# Joint Online Learning and Decision-making via Dual Mirror Descent

**Alfonso Lobos** [1] **Paul Grigas** [1] **Zheng Wen** [2]

## 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 cost 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 & Popescu, 2003; Boyd & 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. Also, our setup allows the cost and revenue functions to take negative values which is key for some problems in chemistry or finance.

### 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 used to measure 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 an “infeasible sequence of context vector arrivals" for which it is impossible to satisfy the lower cost constraints. These infeasible context vector sequences may occur even if non-positive cost lower bounds

[1] University of California, Berkeley [2] Google DeepMind, Mountain view, California. Correspondence to: Alfonso Lobos <alobos@berkeley.edu>, Paul Grigas <pgrigas@berkeley.edu>, Zheng Wen <zhengwen@google.com>.

*Proceedings of the $38^{th}$ International Conference on Machine Learning*, PMLR 139, 2021.

are used. 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 $\mathcal{O}(\sqrt{T})$ under a Slater condition. Given the generic setup of our problem, this is a novel regret bound in the pure online optimization setting. In the general case with parameter learning, our regret decomposes between terms upper bounded by $\mathcal{O}(\sqrt{T})$ and terms coming from the convergence of the generic parameter learning process.

*(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 algorithm construction. Therefore, our methodology is asymptotically feasible in the pure 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 have since been considered. 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 (Devanur et al., 2011; Jenatton et al., 2016; Chen et al., 2017; Yuan & 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 objective function 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 levels of constraint violations have been achieved (Mahdavi et al., 2012; Jenatton et al., 2016; Chen et al., 2017; Yuan & Lamperski, 2018). In the i.i.d. setting and under linear constraints, 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, 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 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 (Balseiro et al., 2020a). Compared to Balseiro et al. (2020b;a), our work generalizes their setup by allowing "infeasible context vectors," which are possible since we consider cost lower bounds constraints to the problem formulation and allow the cost and revenue functions to take negative values. The latter generalization motivated the novel and stricter benchmark proposed here. The proposed benchmark can be informative when previous benchmarks, such as the best dynamic solution, are not (our benchmark generalizes the best dynamic solution benchmark). In addition, given the generality of our setup, we bound the possible worst-constraint violation for the lower cost constraints and propose a different way to bound the dual variables, which we achieve through a Slater condition. Also, our incorporation of generic parameter learning into the online decision-making setup is novel.

A stream of literature studying a similar problem to ours considers "Bandits with Knapsacks" (BwK) and extensions thereof. 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 actions and observes a reward and a cost vector. As in our setting, 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 Immorlica et al. (2019) for the adversarial case). Assuming concave reward functions, Agrawal & Devanur (2014) proposes an upper-confidence bound type of algorithm which achieves sublinear rates of regret and constraint violations. Badanidiyuru et al. (2018) proposes a primal-dual algorithm to solve BwK with has a sublinear regret. By problem construction, their cost constraints are satisfied. Compared to this literature stream, our work allows an *arbitrary* action space and extends the literature as commented in the paragraph above for Balseiro et al. (2020b;a).

## 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 $(x^1, \dots, x^N) \in A^1 \times \cdots \times A^N$ represents $x^i \in A^i$ for all $i \in [n]$. We reserve capital calligraphic letters to denote sets. For any $x \in \mathbb{R}^N$, $[x]_+ := (\max\{x_1, 0\}, \dots, \max\{x_N, 0\})$ and $\mathbb{1}(x \in A) := 1$ 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 $x$.

## 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)* $z^t \in \mathcal{Z} \subseteq \mathbb{R}^d$, 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)* $w^t \in \mathcal{W} \subseteq \mathbb{R}^m$, 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 $f(\cdot;\cdot,\cdot) : \mathcal{Z} \times \Theta \times \mathcal{W} \to \mathbb{R}$ denote the revenue function and let $c(\cdot;\cdot,\cdot) : \mathcal{Z} \times \Theta \times \mathcal{W} \to \mathbb{R}^K$ denote the collection of $K$ different cost functions. We assume that these functions are bounded, namely for the true revenue function it holds that $\sup_{z \in \mathcal{Z}, w \in \mathcal{W}} f(z; \theta^*, w) \leq \bar{f}$ with $\bar{f} > 0$ and for the cost functions it holds that $\sup_{z \in \mathcal{Z}, \theta \in \Theta, w \in \mathcal{W}} \|c(z; \theta, w)\|_\infty \leq \bar{C}$ 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 $\mathcal{P}$ over $\mathcal{W}$. For example, if $\mathcal{W}$ 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 $w^t \in \mathcal{W}$, then it also observes or otherwise have knowledge of the parametric forms of revenue and cost functions $f(\cdot;\cdot, w^t) : \mathcal{Z} \times \Theta \to \mathbb{R}$ and $c(\cdot;\cdot, w^t) : \mathcal{Z} \times \Theta \to \mathbb{R}^K$. Although the true parameter $\theta^*$ may be unknown to the decision maker at time $t$, whenever a decision $z^t \in \mathcal{Z}$ 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 $(w^1, \dots, w^T)$ of future context vector arrivals and would set the decision sequence $(z^1, \dots, z^T)$ by solving the full observability (or hindsight) problem:

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

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 $\underline{b} > 0$. The vector $b$ can be thought as a resource or budget vector proportional to each period. Then, (1) 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 (1) may be infeasible due 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 (1). However, as long as there is any positive probability of (1) 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 (1) 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:

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

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

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

and let us further define

$$\mathrm{OPT}(\mathcal{P}) := \max_{\gamma \in [0,1]} \mathrm{OPT}(\mathcal{P}, \gamma). \qquad (2)$$

Note that $\mathrm{OPT}(\mathcal{P},0)$ is exactly the expected optimal value of the hindsight problem (1). On the other hand, $\mathrm{OPT}(\mathcal{P},1)$ corresponds to a deterministic approximation of (1) 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 $\mathrm{OPT}(\mathcal{P},0)$ and the expectation problem $\mathrm{OPT}(\mathcal{P},1)$. Importantly, the benchmark we consider is $\mathrm{OPT}(\mathcal{P})$, which considers the *worst case* between these two extremes. It is possible to have $\mathrm{OPT}(\mathcal{P}) = \mathrm{OPT}(\mathcal{P},0)$, $\mathrm{OPT}(\mathcal{P}) = \mathrm{OPT}(\mathcal{P},1)$, $\mathrm{OPT}(\mathcal{P}) = \mathrm{OPT}(\mathcal{P},\gamma)$ for some $\gamma \in (0,1)$, and $\mathrm{OPT}(\mathcal{P}) = -\infty$. It is also possible to have a unique $\gamma$ that maximizes $\mathrm{OPT}(\mathcal{P},\gamma)$ 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 $\mathcal{I}^t := (z^t, \theta^t, w^t, f^t(z^t;\theta^*,w^t), c(z^t;\theta^*,w^t))$ denote the information obtained during period $t$, and let $\mathcal{H}^t := (\mathcal{I}^1,\ldots,\mathcal{I}^t)$ 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 $c(z^t;\theta^*,w^t)$, but there is a possibility of including additional randomness in the observed revenue. In particular, the observed revenue $f^t(z^t;\theta^*,w^t)$ satisfies $f^t(z^t;\theta^*,w^t) = f(z^t;\theta^*,w^t) + \xi_t$ 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

$$\begin{aligned} \theta^t &= A_\theta\left(\mathcal{H}^{t-1}\right), \\ z^t &= A_z\left(f(\cdot;\theta^t,w^t), c(\cdot;\theta^t,w^t), \mathcal{H}^{t-1}\right) \end{aligned} \tag{3}$$

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 1, 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 $\bar{C} := \sup_{(z,\theta,w)\in\mathcal{Z}\times\Theta\times\mathcal{W}} \|c(z;\theta,w)\|_\infty$, which is a constant that we will use as the minimum allowable remaining cost budget. For simplicity we assume that the constant $\bar{C}$ is available although we can easily replace it with an available upper bound.

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.

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

```
Input: Initial estimate $\theta^1 \in \Theta$, and remaining cost budget vector $b^1 \leftarrow Tb$.
for $t = 1, \ldots, T$ do
    1. Update $\theta^t \leftarrow A_\theta\left(\mathcal{H}^{t-1}\right)$.
    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 $z^t \leftarrow A_z\left(f(\cdot;\theta^t,w^t), c(\cdot;\theta^t,w^t), \mathcal{H}^{t-1}\right)$.
    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 $b_k^{t+1} < \bar{C}$ for any $k \in [K]$, break.
end for
```

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 $\mathrm{Regret}(A|\mathcal{P}) := \mathrm{OPT}(\mathcal{P}) - R(A|\mathcal{P})$. 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

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

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

$$\begin{aligned} p(\lambda) &:= \sum_{k\in[K]} b_k([\lambda_k]_+ - \alpha_k[-\lambda_k]_+) \\ D(\lambda;\theta) &:= \mathbb{E}_{\mathcal{P}}[\varphi(\lambda;\theta,w)] + p(\lambda). \end{aligned}$$

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 2.1 that $D(\cdot;\theta^*)$ can be used to obtain an upper bound on $\mathrm{OPT}(\mathcal{P})$.

**Proposition 2.1.** *For any $\lambda \in \Lambda$, it holds that $\mathrm{OPT}(\mathcal{P}) \leq TD(\lambda;\theta^*)$.*

Next, Proposition 2.2 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 2.2.** *Let $\lambda \in \Lambda$, $\theta \in \Theta$, and $w \in \mathcal{W}$ be given. Define $\tilde{g}(\lambda;\theta,w) \in \mathbb{R}^K$ by $\tilde{g}_k(\lambda;\theta,w) := -c_k(z(\lambda;\theta,w);\theta,w) + b_k\left(\mathbb{1}(\lambda_k \geq 0) + \alpha_k \mathbb{1}(\lambda_k < 0)\right)$ for all $k \in [K]$. Then, if $w \sim \mathcal{P}$, it holds that $\tilde{g}(\lambda;\theta,w)$ is a stochastic subgradient estimate of $D(\cdot;\theta)$ at $\lambda$, i.e., $\mathbb{E}_{\mathcal{P}}[\tilde{g}(\lambda;\theta,w)] \in \partial_\lambda D(\lambda;\theta)$.*

We are now ready to describe our dual mirror descent algorithm. Let $h(\cdot) : \Lambda \to \mathbb{R}$ 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 $h(\lambda) \geq h(\lambda') + \langle \nabla h(\lambda'), \lambda - \lambda' \rangle + \frac{\sigma_1}{2}\|\lambda - \lambda'\|_1^2$ 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 $h_k(\cdot) : \mathbb{R} \to \mathbb{R}$ is a convex univariate function for all $k \in [K]$. Define $V_h(\lambda,\lambda') := h(\lambda) - h(\lambda') - \nabla h(\lambda')^T(\lambda - \lambda')$, 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 & 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 2.2 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 $\lambda_k^t \leftarrow \lambda^t \exp(-\eta_t \tilde{g}_k^t)$ (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.

**Algorithm 2** Online Learning and Decision-making via Dual Mirror Descent

```
**Input:** Initial estimate $\theta^1 \in \Theta$, remaining cost budget vector $b^1 = Tb$, and initial dual solution $\lambda^1$.
**for** $t = 1, \dots, T$ **do**
    1. Update $\theta^t \leftarrow A_\theta\left(\mathcal{H}^{t-1}\right)$.
    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. Make primal decision $z^t \leftarrow z(\lambda^t;\theta^t,w^t)$, i.e.,
           $z^t \in \arg\max_{z\in\mathcal{Z}} f(z;\theta^t,w^t) - (\lambda^t)^T c(z;\theta^t,w^t)$.
    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 $b_k^{t+1} < \bar{C}$ for any $k \in [K]$, **break**.
    6. Obtain dual stochastic subgradient $\tilde{g}^t$ where $\tilde{g}_k^t \leftarrow -c_k(z^t;\theta^t,w^t) + b_k\left(\mathbb{1}(\lambda_k \geq 0) + \alpha_k \mathbb{1}(\lambda_k < 0)\right)$ for all $k \in [K]$.
    7. Choose "step-size" $\eta_t$ and take dual mirror descent step
           $\lambda^{t+1} \leftarrow \arg\min_{\lambda\in\Lambda} \lambda^T \tilde{g}^t + \frac{1}{\eta_t} V_h(\lambda,\lambda^t)$.
**end for**
```

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. Considering only the revenue function for simplicity, let us discuss a general form of performing Step 1. at any iteration $t \in [T]$. At iteration $t$, we have observed the set $\{z^s, w^s, f^s(z^s;\theta^*,w^s)\}_{s=1}^t$. Then, Step 1. at iteration $t$ could aim to solve or approximate a solution of $\min_{\theta\in\Theta} \sum_{s=1}^T \|f(z^s;\theta;w^s) - f^s(z^s;\theta^*;w^s)\|^2$. Interesting parametric forms could be $f(z;\theta,w) = \exp(-\sum_{i=1}^N w_i \theta_i z_i)$, $f(z;\theta,w) = z^T(ww^T)\theta$, or others. In particular, in Section 4 we study a linear contextual bandits experiment with bounds on the number of actions. There, the regression problem shown above corresponds to a ridge regression problem (once a quadratic penalization term is added), but we also suggest a Thompson Sampling type to use as Step 1. as well. How to perform Step 1. such that $\theta^*$ is learned properly depends both on the problem structure and underlying randomness of the data arrivals.

## 3. Regret Bound and Related Results

In this section, we present our main theoretical result, Theorem 3.1, 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 3.1. 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 $\mathbb{E}\left[\|\nabla h(\lambda^t)\|_\infty\right] \le C_h$ for all $t \in [T]$.*

Note that, in the Euclidean case where $h(\lambda) = \frac{1}{2}\|\lambda\|_2^2$, 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 $\sum_{t=1}^{\tau_A} c_k(z^t;\theta^*,w^t) + \bar{C} > b_k T$.*

Finally, recall that we defined constants $\bar{f} > 0$, $\bar{C} > 0$, $\underline{b} > 0$ and $\bar{b} > 0$ such that $\sup_{z\in\mathcal{Z},w\in\mathcal{W}} f(z;\theta^*,w) \le \bar{f}$, $\sup_{z\in\mathcal{Z},\theta\in\Theta,w\in\mathcal{W}} \|c(z;\theta,w)\|_\infty \le \bar{C}$, $\underline{b} := \min_{k\in[K]} b_k$ and $\bar{b} := \max_{k\in[K]} b_k$. Also, $\sigma_1$ refers to the strong convexity constant of $h(\cdot)$. We are now ready to state Theorem 3.1, which presents our main regret bound.

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

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

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

**Corollary 3.1.** *Consider the same setting as Theorem 3.1 with $\theta^t \leftarrow \theta^*$ for all $t \in [T]$. Then, $\Delta_{Learn} = 0$ and if we choose $\eta = \gamma/\sqrt{T}$ for some constant $\gamma > 0$ we have* $\mathrm{Regret}(A|\mathcal{P}) \in \mathrm{O}(\sqrt{T})$.

Theorem 3.1 states that the regret of Algorithm 2 can be upper bounded by the sum of two terms: *(i)* a quantity $\Delta_{\mathrm{DM}}$ that relates to the properties of the decision-making algorithm, dual mirror descent, and *(ii)* a quantity $\Delta_{\mathrm{Learn}}$ that relates to the convergence of the learning algorithm $A_\theta$. More generally, $\Delta_{Learn}$ depends on the convergence of $\theta^t$ to $\theta^*$. Corollary 3.1 shows that setting $\eta \leftarrow \gamma/\sqrt{T}$ for some constant parameter $\gamma > 0$ when $\theta^*$ is known, *i.e.*, a pure online optimization case, we have $\Delta_{\mathrm{DM}}$ is $O(\sqrt{T})$ and $\Delta_{\mathrm{Learn}} = 0$. Thus, extending the result presented by Balseiro et al. (2020b). Under a stricter version of Assumption 3.1 and assuming the cost functions are Lipschitz in $\theta$, we demonstrate in the supplementary materials that $\Delta_{Learn}$ 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 $\delta_\theta := \mathbb{E}_{\mathcal{P}}[\max_{z\in\mathcal{Z}} \mathrm{res}(z;\theta,w)]$ with $\mathrm{res}(z;\theta,w) := \min\{\|Tb_k - c_k(z;\theta,w)\|_\infty, \|c_k(z;\theta,w) - T\alpha_k b_k\|_\infty\}$ for all $(z,w) \in \mathcal{Z}\times\mathcal{W}$.*

The following proposition uses the average slack to upper bound $C_h$ in Assumption 3.1.

**Proposition 3.1.** *Assume we run Algorithm 2 with a constant "step-size" rule $\eta_t \leftarrow \eta$ for all $t \ge 1$ where $\eta > 0$. Assume also that there exists $\delta > 0$ such that $\delta_\theta \ge \delta$ for all $\theta \in \Theta$, and let $C^{\triangleright} := 2(\eta\frac{(\bar{C}^2+\bar{b}^2)}{\sigma_1} + \bar{f})$. 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 $C_h \le \max\{\|\lambda^1\|_\infty, \sqrt{2}\sqrt{0.5(C^{\triangleright}/\delta)^2 + \eta C^{\triangleright}}\}$.*

The Slater requirement in Proposition 3.1 can be understood as a measure of the feasibility of problem $\mathrm{OPT}(\mathcal{P}, 0)$ for all $\theta \in \Theta$. Also, notice that under the conditions of the proposition and using $\eta_t = \eta = \gamma/\sqrt{T}$ for all $t \in [T]$ we have that $C_h$ is bounded by the maximum of $\|\lambda^1\|_\infty$ and a term in $O(T^{-1/4})$, proving that $C_h$ is in $O(1)$.

### 3.2. Proof Sketch and Cost Feasibility

The proof sketch for Theorem 3.1 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 $\mathbb{E}[\tau_A]$ 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 $\mathbb{E}[\tau_A D(\sum_{t=1}^{\tau_A} \frac{1}{\tau_A}\lambda^t; \theta^*) - \sum_{t=1}^{\tau_A} f(z^t; \theta^*, w^t)]$ 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 $\bar{f}\mathbb{E}[T - \tau_A]$. (Note that the two major steps of our proof mimic those of Balseiro et al. (2020b) 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 3.2 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 $\theta^t$ converges to $\theta^*$ again appear, analogously to Theorem 3.1.

**Proposition 3.2.** *Assume we run Algorithm 2 under Assumption 3.1 using $\eta_t = \eta$ for all $t \geq 1$. For any $k \in [K]$ with $\alpha_k \neq -\infty$ it holds:*

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

## 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. The experiments' code is located at https://tinyurl.com/br3dzeak.

### 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 '$\mathrm{mp}^t$' 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)):

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

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

where $Tb$ represent the maximum the clients would like to spent, and $\alpha \in [0, 1)^K$ the minimum percentage to be spent. The pair $(z^t, x^t) \in \mathbb{R}_+ \times \Delta$ 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 $\mathbb{1}\{z^t \geq \mathrm{mp}^t\}$ to indicate that the bidder wins the auction $t \in [T]$ only if its bid is higher than $\mathrm{mp}^t$. Here we assume $r(\cdot) : \mathcal{W} \to \mathbb{R}_+^K$ 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 $\mathrm{mp}$ , making this experiment fall in the pure online optimization case. The latter occurs as $\varphi(\lambda;(w,\mathrm{mp})) = \max_{(z,x)\in\mathcal{R}_+\times\mathcal{X}} \sum_{k=1}^{K}(r_k(w)(1-\lambda_k)-\mathrm{mp})x_k\mathbb{1}\{z \geq \mathrm{mp}\}$ 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 $\varphi(\lambda;(w,\mathrm{mp}))$.
```

**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). We validated the conversion prediction model parameters choosing those that maximized the validation AUC score. We used the predictions coming from this architecture as if they were the truthful probabilities of conversion. To obtain the $r(w^t) \in \mathbb{R}_+^K$ vectors at testing time we used the conversion predictions predictions coming form this architecture times a predefined price per 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. In all simulations $T = 21073$ (more experimental details in the supplement).

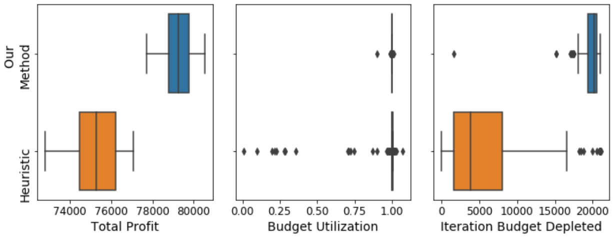

*Figure 1.* Box plots of the total profit obtained, and average budget utilization and budget depletion iteration per advertiser over 100 simulations. Budget utilization corresponds to the percentage of the total budget that an advertiser spent. If an advertiser never depleted its budget, its depletion time equals the simulation length.

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 $\mathbb{E}[r^t] = (W_{i(t)}^t)^T\theta^*$, where $W_{i(t)}^t$ is the $i(t)^{th}$ row of $W^t$ and $\theta^*$ is an unknown parameter. We assume that $r^t = (W_{i(t)}^t)^T\theta^* + \epsilon$ 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 $0.5T$. The agent does not know the distribution $\mathcal{W}$ over which $W^t$ is sampled (but knows that they are sampled i.i.d.). We can model this problem as having $\mathcal{Z} = \{z \in \mathbb{R}_+^K : \sum_{i=1}^T z_i \leq 1\}$, $\mathcal{W}$ being the set of possible matrix arrivals, $f(z;\theta,W^t) = ((W_1^t)^T\theta,\ldots,(W_d^t)^T\theta)^T z$, and $c(z;\theta,W^t) = (\rho,\ldots,\rho)\odot z$. Even when $\mathcal{Z}$ 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. $\mathrm{Uniform}(-0.5, 0.5)$ random variables. Both $\theta^*$ and $W$ are then normalized to satisfy $\|\theta^*\|_2 = 1$ and $\|W_{d'}\|_2 = 1$ 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 $\mathrm{Uniform}(-0.1, 0.1)$ random variable. Also, given a selected action $i(t) \in [d]$ on period $t \in [T]$, the observed revenue is either $W_{i(t)}^T \theta^*$ or $W_{i(t)}^T \theta^*$ plus either a $\mathrm{Uniform}(-0.1, 0.1)$ or $\mathrm{Uniform}(-0.5, 0.5)$ 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 & 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 $\mathrm{OPT}(\mathcal{P}) = \mathrm{OPT}(\mathcal{P}, 0)$, and each optimization problem inside $\mathrm{OPT}(\mathcal{P}, 0)$ is a bag problem. Please refer to the supplement for detailed descriptions of the methods, more experimental details, and the proof that $\mathrm{OPT}(\mathcal{P}) = \mathrm{OPT}(\mathcal{P}, 0)$.

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 $\mathrm{Uniform}(-0.5, 0.5)$ is added to the observed revenue and that i.i.d. $\mathrm{Uniform}(-0.1, 0.1)$ elements were added element-wise to $W^t$ 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 $\mathrm{OPT}(\mathcal{P})$ as the optimal action is always the same.)

| T = 10000, (d × 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 $\mathrm{Uniform}(-0.5, 0.5)$ is added to the observed revenue and that $i.i.d.$ $\mathrm{Uniform}(-0.1, 0.1)$ elements were added to each coordinate of $W^t$ for each $t \in [T]$.

Table 1 shows interesting patterns. First, Thompson Sampling implemented as in (Agrawal & 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 shown in Table 1 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.

## References

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Pan, J., Xu, J., Ruiz, A. L., Zhao, W., Pan, S., Sun, Y., and Lu, Q. 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*, pp. 1349–1357. International World Wide Web Conferences Steering Committee, 2018.

Wei, X., Yu, H., and Neely, M. J. 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*, pp. 3–4, 2020.

Yuan, J. and Lamperski, A. Online convex optimization for cumulative constraints. In *Advances in Neural Information Processing Systems*, pp. 6137–6146, 2018.