchmark to compare the regret of our algorithm. Our benchmark is considerably stricter in comparison to the expected best optimal solution in hindsight. Our benchmark is specially well suited to handle settings with “infeasible sequence of context vector arrivals” for which it is impossible to satisfy the cost constraints. We construct a dual problem which upper bounds the benchmark and we demonstrate how to efficiently obtain stochastic subgradients for it.

3. In the case when no “generic parameter learning” is needed, we prove that the regret of our algorithm is upper bounded by $O(\sqrt{T})$ under a Slater condition. Given the generic setup of our problem, this is a contribution on the field of online optimization. In the general case, our regret decomposes between terms upper bounded by $O(\sqrt{T})$ and terms coming from the convergence of the generic parameter learning.

4. We prove that the solution given by our algorithm may violate any given lower bound constraint by at most $O(\sqrt{T})$ in the online optimization case, while upper bounds are always satisfied by construction. Therefore, our methodology is asymptotically feasible in the online optimization case (Liakopoulos et al., 2019).

5. We demonstrate that our algorithm is effective and robust as compared to a heuristic approach in a bidding and allocation problem with no generic parameter learning in online advertising. Additionally, we study the effects of different generic parameter learning strategies in a linear contextual bandits problem with bounds on the number of actions taken.

1.2 Related Work

The problem of online revenue maximization under feasibility constraints has been mostly studied under the lens of online convex optimization (Hazan, 2019). While first studied on resource allocation problems under linear constraints (Mehta et al., 2007; Devanur et al., 2011), arbitrary convex revenue and cost functions are used today. Of major importance is the nature of the data arrivals. Typically, data has been assumed to be received in an adversarial (Devanur et al., 2011; Chen et al., 2017) or an i.i.d. manner (Wei et al., 2020; Balseiro et al., 2020b), with the data being...
sampled from an unknown distribution in the latter case. Subgradient methods based on primal-dual schemes have gained attraction \cite{Devanur et al., 2011; Jenatton et al., 2016; Chen et al., 2017; Yuan and Lamperski, 2018} as they avoid taking expensive projection iterations by penalizing the constraints through duality (either Lagrangian or Fenchel). Consequently, it is important to study both regret and the worst possible constraint violation level.

In the adversarial setting, regret is typically measured against the best-static decision in hindsight and algorithms achieving $O(\sqrt{T})$ regret, which is optimal in the adversarial setting, and different level of constraint violations levels have been achieved \cite{Mahdavi et al., 2012; Jenatton et al., 2016; Chen et al., 2017; Yuan and Lamperski, 2018}. On the i.i.d. setting and under linear constraints, \cite{Balseiro et al., 2020b} obtains an $O(\sqrt{T})$ regret bound and no constraint violation by algorithm construction (since they consider linear constraints with no lower bounds). Since they consider a 1-lookup ahead setting with i.i.d. arrivals, \cite{Balseiro et al., 2020b} use the best dynamic solution in hindsight as a benchmark, which is a considerably stricter benchmark than the commonly used best static solution. Our joint online learning and optimization model and algorithmic strategy builds upon the online optimization model and dual Mirror Descent approach for resource allocation presented by \cite{Balseiro et al., 2020b}. Note that our first contribution, the incorporation of arbitrary revenue and cost functions, was simultaneously obtained by the same set of authors on \cite{Balseiro et al., 2020a}.

A stream of literature studying a similar problem to ours is “Bandits with Knapsacks” (BwK) and extensions of it. In BwK, an agent operates over $T$ periods of time. At each period, the agent chooses an action, also known as an arm, from a finite set of possible action and observes a reward and a cost vector. As us, the agent would like to satisfy global cost constraints. BwK is studied both in an adversarial and i.i.d. settings, but here we only emphasize on the latter (see \cite{Immorlica et al., 2019} for the adversarial case). Assuming concave reward functions, \cite{Agrawal and Devanur, 2014} proposes an Upper-Confidence Bound type of algorithms which achieves sublinear rates of regret (2019) for the adversarial case). Assuming concave reward functions, \cite{Agrawal and Devanur, 2014} proposes an Upper-Confidence Bound type of algorithms which achieves sublinear rates of regret and constraint violations. \cite{Badanidiyuru et al., 2018} proposes a primal-dual algorithm to solve BwK with has a sublinear regret, and by algorithm construction, their cost constraints are always satisfied. Our job extends on this literature stream in the following ways. 1. We allow an arbitrary action space and reward and cost functions. 2. Our proposed benchmark is stricter than the best expected dynamic policy. 3. The novel joint learning and decision-making setting proposed here.

### 1.3 Notation

We use $\mathbb{R}_+^N := \{x \geq 0 : x \in \mathbb{R}^N\}$, $\mathbb{R}^N := \{x \leq 0 : x \in \mathbb{R}^N\}$, and $[N] := \{1, \ldots, N\}$ with $N$ being any integer. For any $x \in \mathbb{R}^N$ and $y \in \mathbb{R}^N$, $x \odot y := (x_1 y_1, \ldots, x_N y_N)$ and $x^T y := \sum_{i=1}^N x_i y_i$ representing the element-wise and dot products between vectors of same dimension. We use $x \in A$ to represent that $x$ belongs to set $A$, and $(x^1, \ldots, x^N) \in A_1 \times \cdots \times A_N$ represents $x^i \in A_i$ for all $i \in [n]$. We reserve capital calligraphic letters to denote sets. For any $x \in \mathbb{R}^N$, $[x]_+ := (\max\{x_1, 0\}, \ldots, \max\{x_N, 0\})$ and $1(x \in A) := 1$ if $x \in A$ and 0 otherwise. We use $\|\cdot\|$ to represent a norm operator, and in particular, for any $x \in \mathbb{R}^N$ we use $\|x\|_1 := \sum_{i=1}^N |x_i|$, $\|x\|_2 := \sqrt{\sum_{i=1}^N x_i^2}$, and $\|x\|_\infty = \max_{i \in [N]} |x_i|$. For any real-valued convex function $f : \mathcal{X} \to \mathbb{R}$, we say that $g$ is a subgradient of $f(\cdot)$ at $x \in \mathcal{X}$ if $f(y) \geq f(x) + g^T (y - x)$ holds for all $y \in \mathcal{X}$, and use $\partial f(x)$ to denote the set of subgradients of $f(\cdot)$ at $x$.  

3
2 Preliminaries and Algorithm

We are interested in a real-time decision-making problem over a time horizon of length $T$ involving three objects: (i) $z^t \in Z \subseteq \mathbb{R}^d$, the decision to be made at time $t$, (ii) $\theta^* \in \Theta \subseteq \mathbb{R}^p$, a possibly unknown parameter vector describing the revenue and cost functions that may need to be learned, and (iii) $w^t \in W \subseteq \mathbb{R}^m$, a context vector received at prior to making a decision at time $t$. These three objects describe the revenue and cost functions that are central to the online decision-making problem. In particular, let $f(\cdot; \cdot): Z \times \Theta \times W \rightarrow \mathbb{R}$ denote the revenue function and let $c(\cdot; \cdot, \cdot): Z \times \Theta \times W \rightarrow \mathbb{R}^K$ denote the collection of $K$ different cost functions. We assume that these functions are bounded, namely for the true revenue function it holds that $\sup_{z \in Z, w \in W} f(z; \theta^*, w) \leq \bar{f}$ with $\bar{f} > 0$ and for the cost functions it holds that $\sup_{z \in Z, \theta \in \Theta, w \in W} \|c(z; \theta, w)\|_\infty \leq \bar{C}$ with $\bar{C} > 0$.

At each time period $t$, first $w^t$ is revealed to the decision maker and is assumed to be drawn i.i.d from an unknown distribution $P$ over $W$. For example, if $W$ is a finite set, then $w^t$ could represent the scenario being revealed at time $t$. We assume that once the decision maker observes a context vector $w^t \in W$, then it also observes or otherwise have knowledge of the parametric forms of revenue and cost functions $f(\cdot; w^t): Z \times \Theta \rightarrow \mathbb{R}$ and $c(\cdot, \cdot; w^t): Z \times \Theta \rightarrow \mathbb{R}^K$. Although the true parameter $\theta^*$ may be unknown to the decision maker at time $t$, whenever a decision $z^t \in Z$ is made the revenue earned is equal to $f(z^t, \theta^*, w^t)$ and the vector of cost values incurred is equal to $c(z^t, \theta^*, w^t)$.

In an ideal but unrealistic situation, the decision planner would be able to observe the sequence $(w^1, \ldots, w^T)$ of future context vector arrivals and would set the decision sequence $(z^1, \ldots, z^T)$ by solving the full observability (or hindsight) problem:

$$
(O) : \max_{(z^1, \ldots, z^T) \in \mathcal{Z}^T} \sum_{t=1}^{T} f(z^t; \theta^*, w^t) \\
\text{s.t. } T \alpha \odot b \leq \sum_{t=1}^{T} c(z^t; \theta^*, w^t) \leq Tb
$$

where $b \in \mathbb{R}^K_{++}$, and $\alpha \in [-1, 1]^K \cup \{-\infty\}$ with $\alpha_k = -\infty$ meaning that no lower bounds are present for coordinate $k$. Define $\underline{b} := \min_{k \in [K]} b_k$ and $\bar{b} := \max_{k \in [K]} b_k$, and we assume that $\bar{b} > 0$. The vector $b$ can be thought as a resource or budget vector proportional to each period. Then, (O) is a revenue maximization problem over the time horizon $T$ with lower and upper cost constraints. Setting $-1$ as the lower bound for $\alpha_k$ for all $k \in [K]$ is an arbitrary choice only affecting some of the constants in the regret bounds we prove.

Before providing more details on the dynamics of the problem and our proposed algorithm, we introduce a novel benchmark to evaluate the performance/regret of our algorithm. The primary need for a new benchmark in our context is that the generality of our problem leads to feasibility issues. Indeed, for some combinations of context vector arrivals, problem (O) may be infeasible due to the presence of both lower and upper bound constraints as well as the fact that the costs functions are generic. We now define an offline benchmark as follows. A natural benchmark to consider is the expected optimal value of (O). However, as long as there is any positive probability of (O) being infeasible, then this benchmark will be $-\infty$, which will lead to trivial regret bounds. Thus, to avoid such trivialities, we consider a benchmark that interpolates between the expected optimal value of
and a deterministic problem that replaces the random revenue and cost functions with their expected values. In particular, let \( \gamma \in [0, 1] \) denote this interpolation parameter. For any \( z \in \mathcal{Z} \), \( \theta \in \Theta \), \( w' \in \mathcal{W} \), \( w \sim \mathcal{P} \), and \( \gamma \in [0, 1] \) we define:

\[
\text{rev}(z; \theta, w', \gamma) := (1 - \gamma) f(z; \theta, w') + \gamma \mathbb{E}_{\mathcal{P}}[f(z; \theta, w)] \\
\text{cost}(z; \theta, w', \gamma) := (1 - \gamma) c(z; \theta, w') + \gamma \mathbb{E}_{\mathcal{P}}[c(z; \theta, w)].
\]

Let \( \mathcal{P}^T := \mathcal{P} \times \cdots \times \mathcal{P} \) denote a product distribution of length \( T \), i.e., the distribution of \((w^1, \ldots, w^T)\). Now, for any \( \gamma \in [0, 1] \), let us define

\[
\text{OPT}(\mathcal{P}, \gamma) := \mathbb{E}_{\mathcal{P}^T} \left[ \max_{z^t \in \mathcal{Z}: t \in [T]} \sum_{t=1}^T \text{rev}(z^t; \theta^t, w^t, \gamma) \\
\text{s.t. } T \alpha \odot b \leq \sum_{t=1}^T \text{cost}(z^t; \theta^t, w^t, \gamma) \leq Tb \right]
\]

and let us further define

\[
\text{OPT}(\mathcal{P}) := \max_{\gamma \in [0, 1]} \text{OPT}(\mathcal{P}, \gamma).
\]

Note that \( \text{OPT}(\mathcal{P}, 0) \) is exactly the expected optimal value of the hindsight problem \(\Pi\). On the other hand, \( \text{OPT}(\mathcal{P}, 1) \) corresponds to a deterministic approximation of \(\Pi\) that replaces all random quantities with their expectations and is typically a feasible problem. Then, we can understand \( \gamma \in [0, 1] \) as an interpolation parameter between the more difficult hindsight problem \(\text{OPT}(\mathcal{P}, 0)\) and the expectation problem \(\text{OPT}(\mathcal{P}, 1)\). Importantly, the benchmark we consider is \(\text{OPT}(\mathcal{P})\), which considers the worst case between these two extremes. It is possible to have \(\text{OPT}(\mathcal{P}) = \text{OPT}(\mathcal{P}, 0)\), \(\text{OPT}(\mathcal{P}) = \text{OPT}(\mathcal{P}, 1)\), \(\text{OPT}(\mathcal{P}) = \text{OPT}(\mathcal{P}, \gamma)\) for some \(\gamma \in (0, 1)\), and \(\text{OPT}(\mathcal{P}) = -\infty\). It is also possible to have a unique \(\gamma\) that maximizes \(\text{OPT}(\mathcal{P}, \gamma)\) as well as infinitely many such maximizers. Examples of all of these possibilities are included in the supplementary materials.

### 2.1 Joint Learning and Decision-making Dynamics and Regret Definition

Now we describe the dynamics of our joint online learning and decision-making problem as well as a generic “algorithmic scheme.” In Section 2.2 we give a complete algorithm after building up the machinery of dual mirror descent. Let \( I^t := (z^t, \theta^t, w^t, f^t(z^t; \theta^t, w^t), c(z^t; \theta^t, w^t)) \) denote the information obtained during period \( t \), and let \( \mathcal{H}^t := (I^1, \ldots, I^T) \) denote the complete history up until the end of period \( t \). Note that it is assumed that the decision planner observes the exact incurred cost value vector \( c(z^t; \theta^t, w^t) \), but there is a possibility of including additional randomness in the observed revenue. In particular, the observed revenue \( f^t(z^t; \theta^t, w^t) \) satisfies \( f^t(z^t; \theta^t, w^t) = f(z^t; \theta^t, w^t) + \xi_t \) where \( \xi_t \) is a mean zero random variable that is allowed to depend on \( w^t \) but is independent of everything else.

Let \( A_\theta \) refer to a generic learning algorithm and let \( A_z \) refer to a generic decision-making algorithm. Then, at any time period \( t \), the decision planner sets

\[
\theta^t = A_\theta(\mathcal{H}^{t-1}) \\
z^t = A_z(f(\cdot; \theta^t, w^t), c(\cdot; \theta^t, w^t), \mathcal{H}^{t-1})
\]

We refer to \((A_z, A_\theta)\) as \( A \) when no confusion is possible. Note that an important special case is when \( A_\theta \) outputs \( \theta^* \) for all inputs, which is the case where \( \theta^* \) is known. Algorithm \(\Pi\), which alternates between an online learning step using \( A_\theta \) and an online decision-making step using \( A_z \), specifies the
precise sequence of events when using the generic algorithm $A$. Recall that $	ilde{C} := \sup_{(z;\theta,w)\in\mathcal{Z}\times\Theta\times\mathcal{W}} \|c(z;\theta,w)\|_\infty$, which is a constant that we will use as the minimum allowable remaining cost budget. For simplicity we assume that the constant $\tilde{C}$ is available although we can easily replace it with an available upper bound.

**Algorithm 1** Generic Online Learning and Decision-making Algorithmic Scheme

**Input:** Initial estimate $\theta^1 \in \Theta$, and remaining cost budget vector $b^1 \leftarrow T b$. 

for $t = 1, \ldots, T$ do 

1. Update $\theta^t \leftarrow A_0(H^{t-1})$. 
2. Receive $w^t \in \mathcal{W}$, which is assumed to be drawn from an unknown distribution $\mathcal{P}$ and is independent of $\mathcal{H}^{t-1}$. 
3. Set $z^t \leftarrow A_z(f(\cdot;\theta^t, w^t), c(\cdot;\theta^t, w^t), \mathcal{H}^{t-1})$. 
4. Update remaining cost budget $b^{t+1} \leftarrow b^t - c(z^t; \theta^*, w^t)$, and earn revenue $f^t(z^t; \theta^*, w^t)$. 
5. If $b^{t+1}_k < \tilde{C}$ for any $k \in [K]$, break. 
end for

Note that Steps 4. and 5. of Algorithm 1 ensure that the total cost incurred is always less than or equal to $bT$, which ensures that the upper bound constraints in (1) are always satisfied, while there is a chance that some lower bound constraints may not be satisfied. These steps make our later theoretical analysis simpler, but less conservative approaches can be used, for example allowing the algorithm to exceed $bT$ once.

Define $R(A|\mathcal{P}) = \mathbb{E}_{\mathcal{P}^T} \left[ \sum_{t=1}^T f(z^t; \theta^*, w^t) \right]$ as the expected revenue of algorithm $A$ over distribution $\mathcal{P}^T$, where $z^t$ is computed as in (3). We define the regret of algorithm $A$ as $\text{Regret}(A|\mathcal{P}) := \text{OPT}(\mathcal{P}) - R(A|\mathcal{P})$. Since the probability distribution $\mathcal{P}$ is unknown to the decision maker, our goal is to design an algorithm $A$ that works well for any distribution $\mathcal{P}$. That is, we would like to obtain a good distribution free regret bound.

### 2.2 Dual Problem and Dual Mirror Descent Algorithm

We now consider a Lagrangian dual approach that will naturally lead to a dual mirror descent algorithm. Let $\lambda \in \mathbb{R}^K$ denote a vector of dual variables, and we define the set of feasible dual variables as $\Lambda := \{\lambda \in \mathbb{R}^K : \lambda_k \geq 0 \text{ for all } k \text{ with } \alpha_k = -\infty\}$. For any triplet $(\lambda, \theta, w) \in \Lambda \times \Theta \times \mathcal{W}$ define

$$
\varphi(\lambda; \theta, w) := \max_{z \in \mathcal{Z}} f(z; \theta, w) - \lambda^T c(z; \theta, w)
$$

and for any $(\lambda, \theta) \in \Lambda \times \Theta$ define

$$
p(\lambda) := \sum_{k \in [K]} b_k([\lambda_k]_+ - \alpha_k[-\lambda_k]_+)
$$

$$
D(\lambda; \theta) := \mathbb{E}_P[\varphi(\lambda; \theta, w)] + p(\lambda).
$$

This works assumes that $z(\lambda; \theta, w)$ exists and can be efficiently computed for any $(\lambda, \theta, w) \in (\Lambda, \Theta, \mathcal{W})$. Furthermore, in case there are multiple optimal solutions corresponding to $\varphi(\lambda; \theta, w)$...
we assume that the subroutine for computing \( z(\lambda; \theta, w) \) breaks ties in a deterministic manner. We call \( D(\cdot; \theta) \) the dual function given parameters \( \theta \), which is a key component of the analysis and algorithms proposed in this work. In particular, we first demonstrate in Proposition 1 that \( D(\cdot; \theta^*) \) can be used to obtain an upper bound on \( \text{OPT}(P) \).

**Proposition 1.** For any \( \lambda \in \Lambda \), it holds that \( \text{OPT}(P) \leq TD(\lambda; \theta^*) \).

Next, Proposition 2 demonstrates that a stochastic estimate of a subgradient of \( D(\cdot; \theta) \) can be easily obtained during the sequence of events described in Algorithm 1.

**Proposition 2.** Let \( \lambda \in \Lambda, \theta \in \Theta, \) and \( w \in W \) be given. Define \( \tilde{g}(\lambda; \theta, w) \in \mathbb{R}^K \) by \( \tilde{g}_k(\lambda; \theta, w) := -c_k(z(\lambda; \theta, w); \theta, w) + b_k(\mathbb{1}(\lambda_k \geq 0) + \alpha_k(\mathbb{1}(\lambda_k < 0)) \) for all \( k \in [K] \). Then, if \( w \sim P \), it holds that \( \tilde{g}(\lambda; \theta, w) \) is a stochastic subgradient estimate of \( D(\cdot; \theta) \) at \( \lambda \), i.e., \( \mathbb{E}_P[\tilde{g}(\lambda; \theta, w)] \in \partial_\lambda D(\lambda; \theta) \).

We are now ready to describe our dual mirror descent algorithm. Let \( h(\cdot) : \Lambda \rightarrow \mathbb{R} \) be the reference function for mirror descent, which we assume is \( \sigma_1 \)-strongly convex in the \( \ell_1 \)-norm, i.e., for some \( \sigma_1 > 0 \) it holds that \( h(\lambda) \geq h(\lambda') + \langle \nabla h(\lambda'), \lambda - \lambda' \rangle + \frac{\sigma_1}{2} \| \lambda - \lambda' \|^2_1 \) for any \( \lambda, \lambda' \in \Lambda \). Also, we assume that \( h(\cdot) \) is a separable function across components, i.e., it satisfies \( h(\lambda) = \sum_{k=1}^K h_k(\lambda_k) \) where \( h_k(\cdot) : \mathbb{R} \rightarrow \mathbb{R} \) is a convex univariate function for all \( k \in [K] \). Define \( V_k(\lambda, \lambda') := h(\lambda) - h(\lambda') - \nabla h(\lambda')^T (\lambda - \lambda') \), the Bregman divergence using \( h(\cdot) \) as the reference function.

Algorithm 2 presents the main algorithm of this work. Algorithm 2 is a specific instance of the more general algorithmic scheme, presented in Algorithm 1 where we fill in the generic decision making subroutine \( A_z \) with a dual stochastic mirror descent (Hazan 2019; Beck and Teboulle 2003) step with respect to the current estimate of the dual problem \( \min_{\lambda \in \Lambda} D(\lambda; \theta^t) \). Note that the learning subroutine \( A_\theta \) is left as a generic subroutine; the regret bounds that we prove in Section 3 hold for any learning algorithm \( A_\theta \) and naturally get better when \( A_\theta \) has better convergence properties.

Note that Proposition 2 ensures that \( \tilde{g}^t \) from Step 6 of Algorithm 2 is a stochastic subgradient of \( D(\cdot; \theta^t) \) at \( \lambda^t \). The specific form of the mirror descent step in Step 7. depends on the reference function \( h(\cdot) \) that is used. A standard example is the Euclidean reference function, i.e., \( h(\cdot) := \frac{1}{2} \| \cdot \|^2_2 \); in which case Step 7. is a projected stochastic subgradient descent step. Namely, \( \lambda_k^{t+1} \leftarrow [\lambda_k^t - \eta \tilde{g}_k^t]_+ \) for all \( k \in [K] \) with \( \alpha_k = -\infty \) and \( \lambda_k^{t+1} \leftarrow \lambda_k^t - \eta \tilde{g}_k^t \) otherwise. A simple extension of this example is \( h(\lambda) := \lambda^T Q \lambda \) for some positive definite matrix \( Q \). When no lower bounds are present, i.e., \( \alpha_k = -\infty \) for all \( k \in [K] \), we can use an entropy-like reference function \( h(\lambda) := \sum_{k \in [K]} \lambda_k \log(\lambda_k) \) wherein Step 7. becomes a multiplicative weight update \( \lambda_k^t \leftarrow \lambda^t \exp(-\eta \tilde{g}_k^t) \) (Arora et al. 2012). Finally, note that since the reference function is component wise separable, one may use a different type of univariate reference function for different components.

While Algorithm 2 fills in the gap for \( A_z \) using mirror descent, the learning algorithm \( A_\theta \) in Step 1. is still left as generic and there are a range of possibilities that one might consider depending on the specific problem being addressed. For example, considering only the revenue function for simplicity, suppose that there is a feature map \( f' : Z \times W \rightarrow \mathbb{R}^p \) such that \( f(z; \theta, w) = f'(z; \theta)w \) for \( (z, \theta, w) \in Z \times \Theta \times W \) and we observe both \( f(z; \theta^*, w^t) \) and \( f'(z^t; w^t) \) at time \( t \). Then, one could use \( (f'(z^t; \theta^*, w^t), f(z^t; w^t))_{t=1}^{T-1} \) to fit a linear model (possibly with regularization) for implementing \( A_\theta \) at time \( t \). Depending on the underlying structure of the problem and randomness of the data arrivals, the previous methods may not converge to \( \theta^* \). Different ways of applying
Algorithm 2 Online Learning and Decision-making via Dual Mirror Descent

\textbf{Input:} Initial estimate $\theta^1 \in \Theta$, remaining cost budget vector $b^1 = Tb$, and initial dual solution $\lambda^1$.

\begin{algorithmic}
\State for $t = 1, \ldots, T$ do
\State 1. Update $\theta^t \leftarrow A_\theta (\mathcal{H}^{t-1})$.
\State 2. Receive $w^t \in \mathcal{W}$, which is assumed to be drawn from an unknown distribution $\mathcal{P}$ and is independent of $\mathcal{H}^{t-1}$.
\State 3. Make primal decision $z^t \leftarrow z(\lambda^t; \theta^t, w^t)$, i.e.,
\State \hspace{1em} $z^t \in \arg \max_{z \in \mathcal{Z}} f(z; \theta^t, w^t) - (\lambda^t)^T c(z; \theta^t, w^t)$.
\State 4. Update remaining cost budget $b^{t+1} \leftarrow b^t - c(z^t; \theta^*, w^t)$, and earn revenue $f^t(z^t; \theta^*, w^t)$.
\State 5. If $b^{t+1}_k < \bar{C}$ for any $k \in [K]$, break.
\State 6. Obtain dual stochastic subgradient $\tilde{g}^t_k$ where $\tilde{g}^t_k \leftarrow -c_k(z^t; \theta^t, w^t) + b_k (\mathbb{I}(\lambda_k \geq 0) + \alpha_k \mathbb{I}(\lambda_k < 0))$ for all $k \in [K]$.
\State 7. Choose “step-size” $\eta_t$ and take dual mirror descent step
\State \hspace{1em} $\lambda^{t+1} \leftarrow \arg \min_{\lambda \in \Lambda} \lambda^T \tilde{g}^t + \frac{1}{\eta_t} V_h(\lambda, \lambda^t)$.
\State \end{algorithmic}

Step 1. are shown for a linear contextual bandits problem in Section 4. The performance of the different implementations vary drastically depending on the underlying randomness of the data arrivals.

3 Regret Bound and Related Results

In this section, we present our main theoretical result, Theorem 1 which shows regret bounds for Algorithm 2. In particular, the regret of Algorithm 2 can be decomposed as the summation of two parts: (i) the terms that appear when $\theta^*$ is known, which emerge from the properties of the Mirror Descent algorithm and can be bounded sublinearly as $\mathcal{O}(\sqrt{T})$, and (ii) terms that naturally depend on the convergence of the learning process towards $\theta^*$. We also discuss the proof strategy for Theorem 1. Finally, for each lower bound constraint in (1), we prove that our algorithm may violate this lower bound by at most $\mathcal{O}(\sqrt{T})$ plus terms that depend on how $\theta^t$ converges to $\theta^*$.

3.1 Regret Bound

Before presenting our main theorem, we need to establish a few more ingredients of the regret bound. First, we present Assumption 3.1 which can be thought of as a boundedness assumption on the dual iterates.

Assumption 3.1 (Bounded Dual Iterates). There is an absolute constant $C_h > 0$ such that the dual iterates $\{\lambda^t\}$ of Algorithm 2 satisfy $\mathbb{E} \left[ \|\nabla h(\lambda^t)\|_\infty \right] \leq C_h$ for all $t \in [T]$.
Note that, in the Euclidean case where \( h(\lambda) = \frac{1}{2} \| \lambda \|^2 \), we have \( \nabla h(\lambda) = \lambda \) and therefore Assumption 3.1 may be thought of as a type of boundedness condition. After stating our regret bound, we present a sufficient condition for Assumption 3.1 which involves only the properties of the problem and not the iterate sequence of the algorithm.

Now, recall that \( \mathcal{H}^t \) can be understood as all the information obtained by Algorithm 2 up to period \( t \). Then, Step 4. of Algorithm 2 is intrinsically related to the following stopping time with respect to \( \mathcal{H}^{t-1} \).

**Definition 3.1** (Stopping time). Define \( \tau_A \) as the minimum between \( T \) and the smallest time \( t \) such that there exists \( k \in [K] \) with \( \sum_{t=1}^{\tau_A} c_k(z^t; \theta^t, w^t) + C > b_k T \).

Finally, recall that we defined constants \( \bar{f} > 0, \bar{C} > 0, \bar{b} > 0 \) and \( \bar{\delta} > 0 \) such that \( \sup_{z \in Z, \theta \in \Theta, w \in W} | c(z; \theta, w) | \| \infty \leq \bar{C}, \bar{b} := \min_{k \in [K]} b_k \) and \( \bar{\delta} := \max_{k \in [K]} b_k \). Also, \( \sigma_1 \) refers to the strong convexity constant of \( h(\cdot) \). We are now ready to state Theorem 1 which presents our main regret bound.

**Theorem 1.** Let \( A \) denote Algorithm 2 with a constant “step-size” rule \( \eta_t \leftarrow \eta \) for all \( t \geq 1 \) where \( \eta > 0 \). Suppose that Assumption 3.1 holds. Then, for any distribution \( \mathcal{P} \) over \( w \in W \), it holds that \( \text{Regret}(A|\mathcal{P}) \leq \Delta_{\text{DM}} + \Delta_{\text{Learn}} \) where

\[
\Delta_{\text{DM}} := \frac{2(\bar{C}^2 + \bar{b}^2)}{\sigma_1} \eta \mathbb{E}[\tau_A] + \frac{1}{\eta} V_h(0, \lambda^1) + \frac{\bar{f}}{\bar{b}} \left( \bar{C} + \frac{\| \nabla h(\lambda^1) \|_\infty}{\eta} \right)
\]

\[
\Delta_{\text{Learn}} := \mathbb{E} \left[ \sum_{t=1}^{\tau_A} (c(z^t; \theta^*, w^t) - c(z^t; \theta^t, w^t)) \bar{\lambda}^t \right] + \frac{\bar{f}}{\bar{b}} \mathbb{E} \left[ \sum_{t=1}^{\tau_A} (c(z^t; \theta^*, w^t) - c(z^t; \theta^t, w^t)) \right]_\infty.
\]

Theorem 1 states that the regret of Algorithm 2 can be upper bounded by the sum of two terms: (i) a quantity \( \Delta_{\text{DM}} \) that relates to the properties of the decision-making algorithm, dual mirror descent, and (ii) a quantity \( \Delta_{\text{Learn}} \) that relates to the convergence of the learning algorithm \( A_\eta \). It is straightforward to see that setting \( \eta \leftarrow \gamma/\sqrt{T} \) for some constant parameter \( \gamma > 0 \) implies that \( \Delta_{\text{DM}} \) is \( O(\sqrt{T}) \). In the pure online optimization case, \( \theta^* \) is known and hence \( \theta^t = \theta^* \) for all \( t \in [T] \) yielding \( \Delta_{\text{Learn}} = 0 \). Thus, using \( \eta \leftarrow \gamma/\sqrt{T} \) in the pure online optimization case yields \( \text{Regret}(A|\mathcal{P}) \leq O(\sqrt{T}) \) and extends results presented by Balseiro et al. (2020b). More generally, \( \Delta_{\text{Learn}} \) depends on the convergence of \( \theta^t \) to \( \theta^* \). Under a stricter version of Assumption 3.1 and assuming the cost functions are Lipschitz in \( \theta \), we demonstrate in the supplementary materials that \( \Delta_{\text{Learn}} \) is \( O(\mathbb{E} \left[ \sum_{t=1}^{\tau_A} \| \theta^t - \theta^* \|_\theta \right]) \).

Let us now return to Assumption 3.1 and present a sufficient condition for this assumption that depends only on the structural properties of the problem and not directly on the iterations of the algorithm. The type of sufficient condition we consider is an extended Slater condition that requires both lower and upper bound cost constraints to be satisfied in expectation with positive slack for all \( \theta \in \Theta \). Let us first define precisely what the average slack is for a given \( \theta \in \Theta \).

**Definition 3.2.** For a given \( \theta \in \Theta \), we define its slack \( \delta_\theta \in \mathbb{R} \) as \( \delta_\theta := \mathbb{E}_P \left[ \max_{z \in Z} \text{res}(z; \theta, w) \right] \) with \( \text{res}(z; \theta, w) := \min \{ \| T b_k - c_k(z; \theta, w) \|_\infty, \| c_k(z; \theta, w) - T \alpha_k b_k \|_\infty \} \) for all \( (z, w) \in Z \times W \).

The following proposition uses the average slack to upper bound \( C_h \) in Assumption 3.1.
Proposition 3. Assume that there exists $\delta > 0$ such that $\delta_\theta \geq \delta$ for all $\theta \in \Theta$, and let $C^\circ := 2(\eta(C^2 + \bar{f}) + \bar{f})$. Suppose that we use the Euclidean reference function $h(\cdot) := \frac{1}{2}\|\cdot\|_2^2$, which corresponds to the traditional projected stochastic subgradient method. Then, it holds that $C_h \leq \max\{\|\lambda\|_\infty \sqrt{2/\sqrt{0.5(C^\circ / \bar{f})^2 + \eta C^\circ}\}.$

3.2 Proof Sketch and Cost Feasibility

The proof sketch for Theorem 1 is informative of how the algorithm works and therefore we outline it here. At a high level the proof consists of two major steps. First, we prove that the $E[\tau_A]$ is close to $T$ for the pure online optimization case. In the general case additional terms depending on how $\theta_t$ converges to $\theta^*$ appear. Second, we bound the expected regret up to period $\tau_A$. In particular, we prove $E[\tau_A D(\sum_{t=1}^{\tau_A} \lambda^t; \theta^*) - \sum_{t=1}^{\tau_A} f(z^t; \theta^*, w_t)]$ upper bounds the regret and is $O(\sqrt{T})$ in the pure online optimization case. Finally, the expected regret up to period $T$ is bounded by the sum of the expected regret up to period $\tau_A$ plus the trivial bound $fE[T - \tau_A]$. (Note that the two major steps of our proof mimic those of Balseiro et al. 2020[1], but the generality of our setting as well as the presence of parameter learning leads to new complications.)

A key element of the proof is that if we violate the upper cost constraints this occurs near the final period $T$ (as long as we ‘properly’ learn $\theta^*$). A solution obtained using Algorithm 2 can not overspend, but may underspend. Proposition 4 shows that the amount of underspending can again be bounded by the sum of terms that arise from the decision-making algorithm (mirror descent) and terms that depend on the convergence of the learning process. In the pure online optimization case, these lower constraint violations are bounded by $O(\sqrt{T})$ if we use $\eta = \gamma / \sqrt{T}$ with $\gamma > 0$ arbitrary. To put this result in context, even if constraint violations can occur their growth is considerably smaller than $T$, which is the rate at which the scale of the constraints in (1) grow. In the general case, terms depending on how $\theta^*$ converges to $\theta^*$ again appear, analogously to Theorem 1.

Proposition 4. Assume we run Algorithm 2 under Assumption 2 using $\eta_t = \eta$ for all $t \geq 1$. For any $k \in [K]$ with $\alpha_k \neq -\infty$ it holds:

$$T\alpha_k b_k - E[\sum_{t=1}^{\tau_A} c_k(z^t; \theta^*, w_t)] \leq \left(\frac{\|\nabla h(\lambda^1)\|_\infty + C_h}{\eta}\right) \frac{b + \alpha_k b_k}{b} + \frac{\alpha_k b_k C}{b} + \frac{\alpha_k b_k \|E[\sum_{t=1}^{\tau_A} c(z^t; \theta^*, w_t) - c(z^t; \theta^*, w^t)]\|_\infty}{b} + E[\sum_{t=1}^{\tau_A} c_k(z^t; \theta^*, w_t) - c_k(z^t; \theta^*, w^t)].$$

4 Experiments

This section describes the two experiments performed. The first models the problem of a centralized bidder entity bidding on behalf of several clients. Each client has both lower and upper bounds on their desired spending. This experiment uses data from the online advertising company Criteo (Diemert et al., 2017). The results show that our methodology spends the clients budgets (mostly) in their desired range, depleting their budgets close to the last period ($T$), and obtaining a higher
profit than a highly used heuristic. The second experiment is a linear contextual bandits problem with lower and upper bounds on the number of actions that can be taken. This experiment is illustrative of how different schemes to learn $\theta^*$, i.e., implementations of Step 1. of Algorithm 2 may be more or less effective depending on the inherent randomness of the data arrivals.

4.1 Centralized repeated bidding with budgets

Consider a centralized bidding entity, which we here call the bidder, who bids on behalf of $K \geq 1$ clients. The bidder can participate in at most $T \geq 1$ auctions which are assumed to use a second-price mechanism. In the case of winning an auction, the bidder can only assign the reward of the auction to at most one client at a time. At the beginning of each auction, the bidder observes a vector $w \in \mathcal{W}$ of features and a vector $r(w) \in \mathcal{R}_+^K$. Each coordinate of $r(w)$ represents the monetary amount the $k^{th}$ client offers the bidder for the auction reward. For each auction $t \in [T]$, call ‘$\text{mp}^t$’ to the highest bid from the other bidders. The goal of the bidder is to maximize its profit while satisfying its clients lower and upper spending bounds. Defining $\mathcal{X} := \{x \in \mathbb{R}_+^K : \sum_{i=1}^K x_i \leq 1\}$, the problem the bidder would like to solve is (special case of Problem (1)):

$$\max_{(z^t, x^t) \in \mathcal{R}_+ \times \mathcal{X}: t \in [T]} \sum_{t=1}^T \sum_{k=1}^K (r_k(w^t) - \text{mp}^t)x_k^t \mathbb{1}\{z^t \geq \text{mp}^t\}$$

s.t. $T \alpha \odot b \leq \sum_{t=1}^T r(w^t) \odot x^t \mathbb{1}\{z^t \geq \text{mp}^t\} \leq Tb.$

where $Tb$ represent the maximum the clients would like to spend, and $\alpha \in [0, 1)^K$ the minimum percentage to be spent. The pair $(z^t, x^t) \in \mathcal{R}_+ \times \Delta$ represents the submitted bid and the probabilistic allocation of the reward chosen by the bidder at period $t$ (we later show that our algorithm uses a binary allocation policy). We use $\mathbb{1}\{z^t \geq \text{mp}^t\}$ to indicate that the bidder wins the auction $t \in [T]$ only if its bid is higher than $\text{mp}^t$. Here we assume $r(\cdot): \mathcal{W} \rightarrow \mathbb{R}_+^K$ as known, but the extension to the case when we need to learn it is natural.

An important property of this problem is that we can implement our methodology without learning the distribution of $\text{mp}$, making this experiment fall in the pure online optimization case. The latter occurs as $\varphi(\lambda; (w, \text{mp})) = \max_{(z, x) \in \mathcal{R}_+ \times \mathcal{X}} \sum_{k=1}^K (r_k(w)(1 - \lambda_k) - \text{mp})x_k \mathbb{1}\{z \geq \text{mp}\}$ can be solved as Algorithm 3 shows.

Algorithm 3 Solving $\varphi(\cdot; \cdot, \cdot)$

**Input:** Pair $(\lambda, w) \in \mathcal{R}_+^K \times \mathcal{W}$, and reward vector $r(w)$.

1. Select $k^* \in \arg \max_{k \in [K]} r_k(w)(1 - \lambda_k)$.
2. If $r_{k^*}(w)(1 - \lambda_{k^*}) \geq 0$ set $z = r_{k^*}(w)(1 - \lambda_{k^*})$, $x_{k^*} = 1$ and $x_k = 0$ for all $k \in [K] \neq k^*$, otherwise choose $z = x_k = 0$ for all $k \in [K]$.

**Output:** $(z, x)$ optimal solution for $\varphi(\lambda; (w, \text{mp}))$.

Experiment Details. This experiment is based on data from Criteo (Diemert et al., 2017). Criteo is a Demand-Side Platform (DSP), which are entities who bid on behalf of hundreds or thousands of advertisers which set campaigns with them. The dataset contains millions of bidding logs during one
month of Criteo’s operation. In all these logs, Criteo successfully acquired ad-space for its clients through real-time second-price auctions (each log represents a different auction and ad-space). Each log contains information about the ad-space and user to which it was shown, the advertiser who created the ad, the price paid by Criteo for the ad-space, and if a conversion occurred or not (besides from other unused columns). The logs from the first three weeks were used as training data, the next two days as validation, and the last week as test.

The experiment was performed as follows. The user’s information and advertiser ids from the train data were used to train the neural network for conversion prediction from Pan et al. (2018). This prediction model was validated using the validation data. Once selected and saved the set of parameters with highest validation AUC, we use the predictions coming from this architecture as if they were the truthful probabilities of conversion. From the test data, we obtained total budgets to spend for each advertiser, assuming that all advertisers expect their budget to be spent at least by 95% ($\alpha_k = 0.95$ for all $k \in [K]$). To simulate a real operation, we read the test logs in order using batches of 128 logs (as updating a system at every arrival is not realistic). We use 100 simulations for statistical significance and use traditional subgradient descent on Step 7. of Algorithm 2 (more experimental details in the supplement).

Figure 1 shows that our methodology obtains a higher profit in comparison to the baseline. Also, almost all advertisers got their total spending on the feasible range (above 95% of their total target budget). In addition, advertisers tend to deplete their budgets close to the end of the simulations. Observe that few advertisers spent their budgets in average closer to the beginning rather than the simulations end. We found that those advertisers had relatively small budgets. We saw that as budgets increased, advertisers average depletion time steadily approached the simulation end.

4.2 Linear contextual bandits with bounds over the number of actions.

At each period $t \in [T]$, an agent observes a matrix $W^t \in \mathbb{R}^d \times \mathbb{R}^n$ and can decide between playing an action or not. If it plays an action, it incurs a cost of $\rho$ and selects a coordinate $i(t) \in [d]$. It then
observes a reward \( r^t \) with mean \( \mathbb{E}[r^t] = (W^t_{i(t)})^T \theta^* \), where \( W^t_{i(t)} \) is the \( i(t)^{th} \) row of \( W^t \) and \( \theta^* \) is an unknown parameter. We assume that \( r^t = (W^t_{i(t)})^T \theta^* + \epsilon \) with \( \epsilon \) being a zero-mean noise independent of the algorithm history. If the agent does not play an action it incurs no cost. The agent operates at most for \( T \) periods, requiring its total cost to be lower than \( T \) and higher than 0.5\( T \). The agent does not know the distribution \( W \) over which \( W^t \) is sampled (but knows that they are sampled i.i.d.). We can model this problem as having \( Z = \{ z \in \mathbb{R}_+^K : \sum_{i=1}^T z_i \leq 1 \} \), \( W \) being the set of possible matrix arrivals, \( f(z; \theta, W^t) = ((W^t_1)^T \theta, \ldots, (W^t_d)^T \theta)^T z \), and \( c(z; \theta, W^t) = (\rho, \ldots, \rho) \odot z \). Even when \( Z \) allows probabilistic allocations, there is always a solution of Step 3. of Algorithm 2 which takes at most one action per period.

**Experiment Details.** We tried eight combinations of \( d \times n \), run Algorithm 2 using \( T = 1000, 5000, 10000 \), use \( \rho = 4 \), and run 100 simulations of each experiment setting. Each simulation uses a unique seed to create \( \theta^* \) and the mean matrix \( W \) by sampling i.i.d. Uniform\((−0.5, 0.5)\) random variables. Both \( \theta^* \) and \( W \) are then normalized to satisfy \( \|\theta^*\|_2 = 1 \) and \( \|W^t\|_2 = 1 \) for all \( d' \in [d] \).

Besides the eight \( d \times n \) configurations and three possible \( T \) values, we tried six ways of obtaining the revenue terms (making a total of 144 experiment configurations). First, to create \( W^t \) we either use \( W^t = W \) for all \( t \in [T] \), i.e., no randomness, or \( W^t = W + \xi^t \) with \( \xi^t \) a random matrix with each element being sampled i.i.d. from a Uniform\((−0.1, 0.1)\) random variable. Also, given a selected action \( i(t) \in [d] \) on period \( t \in [T] \), the observed revenue is either \( W^t_{i(t)} \theta^* \) or \( W^t_{i(t)} \theta^* \) plus either a Uniform\((−0.1, 0.1)\) or Uniform\((−0.5, 0.5)\) random term. We run Step 7. of algorithm 2 using subgradient descent.

We implemented Step 1. of Algorithm 2 in the following ways. 1. Gaussian Thompson-Sampling as in Agrawal and Goyal (2013). 2. Least-squares estimation. 3. Ridge regression estimation. 4. Ridge regression estimation plus a decaying randomized perturbation. 5. ‘Known \( \theta^* \)’. The last method represents the case of a pure online optimization problem. We also solve (1) optimally for each combination of experiment setting and simulation. In this case \( \text{OPT}(\mathcal{P}) = \text{OPT}(\mathcal{P}, 0) \), and each optimization problem inside \( \text{OPT}(\mathcal{P}, 0) \) is a bag problem. Please refer to the supplement for detailed descriptions of the methods, more experimental details, and the proof that \( \text{OPT}(\mathcal{P}) = \text{OPT}(\mathcal{P}, 0) \).

Table 1 shows the percentage of the average revenue obtained against the best possible revenue achievable over the 100 simulations when using \((d \times n)\) equal to \((50, 50)\). A column label, such as \((0.5, 0.1)\) indicates that a Uniform\((−0.5, 0.5)\) is added to the observed revenue and that i.i.d. Uniform\((−0.1, 0.1)\) elements were added element-wise to \( W \) for each \( t \in [T] \). ‘0.0’ indicates that no randomness was added either to the revenue or \( W^t \) matrices depending on the case. (When \( W \) has no randomness, the ‘Known \( \theta^* \)’ method matches \( \text{OPT}(\mathcal{P}) \) as the optimal action is always the same.)

Table 1 shows interesting patterns. First, Thompson Sampling implemented as in Agrawal and Goyal (2013) was the best performing ‘learning’ method when no randomness was added, but performs terribly when the revenue had added randomness. Differently, the Least Squares and the Ridge Regression methods increased their relative performance greatly when randomness was added to the revenue term. Interestingly, adding uncertainty to ridge regression was a clear improvement when \( W^t = W \), but it did not help when \( W^t \) had randomness. These results show that how to apply Step 1. of Algorithm 2 should depend on the application and randomness. Finally, the results
| $T = 10000, (d \times n) = (50,50)$ | $(0.0,0.0)$ | $(0.1,0.0)$ | $(0.5,0.0)$ | $(0.0,0.1)$ | $(0.1,0.1)$ | $(0.5,0.1)$ |
|---------------------------------|-------------|-------------|-------------|-------------|-------------|-------------|
| Least Squares                   | 43.2        | 51.2        | 59.5        | 91.4        | 91.5        | 85.8        |
| Thompson Sampling               | 98.1        | 13.2        | 2.3         | 93.1        | 19.7        | 3.5         |
| Ridge Reg.                      | 44.9        | 52.9        | 65.0        | 95.6        | 94.5        | 84.9        |
| Ridge Reg. + Perturbation       | 59.3        | 63.2        | 67.7        | 95.5        | 94.4        | 85.2        |
| Known $\theta^*$               | 100         | 100         | 99.9        | 96.7        | 96.7        | 96.8        |

Table 1: The results shown are the average revenue over 100 simulations relative to the best value possible. A column label, such as $(0.5, 0.1)$ indicates that a Uniform$(-0.5, 0.5)$ is added to the observed revenue and that $i.i.d.$ Uniform$(-0.1, 0.1)$ elements were added to each coordinate of $W^t$ for each $t \in [T]$.

shown in Table I should be considered just as illustrative as the methods’ parameters were not tuned carefully, and neither the method’s particular implementation as in the case of Thompson Sampling.

Bibliography

Agrawal, S. and Devanur, N. (2016). Linear contextual bandits with knapsacks. In Advances in Neural Information Processing Systems, pages 3450–3458.

Agrawal, S. and Devanur, N. R. (2014). Bandits with concave rewards and convex knapsacks. In Proceedings of the fifteenth ACM conference on Economics and computation, pages 989–1006.

Agrawal, S. and Goyal, N. (2013). Thompson sampling for contextual bandits with linear payoffs. In International Conference on Machine Learning, pages 127–135.

Arora, S., Hazan, E., and Kale, S. (2012). The multiplicative weights update method: a meta-algorithm and applications. Theory of Computing, 8(1):121–164.

Badanidiyuru, A., Kleinberg, R., and Slivkins, A. (2018). Bandits with knapsacks. Journal of the ACM (JACM), 65(3):1–55.

Balseiro, S., Lu, H., and Mirrokni, V. (2020a). The best of many worlds: Dual mirror descent for online allocation problems. arXiv preprint arXiv:2011.10124.

Balseiro, S., Lu, H., and Mirrokni, V. (2020b). Dual mirror descent for online allocation problems. In International Conference on Machine Learning, pages 613–628. PMLR.

Beck, A. and Teboulle, M. (2003). Mirror descent and nonlinear projected subgradient methods for convex optimization. Operations Research Letters, 31(3):167–175.

Bertsimas, D. and Popescu, I. (2003). Revenue management in a dynamic network environment. Transportation science, 37(3):257–277.

Boyd, E. A. and Bilegan, I. C. (2003). Revenue management and e-commerce. Management science, 49(10):1363–1386.

Chen, T., Ling, Q., and Giannakis, G. B. (2017). An online convex optimization approach to
proactive network resource allocation. *IEEE Transactions on Signal Processing*, 65(24):6350–6364.

Devanur, N. R., Jain, K., Sivan, B., and Wilkens, C. A. (2011). Near optimal online algorithms and fast approximation algorithms for resource allocation problems. In *Proceedings of the 12th ACM conference on Electronic commerce*, pages 29–38.

Diemert, E., Meynet, J., Galland, P., and Lefortier, D. (2017). Attribution modeling increases efficiency of bidding in display advertising. *arXiv preprint arXiv:1707.06409*.

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

Immorlica, N., Sankararaman, K. A., Schapire, R., and Slivkins, A. (2019). Adversarial bandits with knapsacks. In 2019 *IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 202–219. IEEE.

Jenatton, R., Huang, J., and Archambeau, C. (2016). Adaptive algorithms for online convex optimization with long-term constraints. In *International Conference on Machine Learning*, pages 402–411. PMLR.

Liakopoulos, N., Destounis, A., Paschos, G., Spyropoulos, T., and Mertikopoulos, P. (2019). Cautious regret minimization: Online optimization with long-term budget constraints. In *International Conference on Machine Learning*, pages 3944–3952.

Mahdavi, M., Jin, R., and Yang, T. (2012). Trading regret for efficiency: online convex optimization with long-term constraints. *The Journal of Machine Learning Research*, 13(1):2503–2528.

Mehta, A., Saberi, A., Vazirani, U., and Vazirani, V. (2007). Adwords and generalized online matching. *Journal of the ACM (JACM)*, 54(5):22.

Pan, J., Xu, J., Ruiz, A. L., Zhao, W., Pan, S., Sun, Y., and Lu, Q. (2018). Field-weighted factorization machines for click-through rate prediction in display advertising. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pages 1349–1357. International World Wide Web Conferences Steering Committee.

Wei, X., Yu, H., and Neely, M. J. (2020). Online primal-dual mirror descent under stochastic constraints. In *Abstracts of the 2020 SIGMETRICS/Performance Joint International Conference on Measurement and Modeling of Computer Systems*, pages 3–4.

Yuan, J. and Lamperski, A. (2018). Online convex optimization for cumulative constraints. In *Advances in Neural Information Processing Systems*, pages 6137–6146.
5 Additional Theoretical Results and Examples

5.1 Different Cases for $\arg\max_{\gamma \in [0, 1]} \text{OPT}(P, \gamma)$

Take the case of $T = 1$, $Z = \{(0, 1)\}$, $W = \{w_1, w_2\}$ with equal probability of occurring, $b = 1$, and $\alpha = 0.5$. Call $\Pi(\cdot \in A)$ to the function that takes the value of 0 if condition $A$ holds and $-\infty$ otherwise. We show examples in which $\arg\max_{\gamma \in [0, 1]} \text{OPT}(P, \gamma)$ match the different cases mentioned in the paper. In most of the examples below the upper bound cost constraint hold trivially, reason why we do not “enforce” it using $\Pi(\cdot \leq 1)$, with the only exception on the $\gamma = \frac{1}{2}$ example.

Infinite solutions. $f(z; \theta^*, w_1) = z$, $c(z; \theta^*, w_1) = z$, $f(z; \theta^*, w_2) = z$, $c(z; \theta^*, w_2) = z$. In this case $\mathbb{E}[f(z; \theta^*, w)] = z$ and $\mathbb{E}[c(z; \theta^*, w)] = z$. Then, for any $\gamma \in [0, 1]$ we have

$$\text{OPT}(P, \gamma) = \frac{1}{2} \left( \max_{z \in [0, 1]} \left\{ z + \Pi(\frac{1}{2} \leq z) \right\} + \max_{z \in [0, 1]} \left\{ z + \Pi(\frac{1}{2} \leq z) \right\} \right)$$

The equality comes directly from the definition of $\text{OPT}(P, \gamma)$. Is direct to see that $z = 1$ maximizes both optimization problems and that $\text{OPT}(P) = \text{OPT}(P, \gamma)$ for all $\gamma \in [0, 1]$.

No solution. $f(z; \theta^*, w_1) = z$, $c(z; \theta^*, w_1) = 0$, $f(z; \theta^*, w_2) = 0$, $c(z; \theta^*, w_2) = 0$. Since the cost terms are always zero, the cost lower bound 0.5 is never achieved and no feasible solution exist.

$\gamma = \frac{1}{2}$ as unique solution. $f(z; \theta^*, w_1) = z$, $c(z; \theta^*, w_1) = 0$, $f(z; \theta^*, w_2) = -z$, $c(z; \theta^*, w_2) = 2z$. In this case $\mathbb{E}[f(z; \theta^*, w)] = 0$ and $\mathbb{E}[c(z; \theta^*, w)] = z$. Then, for any $\gamma \in [0, 1]$ we have

$$\text{OPT}(P, \gamma) = \frac{1}{2} \left( \max_{z \in [0, 1]} \{ (1 - \gamma)z + \Pi(\frac{1}{2} \leq z) \} + \max_{z \in [0, 1]} \{ -(1 - \gamma)z + \Pi(\frac{1}{2} \leq z) \} \right)$$

The second equality uses that the first optimization problem has $z = 1$ as its unique optimal solution whenever $\gamma \neq 1$ and that $0 = \text{OPT}(P, 1) < \text{OPT}(P, 0.5) = \frac{1}{6}$. Is direct from the result above that $\text{OPT}(P, \gamma) = -\infty$ for any $\gamma < 0.5$. Then, we have:

$$\text{OPT}(P) = \frac{1}{2} \left( \max_{z \in [0, 1], \gamma \in [0.5, 1]} \{ (1 - \gamma)z - \Pi(\frac{1}{2} \leq z) \} \right)$$

The first equality uses the definition of $\text{OPT}(P)$ and that we have restricted $\gamma$ to be in $[0.5, 1)$. The second equality uses that for any $\gamma \in [0.5, 1)$ the unique optimal is $z(\gamma) = \frac{1}{2(1 - \gamma)}$ as it maximizes the term $-(1 - \gamma)z$ by taking the smallest feasible $z$ value that satisfies the cost lower bound. Finally, the function $\xi(\gamma) := (1 - \gamma) - \frac{1}{2(1 - \gamma)}$ is differentiable on $\gamma \in [0.5, 1)$ and has strictly negative derivative on $\gamma \in [0.5, 1)$, which implies $\xi(0.5) > \xi(\gamma)$ for any $\gamma \in [0.5, 1)$, proving that $\gamma = 0.5$ is the unique optimal solution.

$\gamma = 0$ as unique solution. $f(z; \theta^*, w_1) = z^2$, $c(z; \theta^*, w_1) = z$, $f(z; \theta^*, w_2) = -z$, $c(z; \theta^*, w_2) = 1 - z$. In this case $\mathbb{E}[f(z; \theta^*, w)] = 0.5(z^2 - z)$ and $\mathbb{E}[c(z; \theta^*, w)] = 0.5$. Then, for any $\gamma \in [0, 1]$ we
have
\[
\text{OPT}(\mathcal{P}, \gamma) = \frac{1}{2} \left( \max_{z \in [0,1]} \left\{ z^2 \left( 1 - \frac{\gamma}{2} \right) - z \frac{\gamma}{2} + \Pi \left( \frac{1}{2} \leq (1 - \gamma) z + \frac{\gamma}{2} \right) \right\} 
+ \max_{z \in [0,1]} \left\{ \frac{\gamma}{2} z^2 - z \left( 1 - \frac{\gamma}{2} \right) + \Pi \left( \frac{1}{2} \leq (1 - \gamma) \left( 1 - z \right) + \frac{\gamma}{2} \right) \right\} \right)
\]

To understand why \( \gamma = 0 \) is the unique solution let us analyze both maximization problems separately. The expression \( \frac{\gamma}{2} z^2 - z \left( 1 - \frac{\gamma}{2} \right) \) in the second maximization problem is non-positive in \((z, \gamma) \in [0,1]^2\) as we can write it as \( \left( \frac{\gamma}{2} z^2 - \frac{1}{2} z \right) - z \left( \frac{1}{2} - \frac{\gamma}{2} \right) \) where each term is non-positive. Then, an optimal solution for it is \((z, \gamma) = (0,0)\) which also satisfies the lower cost constraints. Similarly, the expression \( z^2 \left( 1 - \frac{\gamma}{2} \right) - z \frac{\gamma}{2} \) in \((z, \gamma) \in [0,1]^2\) of the first maximization problem has a maximum in \((z, \gamma) = (1,0)\), optimal pair which also satisfies the lower cost constraints.

\( \gamma = 1 \) as unique solution. \( f(z; \theta^*, w_1) = z, c(z; \theta^*, w_1) = 0, f(z; \theta^*, w_2) = z, c(z; \theta^*, w_2) = z. \) In this case \( E[f(z; \theta^*, w)] = z \) and \( E[c(z; \theta^*, w)] = 0.5z \). Then, for any \( \gamma \in [0,1] \) we have
\[
\text{OPT}(\mathcal{P}, \gamma) = \frac{1}{2} \left( \max_{z \in [0,1]} \left\{ z + \Pi \left( \frac{1}{2} \leq \frac{\gamma}{2} z \right) \right\} + \max_{z \in [0,1]} \left\{ z + \Pi \left( \frac{1}{2} \leq (1 - \frac{\gamma}{2}) z \right) \right\} \right)
\]

The result is direct as \((z, \gamma) = (1,1)\) is the only pair in \([0,1]^2\) which makes the first optimization problem feasible.

### 5.2 Bound on \( \Delta_{\text{Learn}} \)

Before stating this subsection result, we define an stricter version of Assumption \ref{assum:bounded-dual-iterates}.

**Assumption 5.1 (Stricter) Bounded Dual Iterates.** There is an absolute constant \( C'_h > 0 \) such that \( \| \lambda^t \|_1 \leq C'_h \) for all \( t \in [T] \) almost surely.

**Proposition 5.** Run Algorithm \ref{alg:learn} with a constant “step-size” rule \( \eta_t \leftarrow \eta \) for all \( t \geq 1 \) where \( \eta > 0 \). Suppose that Assumption \ref{assum:bounded-dual-iterates} holds and that \( c(\cdot; \cdot, \cdot) \) is Lipschitz on its \( \theta \) argument, in particular, that it exists \( L_c > 0 \), such that \( \| c(z; \theta, w) - c(z; \theta', w) \|_\infty \leq L_c \| \theta - \theta' \|_\theta \) for any \((z, w, \theta, \theta') \in \mathcal{Z} \times \mathcal{W} \times \Theta \times \Theta\). Then, for any distribution \( \mathcal{P} \) over \( w \in \mathcal{W} \), it holds that
\[
\Delta_{\text{Learn}} \leq L_c \left( 1 + C'_h \right) E \left[ \sum_{t=1}^{T_A} \| \theta^* - \theta^t \|_\theta \right).
\]

**Proof.** The proof is obtained directly by bounding each term of \( \Delta_{\text{Learn}} \) separately. First,
\[
E \left[ \sum_{t=1}^{T_A} c(z^t; \theta^*, w^t) - c(z^t; \theta', w^t) \right] \leq E \left[ \sum_{t=1}^{T_A} \| c(z^t; \theta^*, w^t) - c(z^t; \theta', w^t) \|_\infty \right]
\leq L_c E \left[ \sum_{t=1}^{T_A} \| \theta^* - \theta^t \|_\theta \right].
\]
where we have used above that \( c(\cdot;\cdot,\cdot) \) its Lipschitz on its \( \theta \) argument. Now, for any pair \( x,y \) of real vectors of same dimension it holds \( |x^T y| \leq \|x\|\|y\|_1 \). Using the latter fact and again that \( c(\cdot;\cdot,\cdot) \) is Lipschitz on its \( \theta \) argument, we have

\[
\mathbb{E} \left[ \sum_{t=1}^{T_A} (c(z^t; \theta^*, w^t) - c(z^t; \theta^t, w^t)) T^t \lambda^t \right] 
\leq \mathbb{E} \left[ \sum_{t=1}^{T_A} \|c(z^t; \theta^*, w^t) - c(z^t; \theta^t, w^t)\|_\infty \|\lambda^t\|_1 \right] 
\leq L_c \mathbb{E} \left[ \sum_{t=1}^{T_A} \|\lambda^t\|_1 \|\theta^* - \theta^t\| \right] 
\leq L_c C_0^* \mathbb{E} \left[ \sum_{t=1}^{T_A} \|\theta^* - \theta^t\| \right].
\]

\[\square\]

### 5.3 Proof That \( \text{OPT}(\mathcal{P}) = \text{OPT}(\mathcal{P}, 0) \) in the Linear Contextual Bandits Experiment and Solving it Efficiently.

This appendix subsection shows the following three results. 1. That for any \( \rho \geq 0.5 \) we have \( \text{OPT}(\mathcal{P}, \gamma) > -\infty \) for all \( \gamma \in [0,1] \). 2. That \( \text{OPT}(\mathcal{P}, \gamma) \leq \text{OPT}(\mathcal{P}, 0) \) for all \( \gamma \in (0,1) \). 3. How to efficiently solve \( \text{OPT}(\mathcal{P}, 0) \). Take \( Z = \{z \in \mathbb{R}^K : \sum_{i=1}^K z_i \leq 1 \} \) and \( \gamma \in [0,1] \) arbitrary. As notation, here we use superscripts to denote time (but also use \( \cdot^T \) to denote dot operation between vectors when need), use subscripts to denote row indexes, and use \( W, W', W^t, W^{rt} \) to represent matrices of size \( d \times n \). Also, to shorten notation, we write \( W \) to define a sequence \( \{W^1, \ldots, W^T\} \) of \( W^t \) matrices (analogous for \( W' \)). The traditional multiplication between a matrix \( A \) of size \( d \times n \) and a vector \( x \) of size \( n \) is written as \( Ax = ((A_1)^T x, \ldots, (A_d)^T x) \). The term inside the outer expectation of \( \text{OPT}(\mathcal{P}, \gamma) \) corresponds to (for \( \gamma = 1 \) the outer expectation can be removed)

\[
O(W, \gamma) := \max_{z^1 \in Z, t \in [T]} (1 - \gamma) \sum_{t=1}^{T} (W^t \theta^*)^T z^t + \gamma \mathbb{E}_{W' \sim \mathcal{P}}[(W' \theta^*)^T z^t] \]

\[\text{s.t. } 0.5 \ast T \leq \rho \sum_{t=1}^{T} \sum_{i=1}^{d} z^t_i \leq T.\]

Notice that a solution \( z = \{z^1, \ldots, z^T\} \) is either feasible or infeasible independently of the context vector arrivals \( W = \{W^1, \ldots, W^T\} \) and \( \gamma \). For any \( \rho \geq 0.5 \) and \( \gamma \in [0,1] \), it holds \( \text{OPT}(\mathcal{P}, \gamma) > -\infty \) as we can choose \( z \) satisfying \( \sum_{i=1}^{d} z^t_i = 0.5/\rho \) for all \( t \in [T] \) (our problem setup uses \( \rho = 4 \)). A direct application of Jensen inequality shows \( \text{OPT}(\mathcal{P}, 1) \leq \text{OPT}(\mathcal{P}, 0) \), so let us take \( \gamma \in (0,1) \)
arbitrary. For any sequence \( W \), let \( z_\gamma(W) \) be an optimal solution of \( O(W, \gamma) \), we have

\[
\text{OPT}(\mathcal{P}, \gamma) = \mathbb{E}_{W \sim \mathcal{P}^T} \left[ (1 - \gamma) \sum_{t=1}^{T} (W_t^T \theta^*)^T z_\gamma^t(W) + \gamma \mathbb{E}_{W' \sim \mathcal{P}} \left[ (W_t^T \theta^*)^T z_\gamma^t(W) \right] \right]
\]

\[
= \mathbb{E}_{W \sim \mathcal{P}^T} \left[ (1 - \gamma) \sum_{t=1}^{T} (W_t^T \theta^*)^T z_\gamma^t(W) \right] + \mathbb{E}_{W \sim \mathcal{P}^T} \left[ \gamma \sum_{t=1}^{T} \mathbb{E}_{W' \sim \mathcal{P}} \left[ (W_t^T \theta^*)^T z_\gamma^t(W) \right] \right]
\]

\[
= \mathbb{E}_{W \sim \mathcal{P}^T} \left[ (1 - \gamma) \sum_{t=1}^{T} (W_t^T \theta^*)^T z_\gamma^t(W) \right] + \mathbb{E}_{W' \sim \mathcal{P}^T, W \sim \mathcal{P}^T} \left[ \gamma \sum_{t=1}^{T} ((W_t^T \theta^*)^T z_\gamma^t(W')) \right]
\]

\[
\leq \mathbb{E}_{W \sim \mathcal{P}^T} \left[ \sum_{t=1}^{T} (W_t^T \theta^*)^T z_0^t(W) \right] = \text{OPT}(\mathcal{P}, 0).
\]

The second equality uses the linearity of the expectation operator, the third uses that each \( W' \) is sampled i.i.d. from \( \mathcal{P} \), the fourth that \( W \) and \( W' \) are i.i.d. and can be exchanged, the fifth uses the linearity of the expectation operator again, and the final inequality uses the definition of \( z_0(W) \). In particular, the last inequality uses that \((1 - \gamma)z_\gamma(W) + \gamma \mathbb{E}_{W' \sim \mathcal{P}} [z_\gamma(W')] \) is a feasible solution of \( O(W, 0) \). Finally, notice that for any given \( W \) solving \( O(W, 0) \) is equivalent to solving the following knapsack problem

\[
O(W, 0) = \max_{y^t \in [0,1]: t \in T} \sum_{i=1}^{T} \left( \max_{i \in [d]} (W_i^T \theta^*)^T y^t \right)
\]

\[
\text{s.t. \hspace{1em} } 0.5 \times T \leq \rho \sum_{t=1}^{T} y^t \leq T.
\]

Let \( \{m_1, \ldots, m_T\} \) represent the sequence \( \max_{i \in [d]} (W_i^T \theta^*)^T \) ordered from biggest to smallest value. Then, is not hard to see that

\[
O(W, 0) = \max_{i_{\text{max}} \in \left[ \frac{T}{2 \rho}, \left\lfloor \frac{T}{\rho} \right\rfloor \right]} \sum_{i=1}^{i_{\text{max}}} m_i,
\]

where \( \lceil \cdot \rceil \) and \( \lfloor \cdot \rfloor \) are the traditional ceiling and floor integer functions respectively.
6 Proofs

6.1 Proof of Proposition 1

Proof. Let \( P^T \) be the distribution from which the \((w^1, \ldots, w^T)\) vectors are sampled, with each \( w^t \) being sampled i.i.d. from \( P \). For any \( \gamma \in [0, 1] \)

\[
\text{OPT}(P, \gamma) = \mathbb{E}_P \left[ \max_{z^t \in Z^T, t \in [T]} \sum_{t=1}^T (1 - \gamma)f(z^t; \theta^*, w^t) + \gamma \mathbb{E}_P[f(z^t; \theta^*, w)] \right] \text{ s.t. } T\alpha_k b_k \leq \sum_{t=1}^T (1 - \gamma)c_k(z^t; \theta^*, w^t) + \gamma \mathbb{E}_P[c_k(z^t; \theta^*, w)] \leq T b_k \text{ for all } k \in [K] \\
\leq \mathbb{E}_P \left[ \sum_{t=1}^T \max_{z^t \in Z^T, t \in [T]} (1 - \gamma) \left( f(z^t; \theta^*, w^t) - \lambda^T c(z^t; \theta^*, w^t) \right) + \gamma \mathbb{E}_P[f(z^t; \theta^*, w) - \lambda^T c(z^t; \theta^*, w)] \right] + Tp(\lambda) \\
= \sum_{t=1}^T \mathbb{E}_P \left[ \max_{z^t \in Z^T, t \in [T]} f(z^t; \theta^*, w^t) - \lambda^T c(z^t; \theta^*, w^t) \right] + \gamma \mathbb{E}_P \left[ \max_{z^t \in Z^T, t \in [T]} f(z^t; \theta^*, w) - \lambda^T c(z^t; \theta^*, w) \right] + Tp(\lambda) \\
\leq (1 - \gamma) \mathbb{E}_P \left[ \max_{z^t \in Z^T, t \in [T]} f(z; \theta^*, w) - \lambda^T c(z; \theta^*, w) \right] + \gamma T \mathbb{E}_P \left[ \max_{z^t \in Z^T, t \in [T]} f(z; \theta^*, w) - \lambda^T c(z; \theta^*, w) \right] + Tp(\lambda) \\
\leq (1 - \gamma) T \mathbb{E}_P \left[ \varphi(\lambda; \theta^*, w) \right] + \gamma \mathbb{E}_P \left[ \max_{z^t \in Z^T, t \in [T]} f(z; \theta^*, w) - \lambda^T c(z; \theta^*, w) \right] + Tp(\lambda) \\
= T \mathbb{E}_P \left[ \varphi(\lambda; \theta^*, w) \right] + Tp(\lambda) \\
= TD(\lambda; \theta^*)
\]

The first equality is the definition of \( \text{OPT}(P, \gamma) \), the first inequality uses Lagrangian duality for both the lower and upper bounds constraints, the second equality uses that \( p(\lambda) \) can be moved outside the expectation and that the sum can be changed with the maximization operator as there is no constraint linking the \( z^t \) variables. The second inequality uses that for any \( a(\cdot) \) and \( b(\cdot) \) real valued functions we have \( \max_{z \in Z} \{ a(z) + b(z) \} \leq \max_{z \in Z} a(z) + \max_{z \in Z} b(z) \), the third inequality uses that all \( w^t \) are i.i.d sampled, that all maximization problems are the same in the first term, and that the outer expectation can be removed from the second term. The fourth inequality uses the definition of \( \varphi(\cdot; \cdot; \cdot) \) and that \( \max_{z \in Z} \mathbb{E}_P[\cdot] \leq \mathbb{E}_P[\max_{z \in Z} \cdot] \). Finally, we use the definition of \( \varphi(\cdot; \cdot; \cdot) \) again and the fact that \( \gamma + (1 - \gamma) = 1 \). \( \square \)

6.2 Proof of Proposition 2

Proof. First note that the \( p(\cdot) \) function used inside \( D(\cdot; \cdot) \) is convex since \( b \geq 0 \) and \( \alpha \in [-1, 1]^K \). We need to prove that \( D(\lambda; \theta) + \mathbb{E}_P[g(\lambda; \theta, w)]^T (\lambda' - \lambda) \leq D(\lambda'; \theta) \) for any \( \lambda \in \Lambda \) and \( \lambda' \in \Lambda \). Let
Let \( p' \) be any member of \( \partial p(\lambda) \), we have
\[
D(\lambda; \theta) + \mathbb{E}_p[\tilde{g}(\lambda; \theta, w)]^T(\lambda' - \lambda) = \mathbb{E}_p[\varphi(\lambda; \theta, w) + p(\lambda) + \tilde{g}(\lambda; \theta, w)^T(\lambda' - \lambda)] \\
= \mathbb{E}_p[f(z(\lambda; \theta, w); \theta, w) - (\lambda')^T c(z(\lambda; \theta, w); \theta, w) + p(\lambda) + p^T(\lambda' - \lambda)] \\
\leq \mathbb{E}_p[f(z(\lambda; \theta, w); \theta, w) - (\lambda')^T c(z(\lambda; \theta, w); \theta, w) + p(\lambda')] \\
\leq D(\lambda'; \theta).
\]
The first equality uses the definition of \( D(\lambda; \theta) \), the second equality uses the definition of \( \tilde{g}(\lambda; \theta, w) \), the first inequality uses the subgradient inequality for \( p(\cdot) \), and the second inequality uses the definition of \( D(\lambda'; \theta) \).

### 6.3 Intermediate Results

The following propositions were not mentioned in the paper. Proposition 6 shows an inequality that holds for Step 7. of Algorithm 2 under the conditions given for \( \Lambda \) and \( h(\cdot) \) on the paper. Propositions 7 and 8 are intermediate steps to prove Theorem 1. Proposition 7 bounds \( g \). Propositions 6 and 8 are key results used to prove Proposition 9.

**Proposition 6.** Let \( \Lambda \subseteq \mathbb{R}^K \) be a set which can be defined separately for each dimension \( k \in [K] \), either being \( \Lambda_k = \mathbb{R} \) or \( \Lambda_k = \mathbb{R}_+ \). Let \( h(\cdot) : \Lambda \rightarrow \mathbb{R} \) be a function that satisfies \( h(\lambda) = \sum_{k=1}^K h_k(\lambda_k) \), with \( h_k(\cdot) \) being a strongly convex univariate differentiable function for all \( k \in [K] \). Given arbitrary \( \lambda' \in \Lambda, \tilde{g} \in \mathbb{R}^K \), and \( \eta > 0 \) define \( \lambda^+ = \arg \min_{\lambda \in \Lambda} \lambda^T \tilde{g}^2 + \frac{1}{\eta} V_h(\lambda, \lambda') \). Then, for all \( k \in [K] \) it holds

1. If \( \Lambda_k = \mathbb{R} \), then \( \dot{h}_k(\lambda^+_k) = \dot{h}_k(\lambda'_k) - \eta \tilde{g}_k \).
2. If \( \Lambda_k = \mathbb{R}_+ \), then \( \dot{h}_k(\lambda^+_k) = \dot{h}_k(\lambda'_k) - \eta \tilde{g}_k \) if \( \lambda^+_k > 0 \) or \( \dot{h}_k(\lambda^+_k) \geq \dot{h}_k(\lambda'_k) - \eta \tilde{g}_k \) if \( \lambda^+_k = 0 \).

Therefore, proving that \( \nabla h(\lambda^+) \geq \nabla h(\lambda') - \eta \tilde{g} \).

**Proof.** Notice that \( \min_{\lambda \in \Lambda} \lambda^T g^2 + \frac{1}{\eta} V_h(\lambda, \lambda') = \sum_{k \in [K]} \min_{\lambda_k \in \Lambda_k} \phi_k(\lambda_k; \lambda'_k, \tilde{g}_k) \) with \( \phi_k(\lambda_k; \lambda'_k, \tilde{g}_k) := \tilde{g}_k \lambda_k + \frac{1}{\eta} (\dot{h}_k(\lambda_k) - \dot{h}_k(\lambda'_k) (\lambda_k - \lambda'_k)) \) for all \( k \in [K] \). Then, independently per coordinate we minimize a strongly convex function under a non-empty closed convex set, which shows that \( \lambda^+_k \) exists for each \( k \in [K] \). Also, \( \lambda^+_k \) can be found using first order necessary optimality conditions for each \( k \in [K] \). Taking \( k \in [K] \) arbitrary, we split the proof in two cases.

\( \Lambda_k = \mathbb{R} \). By first order optimality conditions we immediately obtain \( \dot{h}_k(\lambda^+_k) = \dot{h}(\lambda'_k) - \eta \tilde{g}_k \).

\( \Lambda_k = \mathbb{R}_+ \). Define \( \Pi_+(\cdot) : \mathbb{R} \rightarrow \{0\} \cup \{\infty\} \) as the convex function that takes the value of 0 if its input is non-negative and \( \infty \) otherwise. Then, the minimization problem for dimension \( k \) can be rewritten as \( \min_{\lambda \in \Lambda_k} \phi_k(\lambda_k; \lambda'_k, \tilde{g}_k) = \Pi_+(\lambda_k) \). First order necessary optimality conditions imply 0 \( \in \partial(\phi_k(\lambda^+_k; \lambda'_k, \tilde{g}_k) + \Pi_+(\lambda_k)) \). Then, there exists \( g \in \partial(\Pi_+(\lambda^+_k)) \), such that \( \dot{h}_k(\lambda^+_k) = \dot{h}(\lambda'_k) - \eta \tilde{g}_k - \eta g \). The result is obtained directly using that \( \partial(\Pi_+(\lambda_k)) = 0 \) when \( \lambda_k > 0 \) and equal to \( \mathbb{R}_- \) when \( \lambda_k = 0 \). 

\( \square \)
Proposition 7. Run Algorithm 2 with a constant “step-size” rule \( \eta_t \leftarrow \eta \) for all \( t \geq 1 \) where \( \eta > 0 \). Suppose that Assumption 5.1 holds and take \( \tau_A \) as in Definition 3.1. Then,
\[
\mathbb{E}[T - \tau_A] \leq \frac{\bar{C}}{b} + \frac{C_h + \|\nabla h(\lambda^1)\|_\infty}{\eta b^2} + \left( \frac{\mathbb{E}[\sum_{t=1}^{\tau_A} c(z^t; \theta^t, w^t) - c(\hat{z}; \theta^t, w^t)]}{b} \right). 
\]

Proof. Let \( k' \in [K] \) be the index of the first violated upper cost bound, \( i.e. \) the index which activates the stop time \( \tau_A \). Here we assume that some upper cost bound constraint is violated, \( i.e. \) that \( \tau_A < T \), if not the result is trivial. Step 6. of Algorithm 2 defines \( \bar{g}_k' = -c_k'(z^t; \theta^t, w^t) + b_k' \mathbb{I}(\lambda'_k \geq 0) + \alpha_k' \mathbb{I}(\lambda'_k < 0) \), which can be upper bounded by \( \bar{g}_k' \leq -c_k'(z^t; \theta^t, w^t) + b_k' \). Using the definition of \( \tau_A \) and \( \bar{g}_k' \) we have
\[
\sum_{t=1}^{\tau_A} \bar{g}_k' \leq b_k' \tau_A - \sum_{t=1}^{\tau_A} c_k'(z^t; \theta^t, w^t) + \left( \sum_{t=1}^{\tau_A} (c_k'(z^t; \theta^t, w^t) - c_k'(z^t; \theta^t, w^t)) \right)
\]
\[
\leq b_k' \tau_A - b_k' T + \bar{C} + \left( \sum_{t=1}^{\tau_A} (c_k'(z^t; \theta^t, w^t) - c_k'(z^t; \theta^t, w^t)) \right)
\]
\[
\Rightarrow T - \tau_A \leq \frac{1}{b_k'} \left( \bar{C} - \sum_{t=1}^{\tau_A} \bar{g}_k' \right) + \frac{1}{b_k'} \left( \sum_{t=1}^{\tau_A} (c_k'(z^t; \theta^t, w^t) - c_k'(z^t; \theta^t, w^t)) \right).
\]

Using that our update rule satisfies \( \dot{h}_k'(\lambda_k^{t+1}) \geq \dot{h}_k'(\lambda_k^t) - \eta \bar{g}_k' \) for all \( t \leq \tau_A \) and the definitions of \( \bar{C} \) and \( C_h \), we get
\[
- \sum_{t=1}^{\tau_A} \bar{g}_k' \leq \frac{1}{\eta} \left( \dot{h}_k'(\lambda_k^{t+1}) - \dot{h}_k'(\lambda_k^t) \right)
\]
\[
\Rightarrow T - \tau_A \leq \frac{\bar{C}}{b_k'} + \frac{\dot{h}_k'(\lambda_k^{t+1}) - \dot{h}_k'(\lambda_k^t)}{\eta b_k'} + \left( \frac{\sum_{t=1}^{\tau_A} (c_k'(z^t; \theta^t, w^t) - c_k'(z^t; \theta^t, w^t))}{b_k'} \right)
\]
\[
\Rightarrow \mathbb{E}[T - \tau_A] \leq \frac{\bar{C}}{b} + \frac{C_h + \|\nabla h(\lambda^1)\|_\infty}{\eta b^2} + \left( \frac{\mathbb{E}[\sum_{t=1}^{\tau_A} c(z^t; \theta^t, w^t) - c(z^t; \theta^t, w^t)]}{b} \right). 
\]

Proposition 8. Run Algorithm 2 with a constant “step-size” rule \( \eta_t \leftarrow \eta \) for all \( t \geq 1 \) where \( \eta > 0 \). Denote \( \lambda_{\tau_A} = \frac{\sum_{t=1}^{\tau_A} \lambda^t_{\tau_A}}{\tau_A} \) (\( \tau_A \) as in Definition 3.1). It holds
\[
\mathbb{E} \left[ \tau_A D(\bar{\lambda}^t_A; \theta^*) - \sum_{t=1}^{\tau_A} f(z^t; \theta^t, w^t) \right] \leq \frac{2(C^2 + b^2)}{\sigma_1} \eta \mathbb{E}[\tau_A] + \frac{1}{\eta} V_h(\lambda, \lambda^1)
\]
\[
+ \mathbb{E} \left[ \sum_{t=1}^{\tau_A} (c(z^t; \theta^t, w^t) - c(z^t; \theta^t, w^t)) \right]. 
\]

Proof. For clarity, we sometimes use \( \mathbb{E}[\cdot] \), \( \mathbb{E}[w_t[\cdot] \), or \( \mathbb{E}[w_{t-1}[\cdot] \) to indicate the random variable over which the expectation is taken. Using \( \mathbb{E}[\cdot] \) indicates that the expectation is taken over the “whole” randomness of Algorithm 2. Call \( \bar{g}_t \) the vector obtained in Step 6. and define \( \mathbb{E}[\bar{g}_t] = g^t \). The proof is composed of three steps. 1. Bounding \( \bar{g}_t \). 2. Upper bounding \( \mathbb{E} \left[ \sum_{t=1}^{\tau_A} (g^t)^T (\lambda^t - \lambda) \right] \). 3. Lower
bounding $\mathbb{E}\left[\sum_{s=1}^{r_A}(g^s)^T(\lambda^s - \lambda)\right]$. The upper and lower bounds match the left and right hand side of the terms in Proposition.

**Step 1.** Upper bound for $\mathbb{E}[\|g^t\|_\infty^2]$.

$$\mathbb{E}[\|g^t\|_\infty^2] \leq \mathbb{E}[(\|c(z^t; \theta^t, w^t)\|_\infty + \|b\|_\infty)^2] \leq 2\mathbb{E}[\|c(z^t; \theta^t, w^t)\|_\infty^2 + \|b\|_\infty^2] \leq 2(C^2 + b^2)$$

**Step 2.** Upper bound for $\mathbb{E}\left[\sum_{s=1}^{r_A}(g^s)^T(\lambda^s - \lambda)\right]$. Notice

$$\mathbb{E}_{w^t}[(g^t)^T(\lambda^t - \lambda)|\lambda^t, \theta^t] \leq \mathbb{E}_{w^t}\left[(g^t)^T(\lambda^t - \lambda^{t+1}) + \frac{1}{\eta}V_h(\lambda, \lambda^t) - \frac{1}{\eta}V_h(\lambda, \lambda^{t+1})|\lambda^t, \theta^t\right]$$

$$\leq \mathbb{E}_{w^t}\left[(g^t)^T(\lambda^t - \lambda^{t+1}) + \frac{1}{\eta}V_h(\lambda, \lambda^t - \lambda^{t+1}) - \frac{\sigma^1}{2\eta}\|\lambda^{t+1} - \lambda^t\|_1^2|\lambda^t, \theta^t\right]$$

$$\leq \mathbb{E}_{w^t}\left[\frac{\eta}{\sigma^1}\|g^t\|_\infty^2 + \frac{1}{\eta}V_h(\lambda, \lambda^t) - \frac{1}{\eta}V_h(\lambda, \lambda^{t+1})|\lambda^t, \theta^t\right]$$

$$\leq \frac{2\eta}{\sigma^1}(C^2 + b^2) + \frac{1}{\eta}V_h(\lambda, \lambda^t) - \mathbb{E}_{w^t}\left[\frac{1}{\eta}V_h(\lambda, \lambda^{t+1})|\lambda^t, \theta^t\right], \quad (4)$$

where the first inequality is due to the three point property (Lemma 4.1 of Beck and Teboulle (2003)), the second uses $V_h(\lambda^{t+1}, \lambda^t \geq \frac{\sigma^1}{2}\|\lambda^{t+1} - \lambda^t\|_1^2$ given that $h(\cdot)$ is $\sigma^1$-strongly convex with respect to the $\|\cdot\|_1$ norm, the third uses that for any two vectors $a^1$ and $a^2$ of same dimension it holds $(a^1)^T a^2 + 0.5\|a^1\|_\infty^2 \geq -0.5\|a^2\|_1^2$, and the final inequality is just understanding which terms are constant under the conditional expectation. Taking $\mathbb{E}_{\mathcal{H}^{t-1}}[\cdot]$ over both sides of equation (4) and using the law of total expectation we get

$$\mathbb{E}[\eta(g^t)^T(\lambda^t - \lambda)] \leq \frac{2(C^2 + b^2)}{\sigma^1} \eta \mathbb{E}[V_h(\lambda, \lambda^t)] - \mathbb{E}[V_h(\lambda, \lambda^{t+1})], \quad (5)$$

since the pair $(\lambda^t, \theta^t)$ is completely determined by $\mathcal{H}^{t-1} \cup \{w^t\}$ and that $w^t$ is independent of $\mathcal{H}^{t-1}$. Then, regardless of the value of $r_A$, using the telescopic property and that $V_h(\cdot, \cdot)$ is non-negative we obtain

$$\mathbb{E}\left[\sum_{s=1}^{r_A}(g^s)^T(\lambda^s - \lambda)\right] \leq \frac{2(C^2 + b^2)}{\sigma^1} \eta \mathbb{E}[r_A] + \frac{V_h(\lambda, \lambda^1)}{\eta}. \quad (6)$$

**Step 3.** Lower bounds for $\mathbb{E}\left[\sum_{s=1}^{r_A}(g^s)^T(\lambda^s - \lambda)\right]$. By definition of $g^t$, using the subgradient inequality we get

$$(g^t)^T(\lambda^t - \lambda) \geq D(\lambda^t; \theta^t) - D(\lambda; \theta^t) \geq D(\lambda^t; \theta^t) - \left(\mathbb{E}_w[\varphi(\lambda; \theta^t, w)] + \sum_{k \in [K]} b_k([\lambda_k]_+ - \alpha_k[-\lambda_k]_+)\right).$$

For any $w \in \mathcal{W}$ we have $f(z(\lambda^t; \theta^t, w); \theta^t, w) - \lambda^T c(z(\lambda^t; \theta^t, w); \theta^t, w) \leq \varphi(\lambda; \theta^t, w)$ as by definition $z(\lambda^t; \theta^t, w)$ is an optimal solution of $\varphi(\lambda; \theta^t, w)$ not of $\varphi(\lambda; \theta^t, w)$. Defining $\lambda^{r_A} := \frac{1}{r_A} \sum_{r=1}^{r_A} \lambda^r$,
taking $\lambda = (0, 0, \ldots, 0)$, and summing from one to $\tau_A$ we get
\[
\sum_{t=1}^{\tau_A} (g^t) \cdot (\lambda^t - 0) 
\geq \sum_{t=1}^{\tau_A} D(\lambda^t, \theta^t) - \mathbb{E}_w [f(z(\lambda^t, \theta^t), w)] 
\geq \sum_{t=1}^{\tau_A} (D(\lambda^t, \theta^*)) - \mathbb{E}_w [f(z(\lambda^t, \theta^t), \theta^*, w)] + \sum_{t=1}^{\tau_A} (D(\lambda^t, \theta^t) - D(\lambda^t, \theta^*)) 
\]
\[
+ \sum_{t=1}^{\tau_A} (\mathbb{E}_w [f(z(\lambda^t, \theta^t), \theta^*, w)] - f(z(\lambda^t, \theta^t), \theta^t, w)) 
\geq \left( \tau_A D(\xi^{\tau_A}; \theta^*) - \sum_{t=1}^{\tau_A} \mathbb{E}_w [f(z(\lambda^t, \theta^t), \theta^*, w)] \right) + \sum_{t=1}^{\tau_A} (D(\lambda^t, \theta^t) - D(\lambda^t, \theta^*)) 
\]
\[
+ \sum_{t=1}^{\tau_A} (\mathbb{E}_w [f(z(\lambda^t, \theta^t), \theta^*, w)] - f(z(\lambda^t, \theta^t), \theta^t, w)). \tag{6} 
\]

Taking expectation over (6) and using the results from Step 2, we get
\[
\mathbb{E} \left[ \tau_A D(\xi^{\tau_A}; \theta^*) - \sum_{t=1}^{\tau_A} \mathbb{E}_w [f(z(\lambda^t, \theta^t), \theta^*, w)] \right] \leq \frac{2(C^2 + \tilde{b}^2)}{\sigma^2} \eta \mathbb{E}[\tau_A] + \frac{1}{\eta} V_h(0, \lambda^t) 
\]
\[
+ \mathbb{E} \left[ \sum_{t=1}^{\tau_A} \mathbb{E}_w [c(z(\lambda^t, \theta^t), \theta^*, w)] \right] 
\]
\[
- \mathbb{E} \left[ \sum_{t=1}^{\tau_A} \mathbb{E}_w [c(z(\lambda^t, \theta^t), \theta^*, w)] \right] \tau^1, \tag{7} 
\]

where we have used the definition of $D(\cdot, \cdot)$ to reduce the second line of (7) to use only the cost functions. Equation (7) almost matches the conclusion of Theorem 1 except that (7) uses a $\mathbb{E}[\sum_{t=1}^{\tau_A} \mathbb{E}_w [\cdot]]$ term, while the theorem uses $\mathbb{E}[\sum_{t=1}^{\tau_A} \mathbb{E}_w [\cdot]]$. The previous issue is solved using the Optional Stopping Theorem. We prove now that $\mathbb{E} \left[ \sum_{t=1}^{\tau_A} \mathbb{E}_w [f(z(\lambda^t, \theta^t), \theta^*, w)] \right]$ equals $\mathbb{E} \left[ \sum_{t=1}^{\tau_A} \mathbb{E}_w [f(z(\lambda^t, \theta^t), \theta^*, w)] \right]$ (the analysis for the cost terms appearing in the second line of (7) is analogous). First notice
\[
\mathbb{E}_w [f(z(\lambda; \theta, \theta^*, w)| \lambda = \lambda^t, \theta = \theta^t] = \mathbb{E}_w [f(z(\lambda^t; \theta^t, \theta^*, w)| \mathcal{H}^{t-1}]. 
\]

Define the martingale $M^t = \sum_{s=1}^{t} f(z(\lambda^t; \theta^t, \theta^*, w^t), \theta^*, w^t) - \mathbb{E}_w [f(z(\lambda^t; \theta^t, \theta^*, w)| \mathcal{H}^{t-1}]$ for all $t \leq T$. Using that $\tau_A$ is a stop time w.r.t. to the filtration $\mathcal{H}^t$, the Optional Stopping Time ensures $\mathbb{E}[M^{\tau_A}] = \mathbb{E}[M^1] = 0$, therefore:
\[
\mathbb{E} \left[ \sum_{t=1}^{\tau_A} \mathbb{E}_w [f(z(\lambda^t; \theta^t, \theta^*, w)| \mathcal{H}^{t-1})] \right] = \mathbb{E} \left[ \sum_{t=1}^{\tau_A} f(z(\lambda^t; \theta^t, \theta^*, w)| \mathcal{H}^{t-1}) \right] 
\]

concluding the proof. \qed

**Proposition 9.** Run Algorithm 2 with a constant “step-size” rule $\eta_t \leftarrow \eta$ for all $t \geq 1$ where $\eta > 0$. Using $\delta_\theta$ as in Definition 3.2, for each $t \in [T-1]$ it holds (here we use 0 to refer to the zero-vector $(0, \ldots, 0)$ of dimension $K$):
\[
\mathbb{E} [V_h(0, \lambda^t+1)| \lambda^t, \theta^t] \leq \eta \left( \frac{2\eta}{\sigma^2} (C^2 + \tilde{b}^2) + 2\tilde{C} \delta_\theta + \lambda^t \right) + V_h(0, \lambda^t). 
\]
Proof. Let \( \bar{g}^t \) be the \( \lambda^t \) stochastic subgradient obtained in Step 6. of Algorithm \[2\]. Here we abuse notation and use, e.g., \( \mathbb{E}[\bar{g}^t|\lambda^t, \theta^t] \) to represent that \( \bar{g}^t \) is a random variable on \( w \) given a fixed pair \( (\lambda^t, \theta^t) \in (\Lambda \times \Theta) \). The following bound holds

\[
\mathbb{E}_{p}[||\bar{g}^t||^2_{\infty}] \leq \mathbb{E}[(||c(z^t; \theta^t, w^t)||_{\infty} + ||b||_{\infty})^2] \leq 2\mathbb{E}[||c(z^t; \theta^t, w^t)||^2_{\infty} + ||b||_{\infty}^2] \leq 2(C^2 + \bar{b}^2).
\]

For any \( \lambda \in \Lambda \) we have

\[
\mathbb{E}[\bar{g}^t|\lambda^t, \theta^t]^T(\lambda^t - \lambda) = \mathbb{E}[(\bar{g}^t)^T(\lambda^t - \lambda)|\lambda^t, \theta^t] \leq \mathbb{E}[(\bar{g}^t)^T(\lambda^t - \lambda^t_{t+1}) + \frac{1}{\eta}V_h(\lambda, \lambda^t) - \frac{1}{\eta}V_h(\lambda, \lambda^t_{t+1})]\lambda^t, \theta^t] + \frac{\mathbb{E}[\eta ||\bar{g}^t||^2_{\infty}]}{\sigma_1} + \frac{1}{\eta}V_h(\lambda, \lambda^t) - \frac{1}{\eta}V_h(\lambda, \lambda^t_{t+1})\lambda^t, \theta^t] \leq \mathbb{E}[(\bar{g}^t)^T(\lambda^t - \lambda^t_{t+1}) + \frac{1}{\eta}V_h(\lambda, \lambda^t) - \frac{1}{\eta}V_h(\lambda, \lambda^t_{t+1})]\lambda^t, \theta^t]
\]

where we have used linearity of the expectation, the three point property, that \( V_h(\cdot, \cdot) \) is \( \sigma_1 \) strongly convex on with respect to the \( ||\cdot||_1 \) norm, Cauchy-Schwartz, and the bound for \( \mathbb{E}[||\bar{g}^t||^2_{\infty}] \) obtained before (same steps as in Step 1. and 2. of Proof \[3\]). Choosing \( \lambda = (0, \ldots, 0) \) we get

\[
\mathbb{E} [V_h(0, \lambda^t_{t+1})|\lambda^t, \theta^t] \leq \frac{2\mathbb{E}[\eta ||\bar{g}^t||^2_{\infty}]}{\sigma_1} - \mathbb{E}[\bar{g}^t|\lambda^t, \theta^t]^T\lambda^t] + V_h(0, \lambda^t).
\]

To finish the proof we now show that \( \mathbb{E}[\bar{g}^t|\lambda^t, \theta^t]^T\lambda^t \geq ||\lambda^t||_1\delta_{\theta^t} - \bar{f} \). Notice first that for any \( (\lambda^t, \theta^t) \in (\Lambda \times \Theta) \) we have \( \mathbb{E}[\bar{g}^t(w)]^T\lambda^t = -\mathbb{E}[c(z(\lambda^t; \theta^t, w); \theta^t, w)^T\lambda^t + p(\lambda^t) \) using that by definition \( p(\lambda) = \sum_{k \in [K]} b_k(\lambda_k^+ + \alpha_k[\lambda_k^-]) \). Let \( \{z(w)\}_{w \in W} \) be a series that satisfies \( \delta_{\theta^t} = \mathbb{E}_{p}[\min\{||Tb_k - c_k(z(w); \theta^t, w)||_{\infty}, \|c_k(z(w); \theta^t, w) - T\alpha_k b_k||_{\infty}\}] \). Then,

\[
\mathbb{E}[\bar{g}^t|\lambda^t, \theta^t]^T\lambda^t
\]

where we have used the definition of \( D(\lambda^t; \theta^t), \bar{f}, \delta_{\theta^t}, \) and the fact that \( ||\lambda^t||_1 = \sum_{k \in [K]} (\lambda_k^+\lambda_k^-) \).
6.4 Proof of Theorem 1

Proof. For any distribution \( \mathcal{P} \) over \( \mathcal{W} \) and for any \( t' \in [T] \) we have

\[
OPT(\mathcal{P}) \leq \frac{t'}{T} OPT(\mathcal{P}) + \frac{T - t'}{T} OPT(\mathcal{P})
\]

\[
\leq t'D(\bar{\lambda}^t; \theta^*) + (T - t') \bar{f},
\]

where we have used Proposition 1 and that a loose upper bound for \( OPT(\mathcal{P}) \) is \( T \bar{f} \). Therefore,

\[
\text{Regret}(A|\mathcal{P}) = OPT(\mathcal{P}) - R(A|\mathcal{P})
\]

\[
\leq E \left[ \tau_A D(\bar{\lambda}^A; \theta^*) + (T - \tau_A) \bar{f} - \sum_{t=1}^{\tau_A} f(z^t; \theta^*, w^t) \right]
\]

\[
= E \left[ \tau_A D(\bar{\lambda}^A; \theta^*) - \sum_{t=1}^{\tau_A} f(z^t; \theta^*, w^t) \right] + E[T - \tau_A] \bar{f}
\]

\[
\leq \frac{2(C^2 + \bar{f}^2)}{\sigma_1} \eta E[\tau_A] + \frac{1}{\eta} V_h(0, \lambda^1) + \frac{\bar{f}}{\eta} \left( \bar{C} + \frac{C_h + \|\nabla h(\lambda)\|_{\infty}}{\eta} \right)
\]

\[
+ E \left[ \sum_{t=1}^{\tau_A} (c(z^t; \theta^*, w^t) - c(z^t; \theta^*, w^t) \| \bar{\lambda}^t \|_1 \right] + \frac{\bar{f}}{\eta} \left\| E \left[ \sum_{t=1}^{\tau_A} c(z^t; \theta^*, w^t) - c(z^t; \theta^*, w^t) \right] \right\|_\infty,
\]

where in the first inequality we have used the definition of \( R(A|\mathcal{P}) \) and the fact that Algorithm 2 runs for \( \tau_A \) periods. The second inequality is obtained directly from Propositions 7 and 8. \( \square \)

6.5 Proof of Proposition 3

Proof. A direct application of Proposition 3 shows that whenever \( \|\lambda^t\|_1 \geq C^\circ / \delta \) we have

\[
E[V_h(0, \lambda^{t+1})|\lambda^t, \theta^t)] \leq V_h(0, \lambda^t). \]

Then, for any \((\lambda^t, \theta^t) \in \Lambda \times \Theta\) we have

\[
E[V_h(0, \lambda^{t+1})|\lambda^t, \theta^t)] \leq \max \left\{ \max_{\|\lambda\|_1 \leq \delta^{-1} C^\circ} V_h(0, \lambda) + \eta C^\circ, V_h(0, \lambda^1) \right\}
\]

\[
\Rightarrow E[V_h(0, \lambda^{t+1})] \leq \max \left\{ \max_{\|\lambda\|_1 \leq \delta^{-1} C^\circ} V_h(0, \lambda) + \eta C^\circ, V_h(0, \lambda^1) \right\}
\]

Take now \( h(\cdot) = \frac{1}{2}\|\cdot\|_2^2 \) then for any \( \lambda \in \Lambda \) we have \( \nabla h(\lambda) = \lambda \) and \( V_h(0, \lambda) = \frac{1}{2}\|\lambda\|_2^2 \), therefore

\[
\max_{\|\lambda\|_1 \leq \delta^{-1} C^\circ} 0.5\|\lambda\|_2^2 = 0.5(C^\circ / \delta)^2. \]

Using Jensen inequality and previous results we get

\[
E[\|\lambda^{t+1}\|_2] \leq \max \left\{ \sqrt{(C^\circ / \delta)^2 + 2\eta C^\circ}, \|\lambda^1\|_2 \right\}
\]

Finally, since \( \|\lambda\|_{\infty} \leq \|\lambda\|_2 \) for any \( \lambda \in \Lambda \) is immediate that \( E[\|\lambda^t\|_\infty] \leq \max \left\{ \sqrt{(C^\circ / \delta)^2 + 2\eta C^\circ}, \|\lambda^1\|_{\infty} \right\} \)

for all \( t \in [T] \) concluding the proof. \( \square \)

26
6.6 Proof of Proposition 4

Proof. Since \( \alpha_k \neq -\infty \) by statement, Proposition 6 shows \( \dot{h}_k(\lambda_{t+1}) = \dot{h}_k(\lambda_t) - \eta \tilde{g}_k^t \) for any \( t \in [T] \), which implies that \( \dot{h}_k(\lambda_{t+1}) - \dot{h}_k(\lambda_t) = -\eta \sum_{t=1}^{\tau_A} \tilde{g}_k^t \) regardless of the \( \tau_A \) value. Then, using the definition of \( \tilde{g}^t \) we get

\[
\sum_{t=1}^{\tau_A} \left( b_k(1(\lambda_k \geq 0) + \alpha_k 1(\lambda_k < 0)) - c_k(z_t^t; \theta^t, w_t^t) \right) = \frac{\dot{h}_k(\lambda_1) - \dot{h}_k(\lambda_{\tau_A+1})}{\eta} + \sum_{t=1}^{\tau_A} c_k(z_t^t; \theta^t, w_t^t) - c_k(z_t^t; \theta^*, w_t^t).
\]

Now, given that \( (1(\lambda' \geq 0) + \alpha_k 1(\lambda' < 0)) \geq \alpha_k \) for any \( \lambda' \in \mathbb{R} \) and that \( \tau_A \leq T \) by definition, we have

\[
\sum_{t=1}^{\tau_A} \left( b_k(1(\lambda_k \geq 0) + \alpha_k 1(\lambda_k < 0)) \right) + (T - \tau_A) \alpha_k b_k \geq T \alpha_k b_k.
\]

Combining the previous results and taking expectation we get

\[
T \alpha_k b_k - \mathbb{E} \left[ \sum_{t=1}^{\tau_A} c_k(z_t^t; \theta^*, w_t^t) \right] \leq \frac{\dot{h}_k(\lambda_1) - \mathbb{E} \left[ \dot{h}_k(\lambda_{\tau_A+1}) \right]}{\eta} + \mathbb{E}[T - \tau_A] \alpha_k b_k + \mathbb{E} \left[ \sum_{t=1}^{\tau_A} c_k(z_t^t; \theta^t, w_t^t) - c_k(z_t^t; \theta^*, w_t^t) \right].
\]

Finally, we conclude the proof by using Proposition 7 and the definition of \( C_h \).  \( \square \)
7 Extra Experimental Details and Results

7.1 Bidding Experiment

This experiment is based on data from Criteo Diemert et al. (2017). Criteo is a Demand-Side Platform (DSP), which are entities which bid on behalf of hundreds or thousands of advertisers which set campaigns with them. The dataset from Diemert et al. (2017) contains millions of bidding logs during one month of Criteo’s operation. These bidding logs are all logs in which Criteo successfully acquired ad-space for its advertising clients through real-time second-price auctions (each log represents a different auction and ad-space). Each of these auctions occur when a user arrives to a website, app, etc., and each user is shown one ad few milliseconds after its “arrival”. Each bidding log contains:

1. Nine anonymized categorical columns containing characteristics of the ad-space and (possibly) about the user who has just “arrived”.
2. The price Criteo paid for the ad-space, which corresponds to the second highest bid submitted to each auction.
3. The day of the auction and the advertiser whose ad was shown in the ad-space (the day is not included directly in the dataset, but appears in a Jupyter Notebook inside the compressed file that contains the dataset).
4. If a conversion occurs after the ad was shown, i.e., if the corresponding user performed an action of interest for the advertiser after watching the advertiser’s ad. The dataset can be downloaded from https://ailab.criteo.com/criteo-attribution-modeling-bidding-dataset.

The experiment was performed as follows. We used the first 21 days of data as training, the next two days as validation, and the remaining seven days as test. The training data was used only to train a neural network to predict the probability of a conversion occurring. The model architecture was taken from Pan et al. (2018) and uses as features the nine anonymized categorical columns, the day of the week, and an advertiser id to make a prediction if a conversion would occur or not. Parameters to be tuned for the neural network were the step-size for the Adam solver, embedding sizes, and other two specific network attributes (in total we tried 120 configurations). Once found the trained model with highest validation AUC (Area Under the Curve), we took this model predictions as if they were the real probabilities of a conversion occurring for unseen data. By having the advertiser id as an input on the model, we can get conversion probability estimates for all advertisers even when Criteo bid on behalf of only one advertiser per bidding log. The advertisers pay the DSP, in our context the bidder, each time the DSP bids on behalf of them. The payment corresponds to the probability of conversion times a known fixed value. The general simulator scheme for this experiment is shown in Algorithm 4.

Algorithm 2 can be naturally incorporated in the simulator scheme by using the online optimization component of it to obtain \((z^t, k^t)\) of Step 3. of the simulator. We only need the online optimization component of Algorithm 2 as we do not need to learn the distribution of the highest competing (mp) to solve Step 3. of Algorithm 2 (shown in Algorithm 3). We compare the performance of Algorithm 2 to using the Greedy Heuristic 5. When \(\gamma = 1\), Algorithm 5 bids ‘truthfully’ on behalf of the advertiser with the highest valuation. This would be the optimal strategy if the advertisers had ‘infinite’ budgets and no lower bound requirements. Then, we can think of \(\gamma\) as a way to increase/decrease the bids in order to take the budgets into account. (For this example, we can think of Algorithm 2 as an online algorithm for obtaining \(\gamma\) variables per advertiser.)

Our test set contains 21073 iterations and 130 advertisers. (The original dataset had 700 advertisers but we removed all advertisers who appeared in less than 10,000 logs in either the training or
Algorithm 4 Simulator Scheme

Input: Trained conversion prediction model $\sigma$, the set of all test bidding logs $X_{test}$, $T$ the number of test bidding logs, $q \in \mathbb{R}_{+}^{K}$ the vector of payment per conversion values for the advertisers, \( \{mp^t\}_{t=1}^T \) the price Criteo paid for each ad spot in the test set in order.

for $t = 1, \ldots, T$ do
  1. Read the $t$ bidding test log and $mp^t$.
  2. Use model $\sigma$ to obtain estimated conversion probabilities conv.prob. Take $r^t_k = \text{conv.prob}_k \cdot q_k$ for all $k \in K$.
  3. Using vector $r^t$ and previous history, obtain $(z^t, k^t)$ a pair of submitted bid and advertiser to bid on behalf of.
  4. If $z^t \geq mp^t$ then the auction is won, advertiser $k^t$ pays $r^t_k$ to the bidder (the DSP), the bidder pays $mp^t$ for the ad spot and obtains $r^t_k - mp^t$ as profit.
end for

Algorithm 5 Greedy Heuristic($\gamma$)

Input: Vector $r \in \mathbb{R}_{+}^{K}$ and $\gamma > 0$.

Let $K'$ be the set of advertisers with non depleted budgets. If $K' = \emptyset$ do not bid, otherwise bid on behalf of $k^* \in \text{arg max}_{k \in K'} r_k$ the amount $\gamma r_{k^*}$.

validation plus test data.) Each iteration of the simulator scheme uses a batch of 128 test logs. The total budget of an advertiser is the total amount Criteo spent bidding on behalf of that advertiser in the test logs multiplied by 100. We run Algorithm 2 using traditional subgradient descent trying the fixed step sizes $\{1 \times 10^{-i}\}_{i=0}^3 \cup \{0.5 \times 10^{-i}\}_{i=0}^3$ and $\{0.25 + 0.05 \times i\}_{i=0}^{25}$ as $\gamma$ parameters for the Greedy Heuristic 5. We run 100 simulations for each parameter and method pair. Each simulation is defined by the price advertisers would pay per conversion, which is the $q$ vector in Algorithm 4. We sample $q_k$ i.i.d. from Uniform$(0.5, 1.5)$ for all $k \in [K]$. We relaxed the ending condition of Algorithm 2 by allowing advertisers to overspend at most on one iteration. After that iteration, we consider an advertiser’s budget as depleted and do not bid on behalf of it until the simulation’s end. The final parameters chosen for Algorithms 2 and 5 were those that achieved the highest average profit.

An advertiser’s budget depletion time correlates with its relative total maximum budget, fact that is shown in Figure 2. The x-axis is in logarithmic scale and shows the proportion of an advertiser budget w.r.t. the highest budget between all advertisers. Observe that as the relative budget increases, the average depletion time gets closer to the simulation end ($T = 21073$).

Finally, we run this experiment using a SLURM managed Linux cluster. We tried 120 parameters combinations for the conversion prediction architecture, running each parameter configuration for 25 epochs. Each parameter configuration took approximately 40 min to run using a Nvidia K80 GPU plus two Intel Xeon 4-core 3.0 Ghz (we used eight GPUs in parallel having a total run time of approximately 12 hours). For the experiment itself, we tried nine different step-sizes to run the subgradient descent step using Algorithm 2 and 26 $\gamma$ values for each configuration running 100 simulations. We used several cluster nodes each having 64GB of RAM and two Xeon 12-core Haswell with 2.3 Ghz per core. If we had used just one node it would have taken approximately 160 hours to run all required configurations.
Figure 2: The x-axis in the figure shows the proportion of an advertiser budget w.r.t. the highest budget between all advertisers (shown on a logarithmic scale).

7.2 Linear Contextual Bandits Experiment

We now describe in detail the methods used to implement Step 1. of Algorithm 2. First, let $y^t$ be the variable that takes the value of one if an action is taken at period $t$ and zero otherwise. Also, remember that $i(t) \in [d]$ is the action taken at period $t$ (if any), and $r^t$ the revenue observed at period $t$. We implemented Step 1. of Algorithm 2 in the following ways.

1. Gaussian Thompson Sampling as in Agrawal and Goyal (2013). Define $B(1) = I_d$ with $I_d$ the identity matrix of size $d$, and $\hat{\theta}^1 = (1/\sqrt{d}, \ldots, 1/\sqrt{d})$. The Thompson Sampling procedure is composed of two steps which are updating a prior and sampling from a Gaussian posterior. We update the prior as follows. If $y^t = 1$, make $B(t + 1) = I_d + \sum_{s \in [t]: y^s = 1} W_{i(s)} W_{i(s)}^T$ and $\hat{\theta}^{t+1} = B(t+1)^{-1}(\sum_{s \in [t]: y^s = 1} W_{i(s)}^T r^s)$, otherwise $B(t+1) = B(t)$ and $\hat{\theta}^{t+1} = \hat{\theta}^t$. After the prior update, we sample $\theta^t$ from $N(\hat{\theta}^t, \nu^2 B(t)^{-1})$ where $N(\cdot, \cdot)$ represents a normal distribution defined by its mean and covariance matrix, and $\nu > 0$ a constant chosen as follows. When no randomness was added to the observed revenue term, we used $\nu = 0.1$ (remember that we could add randomness to both the matrices $W^t$ and the observed revenue separately). When randomness was added to the observed revenue term, we used $\nu = \frac{\text{rev.err}}{10} \times \sqrt{\log T + n}$ with $\text{rev.err} = 0.1$ or 0.5 depending if a Uniform$(-0.1, 0.1)$ or Uniform$(-0.5, 0.5)$ is added to the observed revenue term respectively. (The latter form of choosing $\nu$ was inspired on Agrawal and Goyal (2013) which uses $\nu = R \sqrt{9n \log T}$ to prove a regret bound for Thompson Sampling for linear contextual bandits without constraints.)

2. Least squares. Same as Thompson Sampling as described above, but Step 1. of Algorithm 2 uses $\theta^t = \hat{\theta}^t$. (This update is a core element of many learning approaches for linear contextual bandits Agrawal and Goyal (2013); Agrawal and Devanur (2016) and can be understood as a Least Squares step).

3. Ridge regression. We use the Least Squares procedure as defined above for the first $\sqrt{T}/2$ actions, and then solve a ridge regression problem. We solve a ridge regression problem at
Step 1. of iteration $t$ using the set $\{W_{i(s)}^s, r^s\}_{s \in [t-1]: y^s = 1}$ with an $\ell_2$ penalization parameter of $\alpha = 0.001$.

4. Ridge regression plus error. Same method as above but adds noise to the $\theta^t$ obtained from the ridge regression problem. We add an i.i.d. Uniform($-0.3, 0.3)/\sqrt{\sum_{s=1}^{t} y^s}$ term to each coordinate of $\theta^t$.

5. Known $\theta^*$. Algorithm 2 using $\theta^t = \theta^*$ for all $t \in [T]$.

Figures 3 and 4 show how the different methods perform for $(d \times n)$ being (5, 10) and (50, 50) when $T = 10,000$, respectively. Each element of the x-axis represents a moving window composed of 250 iterations. The x-axis is composed of 9751 ticks. The y-axis shows the average relative revenue obtained in a window with respect to the proportional best revenue that could have been obtained $(\text{OPT}(\mathcal{P}) \cdot \frac{250}{10000})$. Importantly, the number of actions a method takes can vary between windows, which explains the following two facts. First, an initial revenue spike as many actions are taken when a simulation starts. The latter occurs as we took $\lambda_1 = 0$ which makes the cost component in Step 3. of Algorithm 2 zero. Second, a method may obtain a higher average revenue on a window than $\text{OPT}(\mathcal{P}) \cdot \frac{250}{10000}$ if more than 'average' actions are taken on that window.

Tables 2, 3, 4 show the average total relative revenue obtained for the different combinations of $d \times n$ and uncertainty used with respect to $\text{OPT}(\mathcal{P})$. In general, as long as the budget is spent properly, the revenue obtained by the ‘Known $\theta^*$’ method when $W^t = W$ for all $t \in [T]$ should match $\text{OPT}(\mathcal{P})$. The latter as the best action to take is always the same. In the case when we still have $W^t = W$ for all $t \in [T]$, but the observed revenue has randomness, the ‘Known $\theta^*$’ method may obtain a higher total revenue than $\text{OPT}(\mathcal{P})$.

Finally, we run this experiment using a SLURM managed Linux cluster. We used four nodes each having 64 GB of RAM and 20 cores of 2.5Ghz. We parallelized the code to run each combination of experiment setting and simulation number as a different run (the run-time was mostly spent on sampling from a Gaussian distribution for Thompson Sampling and solving Ridge Regression problems with thousands of points). The total running time was 12 hours. Processing the results was done in a local computer (Mac Book Pro 2015 version), spending around 30 minutes to aggregate the results obtained from the cluster.
Figure 3: Moving average revenue for windows of 250 iterations against the proportional best average revenue possible using $d = 5$, $n = 10$. 
Figure 4: Moving average revenue for windows of 250 iterations against the proportional best average revenue possible using $d = 50$, $n = 50$. 
| Method                  | $T = 1,000$   | $d \times n$ | $\alpha$ | $\beta$ | $\gamma$ | $\delta$ | $\epsilon$ |
|-------------------------|--------------|--------------|---------|---------|---------|---------|---------|
| Least Squares           | $5 \times 5$ | 76.2%        | 77.6%   | 84.2%   | 78.9%   | 79.4%   | 79.3%   |
| Ridge Regression        | $5 \times 5$ | 95.2%        | 74.2%   | 21.9%   | 85.4%   | 65.4%   | 19.0%   |
| Ridge Reg. + Perturbation | $5 \times 5$ | 57.6%        | 79.0%   | 85.4%   | 90.4%   | 89.8%   | 83.5%   |
| Known Parameter         | $5 \times 5$ | 99.9%        | 100.1%  | 100.8%  | 92.4%   | 92.4%   | 92.2%   |
| Least Squares           | $5 \times 10$ | 60.9%        | 63.2%   | 73.4%   | 80.1%   | 80.5%   | 82.6%   |
| Ridge Regression        | $5 \times 10$ | 94.2%        | 90.3%   | 51.2%   | 89.4%   | 85.9%   | 48.3%   |
| Ridge Reg. + Perturbation | $5 \times 10$ | 64.5%        | 67.3%   | 76.5%   | 93.0%   | 92.8%   | 90.0%   |
| Known Parameter         | $5 \times 10$ | 100.0%       | 100.0%  | 100.0%  | 95.5%   | 95.5%   | 95.4%   |
| Least Squares           | $10 \times 5$ | 70.9%        | 74.6%   | 78.1%   | 83.7%   | 84.0%   | 82.9%   |
| Ridge Regression        | $10 \times 5$ | 94.5%        | 91.0%   | 50.6%   | 89.0%   | 84.6%   | 47.6%   |
| Ridge Reg. + Perturbation | $10 \times 5$ | 71.0%        | 75.2%   | 78.8%   | 92.5%   | 92.5%   | 89.0%   |
| Known Parameter         | $10 \times 5$ | 100.0%       | 100.0%  | 100.0%  | 95.5%   | 95.5%   | 95.4%   |

Table 2: All percentages shown are the average revenue over 100 simulations divided by the best average revenue achievable (OPT($\mathcal{P}$)).
| $T = 5,000$ | $d \times n$ | (0, 0, 0.0) | (0.1, 0.0) | (0.5, 0.0) | (0.0, 0.1) | (0.1, 0.1) | (0.5, 0.1) |
|--------------|---------------|----------------|----------------|----------------|----------------|----------------|----------------|
| Least Squares | $5 \times 5$ | 76.7% | 79.4% | 87.1% | 91.6% | 91.5% | 90.5% |
| Thompson Sampling | $5 \times 5$ | 98.7% | 88.6% | 42.6% | 93.2% | 80.9% | 36.7% |
| Ridge Regression | $5 \times 5$ | 78.1% | 79.4% | 86.5% | 95.1% | 94.9% | 92.4% |
| Ridge Reg. + Perturbation | $5 \times 5$ | 80.0% | 79.7% | 87.2% | 94.9% | 94.8% | 92.3% |
| Known Parameter | $5 \times 5$ | 100.0% | 100.0% | 99.9% | 95.9% | 95.9% | 96.0% |
| Least Squares | $5 \times 10$ | 61.2% | 63.5% | 75.3% | 93.1% | 93.3% | 92.6% |
| Thompson Sampling | $5 \times 10$ | 97.3% | 96.0% | 71.7% | 95.8% | 93.0% | 68.6% |
| Ridge Regression | $5 \times 10$ | 64.9% | 67.9% | 79.6% | 96.5% | 96.5% | 95.5% |
| Ridge Reg. + Perturbation | $5 \times 10$ | 71.0% | 71.9% | 80.4% | 96.4% | 96.4% | 95.3% |
| Known Parameter | $5 \times 10$ | 100.0% | 100.0% | 100.0% | 97.5% | 97.5% | 97.4% |
| Least Squares | $10 \times 5$ | 71.3% | 72.3% | 80.9% | 93.6% | 93.4% | 93.4% |
| Thompson Sampling | $10 \times 5$ | 96.0% | 96.4% | 70.4% | 95.2% | 92.1% | 67.1% |
| Ridge Regression | $10 \times 5$ | 71.5% | 73.7% | 81.5% | 96.3% | 96.2% | 95.5% |
| Ridge Reg. + Perturbation | $10 \times 5$ | 77.0% | 80.1% | 83.0% | 96.2% | 96.1% | 95.3% |
| Known Parameter | $10 \times 5$ | 100.0% | 100.0% | 100.0% | 97.0% | 97.0% | 97.0% |
| Least Squares | $10 \times 10$ | 58.9% | 63.3% | 70.0% | 91.0% | 90.9% | 91.3% |
| Thompson Sampling | $10 \times 10$ | 96.2% | 83.9% | 29.5% | 94.2% | 80.7% | 30.8% |
| Ridge Regression | $10 \times 10$ | 59.4% | 63.7% | 70.4% | 95.6% | 95.4% | 93.3% |
| Ridge Reg. + Perturbation | $10 \times 10$ | 69.2% | 69.8% | 74.1% | 95.5% | 95.4% | 93.1% |
| Known Parameter | $10 \times 10$ | 100.0% | 100.0% | 100.0% | 96.7% | 96.6% | 96.5% |
| Least Squares | $25 \times 25$ | 44.6% | 54.0% | 58.6% | 85.6% | 85.6% | 78.3% |
| Thompson Sampling | $25 \times 25$ | 97.2% | 12.6% | 1.2% | 88.6% | 15.0% | 1.9% |
| Ridge Regression | $25 \times 25$ | 44.8% | 54.7% | 60.4% | 93.4% | 91.1% | 76.4% |
| Ridge Reg. + Perturbation | $25 \times 25$ | 64.9% | 64.0% | 66.2% | 93.2% | 90.9% | 76.5% |
| Known Parameter | $25 \times 25$ | 100.0% | 100.0% | 100.0% | 96.7% | 96.6% | 96.5% |
| Least Squares | $25 \times 50$ | 41.5% | 48.1% | 57.5% | 87.7% | 87.4% | 84.4% |
| Thompson Sampling | $25 \times 50$ | 94.6% | 36.2% | 7.3% | 93.0% | 39.9% | 8.6% |
| Ridge Regression | $25 \times 50$ | 43.5% | 49.9% | 68.0% | 95.0% | 94.2% | 87.8% |
| Ridge Reg. + Perturbation | $25 \times 50$ | 55.7% | 58.0% | 74.1% | 94.9% | 94.1% | 87.0% |
| Known Parameter | $25 \times 50$ | 100.0% | 99.9% | 99.6% | 96.5% | 96.5% | 96.5% |
| Least Squares | $50 \times 25$ | 49.3% | 53.0% | 57.8% | 87.6% | 87.9% | 85.3% |
| Thompson Sampling | $50 \times 25$ | 97.8% | 34.3% | 5.5% | 92.3% | 38.9% | 7.1% |
| Ridge Regression | $50 \times 25$ | 52.2% | 55.3% | 58.4% | 94.6% | 93.9% | 86.8% |
| Ridge Reg. + Perturbation | $50 \times 25$ | 66.0% | 65.7% | 67.8% | 94.4% | 93.7% | 87.1% |
| Known Parameter | $50 \times 25$ | 100.0% | 100.0% | 100.1% | 96.0% | 96.0% | 96.0% |
| Least Squares | $50 \times 50$ | 41.9% | 52.7% | 60.4% | 85.8% | 86.2% | 79.6% |
| Thompson Sampling | $50 \times 50$ | 96.4% | 10.0% | 1.8% | 89.7% | 14.3% | 2.7% |
| Ridge Regression | $50 \times 50$ | 43.6% | 53.2% | 68.2% | 94.0% | 91.5% | 77.9% |
| Ridge Reg. + Perturbation | $50 \times 50$ | 59.9% | 61.3% | 71.8% | 93.7% | 91.4% | 77.8% |
| Known Parameter | $50 \times 50$ | 100.0% | 100.0% | 100.2% | 95.5% | 95.5% | 95.5% |

Table 3: All percentages shown are the average revenue over 100 simulations divided by the best average revenue achievable ($\text{OPT}(P)$).
| $T = 10,000$ | $d \times n$ | (0.0, 0.0) | (0.1, 0.0) | (0.5, 0.0) | (0.0, 0.1) | (0.1, 0.1) | (0.5, 0.1) |
|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| Least Squares | 5 × 5 | 76.8% | 79.7% | 85.4% | 94.7% | 94.6% | 93.7% |
| Thompson Sampling | 5 × 5 | 98.8% | 92.4% | 52.8% | 95.4% | 85.8% | 47.0% |
| Ridge Regression | 5 × 5 | 78.2% | 79.7% | 87.0% | 96.5% | 96.4% | 95.0% |
| Ridge Reg. + Perturbation | 5 × 5 | 80.1% | 80.0% | 88.6% | 96.4% | 96.4% | 95.0% |
| Known Parameter | 5 × 5 | 100.0% | 100.0% | 100.2% | 97.0% | 97.0% | 97.1% |
| Least Squares | 5 × 10 | 61.2% | 63.5% | 75.8% | 95.9% | 95.9% | 95.4% |
| Thompson Sampling | 5 × 10 | 96.8% | 97.3% | 79.0% | 97.2% | 95.1% | 76.1% |
| Ridge Regression | 5 × 10 | 65.0% | 67.8% | 76.8% | 97.5% | 97.5% | 97.0% |
| Ridge Reg. + Perturbation | 5 × 10 | 70.4% | 71.7% | 81.0% | 97.5% | 97.5% | 97.0% |
| Known Parameter | 5 × 10 | 100.0% | 100.0% | 100.1% | 98.2% | 98.2% | 98.2% |
| Least Squares | 10 × 5 | 71.4% | 73.1% | 81.7% | 95.9% | 95.9% | 95.4% |
| Thompson Sampling | 10 × 5 | 96.7% | 97.7% | 77.7% | 96.8% | 94.3% | 74.6% |
| Ridge Regression | 10 × 5 | 71.6% | 75.0% | 82.4% | 97.3% | 97.3% | 96.8% |
| Ridge Reg. + Perturbation | 10 × 5 | 76.4% | 80.2% | 83.3% | 97.3% | 97.3% | 96.6% |
| Known Parameter | 10 × 5 | 100.0% | 100.0% | 100.0% | 97.8% | 97.8% | 97.8% |
| Least Squares | 10 × 10 | 59.0% | 64.5% | 71.0% | 94.5% | 94.2% | 93.5% |
| Thompson Sampling | 10 × 10 | 96.4% | 89.0% | 38.8% | 96.0% | 86.3% | 40.5% |
| Ridge Regression | 10 × 10 | 59.4% | 65.2% | 71.8% | 96.8% | 96.7% | 95.2% |
| Ridge Reg. + Perturbation | 10 × 10 | 68.9% | 70.4% | 73.0% | 96.7% | 96.6% | 95.0% |
| Known Parameter | 10 × 10 | 100.0% | 100.0% | 100.1% | 97.5% | 97.5% | 97.5% |
| Least Squares | 25 × 25 | 44.5% | 53.7% | 67.1% | 91.4% | 91.2% | 84.7% |
| Thompson Sampling | 25 × 25 | 98.4% | 18.5% | 1.8% | 92.3% | 21.2% | 2.7% |
| Ridge Regression | 25 × 25 | 44.7% | 54.7% | 65.8% | 95.3% | 94.0% | 83.4% |
| Ridge Reg. + Perturbation | 25 × 25 | 65.8% | 63.9% | 69.6% | 95.1% | 94.0% | 83.6% |
| Known Parameter | 25 × 25 | 100.0% | 100.0% | 100.0% | 96.2% | 96.2% | 95.9% |
| Least Squares | 25 × 50 | 41.6% | 48.0% | 58.0% | 92.7% | 92.7% | 90.4% |
| Thompson Sampling | 25 × 50 | 97.8% | 46.3% | 10.4% | 95.4% | 50.8% | 11.8% |
| Ridge Regression | 25 × 50 | 43.6% | 49.5% | 67.1% | 96.4% | 96.0% | 91.1% |
| Ridge Reg. + Perturbation | 25 × 50 | 57.7% | 59.2% | 71.3% | 96.3% | 96.0% | 91.2% |
| Known Parameter | 25 × 50 | 100.0% | 100.0% | 100.0% | 97.4% | 97.4% | 97.4% |
| Least Squares | 50 × 25 | 49.3% | 53.6% | 58.8% | 92.5% | 92.8% | 90.5% |
| Thompson Sampling | 50 × 25 | 98.6% | 44.8% | 7.9% | 94.8% | 50.2% | 10.3% |
| Ridge Regression | 50 × 25 | 52.3% | 55.1% | 65.1% | 96.1% | 95.7% | 91.3% |
| Ridge Reg. + Perturbation | 50 × 25 | 63.9% | 62.6% | 69.9% | 96.0% | 95.7% | 91.1% |
| Known Parameter | 50 × 25 | 100.0% | 100.0% | 100.0% | 97.0% | 97.0% | 97.1% |
| Least Squares | 50 × 50 | 43.2% | 51.2% | 59.5% | 91.4% | 91.5% | 85.8% |
| Thompson Sampling | 50 × 50 | 98.1% | 13.2% | 2.3% | 93.1% | 19.7% | 3.5% |
| Ridge Regression | 50 × 50 | 44.9% | 52.9% | 65.0% | 95.6% | 94.5% | 84.9% |
| Ridge Reg. + Perturbation | 50 × 50 | 59.3% | 63.2% | 67.7% | 95.5% | 94.4% | 85.2% |
| Known Parameter | 50 × 50 | 100.0% | 100.0% | 99.9% | 96.7% | 96.7% | 96.8% |

Table 4: All percentages shown are the average revenue over 100 simulations divided by the best average revenue achievable ($\text{OPT}(P)$).